International Journal of Engineering & Technology, 7 (4) (2018) 5673-5678

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET
doi: 10.14419/ijet.v7i4.14224
Research paper

Extensive analysis of techniques in data streams

Ramesh Balasubramaniam [1] *, K. Nandhini [1]

[1] PG & Research Dept. of Computer Science, Chikkanna Govt. Arts College (Bharathiyar University), Tirupur, India
**Corresponding author E-mail: rameshbala50@gmail.com*

Abstract

Applications are generating huge capacities (volume) of data at high speeds (velocity) from various sources such as images, text, audio, and video (variety). Big data streams are generated by many applications in today's world like IoT devices, online purchases, internet traffic, social media, stock exchanges and more. The data source decides whether the processing should be by batch or stream. It is impossible and unnecessary to record all incoming data, hence the need for data reduction techniques in data streaming. These techniques (sampling, sketching, hashing, dimension reduction, and more) enable us to narrow down the big data to relevant data. This data sampled, filtered, hashed, or processed through other techniques is used as input for data analysts to derive meaningful information. A well-designed Data Stream Management System will strike a balance between the right data processing and the cost of processing. This paper highlights the different techniques used in streaming data, related work in that area and the uses in today's world.

Keywords: *Hashing; Sampling; Sketching; Stream Data Model; Streaming Techniques.*

1. Introduction

In the advent of big data and the 3V's - Volume, Velocity, and Variety (which keeps growing and now includes 6V's - Value, Veracity, and Variability) data is being generated continuously (infinite) by many data sources. Data streams date from satellite information processing systems to today's social networks Internet of Things (IoT). In today's smart environment, data is also classified as periodic or event triggered. The periodic devices generate big data streams because they produce a constant amount of data at regular intervals. Event triggered devices on the other hand are activated only when a certain event is encountered. An event triggered device might in turn trigger other devices which leads to a bigger network of data streaming too. Applications that generate big data streams are online purchases, Social networks, Stock market trading, clickstreams, sensors, GPS systems and more. As Fig 1 indicates stream computing delivers real-time analytic processing constantly on in-motion variable data. It enables descriptive and predictive analytics to support real time decisions.

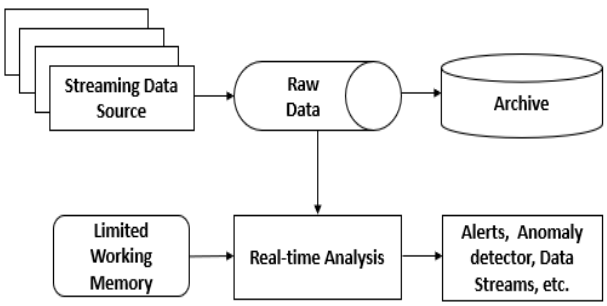

Fig. 1: Data Stream Management System.

This paper's focus is on techniques used in streaming data that provide meaningful data to analysts. First, is the introduction to the stream data model which includes Data Stream Management Systems (DSMS), stream sources, queries, and stream processing. Secondly, is the presentation of several techniques used in streaming algorithms like sketching, clustering, hashing, and more. Finally, in the conclusion are the findings and next steps.

2. Stream data model

There are two problems [15] that a Relational database management system (RDBMS) cannot handle when it comes to streaming data:

- Managing changes in data arrival rate during a specific query lifetime
- The continuous flow of the data stream into the system that cannot be stored in a traditional way; therefore, only the most recent data can be used for answering the queries at a given time point.

To address these issues DSMS was developed by the database community having different semantics associated to the queries [9].
In a Database Management System (DBMS) data is stored on disks and queries are used to define or modify structures and to insert, modify or retrieve the stored data. Data Stream Management System (DSMS) on the other hand has no data stored on any disk storage. The source to the DSMS is a continuous stream of variable (speed unknown) data. If the DSMS is not able to read all data in the source, the unread data is lost. [9] As Table 1 points out, in DBMS data is stored permanently and queries are transient. Whereas, in a DSMS queries are stored permanently in sensors and data is a continuously produced stream [15].
Despite these differences, DSMSs resemble DBMSs, in their processing. All incoming data go through a transformation using common SQL operators like selections, aggregates, joins, and operators defined by relational algebra [6].

Table 1: Comparison between DBMS & DSMS Functional Feature

Feature	DBMS	DSMS

Copyright © 2018 Ramesh Balasubramaniam, K. Nandhini. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Model	Persistent data	Transient data
Table	Set or bag of tuples	Infinite sequence of tuples
Updates	All	Append only
Queries	Transient	Persistent
Query answers	Exact	Often approximate
Query evaluation	Multi-pass	One pass
Operators	Blocking and non-blocking	Non-blocking
Query Plan	Fixed	Adaptive
Data processing	Synchronous	Asynchronous
Concurrency overhead	High	Low

Stream queries: The most popular DBMS query language is the SQL query language. In a DSMS the query language needs to extend to streams. The logic of a query needs to be redefined when applying to data streams hence the introduction of continuous query [15]. The output from a query on a data stream can be another stream or the updation of a permanent table with the contents of the stream. This data is processed as it arrives by queries that are stored permanently. According to Cugola & Margara [6], DSMS can be modelled as a set of standing queries Q, one or more input streams, and four possible outputs:

- The Stream is composed of all the elements of the answer that are produced once and never changed.
- The Store contains the parts of the answer that may be changed or removed at a certain future time point. The stream and the store together define the current answer to Q.
- The Scratch represents the working memory of the system, where it is possible to store data useful to compute the answer but that is not part of the answer.
- The Throw is a sort of recycle bin that is used to throw away unneeded tuples.

Stream processing - batch vs stream: One of the fundamental differences in big data processing is the distinction between batch processing and stream processing. The data source decides whether the processing should be by batch or stream.
An unofficial definition of these two terms [20]:

- Under the batch processing model, data is collected over time, then fed into analytical systems. In other words, you collect a batch of information, then send it in for processing.
- Under the streaming model, data of undefined length is fed into analytics tools piece-by-piece. The processing is usually done in real time. This data will be processed using Stream Processing techniques without having access to all the data.

Batch processing: Batch processing works mostly with big and complex stored data. It works well when you don't need real-time analytical results. The focus is more on processing large volume of information rather than passing data on to analysis. Few scenarios that use batch processing are financial data analysis, last year sales analysis and others.
Stream processing: Stream processing works mostly with latest data that is up to date. It has very fast processing time and is used when you need real-time analytical results. As soon as you feed the data into analytics tools the results are generated instantly. A real-life usage is in fraud detection. If anomalies are detected in transactions a trigger might go off in fraudulent cases that could stop fraud in real time.

3. Techniques used in streaming algorithms

Due to the important role of the Vs of big data there is a need for data reduction - reducing data to a manageable size to uncover maximum knowledge patterns. Reduced data is inferred as much more useful by analysts. It is impossible and irrelevant to analyse raw, redundant, and noisy data that flows from the input stream. A well-designed Data Stream Management System will strike a balance between the right data processing and the cost of processing. To make it beneficial for data analysis, several pre-processing techniques for summarization, sketching, anomaly detection, dimension reduction, noise removal, and outliers detection are applied to reduce, refine, clean, and store a smaller representation of the streaming data especially of unknown length [21]. Different methods for data reduction include sampling, filtering, dimension reduction etc.

3.1. Sampling

Sampling is considered one of the most basic of streaming techniques. A sample collected using random sampling techniques can form as a good basis for statistical inference about the contents of the stream [8]. Simple random sampling (SRS) is one form where a size k is obtained from a population of size n. First a sample element is selected randomly and uniformly from among the n population elements, removed and added to the sample. This sampling step is repeated until k sample elements are obtained. Random sampling is a basic technique used in data streams as many sampling algorithms compute on the sample rather than the entire stream.
Reservoir sampling: Reservoir sampling is a randomized algorithm used to choose k items from an unknown or very large list of size N. For example, products sold on Amazon during the festival seasons. A 'reservoir' is used to store the elements hence the name. According to Vitter [22] the basic idea behind reservoir algorithms is to select a sample of size $\geq$ k, from which a random sample of size k can be generated. So, the reservoir size is k.
Steps to generate a sample:

- The first k records are put into the reservoir [0...k-1]. Remaining records from k+1 are processed sequentially.
- All items from $(k+1)^{th}$ item to n^{th} item are considered. A random number j is generated from 0 to i where i is index of current item in input []. Next, it is checked to see if j already exists in the reservoir. If j is in 0 to k-1, replace reservoir [j] with input[i].

In this method items are given a fair chance to be part of the reservoir regardless of whether they were part of the reservoir initially or not. An algorithm is a reservoir algorithm if it maintains the invariant that after each record is processed a true random sample of size k can be extracted from the current state of the reservoir [22]. At the end of the input stream the final random sample must be extracted from the reservoir.
Bernoulli sampling: This is the simplest of sampling methods. In Bernoulli sampling all elements in the data stream have equal probability of selection (like reservoir sampling) but without the replacement sampling design (as in reservoir sampling) and the inclusion variables are independent. Since each element is considered separately the sample size is not fixed. It follows a binomial distribution where the sample size could be anywhere between 0 to N, where N is the size of the sample. This is a disadvantage especially when the sample size required is small.
Ramos Rojas, Kery, et.al [17] claim that using not just one, but all the sampling algorithms generate a richer understanding of the dataset. For example, random sampling focuses on general features and uncertainty sampling focuses on outliers and features that highlight them. Both sampling methods are complementary to each other, hence used together will produce better insights.

3.2. Sliding window

Another model used in data streams is the sliding window model. The sliding window model expires old items as new items arrive. Two common types of sliding windows [7]:

- Sequence-based windows - which stores the N newest items
- Timestamp-based windows - which store items generated or arrived in the last T time units

Just like a sample, the sliding window technique uses a query not on the entire data stream but only over the recent data of the stream. For example, only sales data from the last week could be considered for query answers and data pertaining to the previous weeks discarded.
Babcock, Babu, Datar, Motwani and Widom [2] state that "sliding windows should not be thought as an approximation technique reluctantly imposed due to the infeasibility of computing over all historical data, but rather as part of the desired query semantics

explicitly expressed as part of the users' query." Sliding windows are well-defined, easily understood, and deterministic, such that users can be confident that what is being given up and random choices is not going to produce a bad approximation.

Most of the world's applications use recent data. It is more relevant to check latest sensor data, traffic records, sales transactions instead of old data. Answers to queries on recent data gives more information and insights during data analysis.

In the below Fig 2 input data stream 7, 3, 9, 2, 7 flow in during Time T1, T4, T6 and T8. If the 5-second sliding window starts processing at Time T1 the output is 7. Since there is no new event/data during T2 and T3 there will be no output. At T4 the output is 7, 3 which falls within the 5 second window and so on. Sliding window will generate output only when a new event/data occurs otherwise no output will be generated.

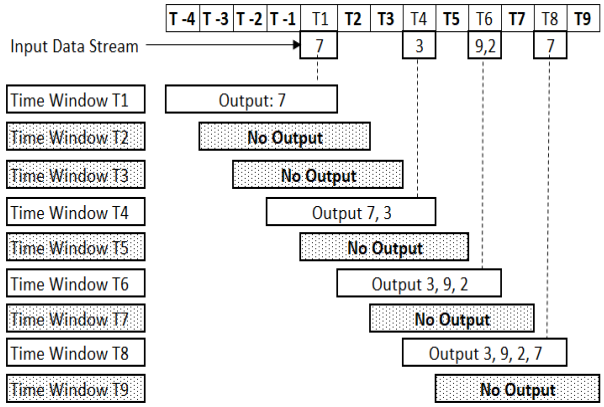

Fig. 2: A 5-Second Sliding Window.

Another windowing techniques used in some scenarios is called "Tumbling Window". Here the query processing is done in a non-overlapping window and an event cannot belong to more than one window. These windowed queries are provided in tools like Amazon Kinesis Data Analytics, Microsoft Azure Stream Analytics.

Sliding window in sampling: Sampling can be obtained from infinite data streams as well using Sliding windows. Unlike Reservoir sampling or Bernoulli sampling which use stationary windows, the task of obtaining a sampling from a sliding window is much harder. The difficulty is because the elements of the sample must be removed as they expire hence a defined size for the sample is not possible. Recollect the sequence-based window and timestamp-based window mentioned previously. Between the two, the better sliding window model used in sampling is the technique which favours recently arrived elements that provide equal- probability sampling within the window.

Let's consider windows W_j each of length n and Elements $W_j = \{e_j, e_{j+1}, e_{j+2}....,. e_{j+n+1}\}$. The many algorithms proposed for sampling from sliding windows need to resolve issues with memory bounds (as stream size is indeterminable) and additional costs which results in performance decrease [8].

Even if a ''complete resampling' is performed where the set of elements in a window are buffered and updated incrementally i.e., W_{j+1} is obtained from W_j by deleting e_j and inserting e_{j+n}, the disadvantages mentioned above still exist. Another sampling method is the 'passive' algorithm here the sample is updated only when the element in a sample expires. This algorithm is like a reservoir algorithm where sample of size k is taken from the first n elements and thereafter a new element is entered into the sample when an expired element is deleted. This way the memory issue is resolved.

Sampling with or without replacement from sequence-based or timestamp-based windows need to provide optimal memory guarantees and reduce complexity-randomized or deterministic [4].

3.3. Filtering

Filtering techniques are used in data streams to reduce the volume of input data during analysis. During filtering errors are detected and corrected so that the impact of errors during the analysis phase is minimized. Filtering in a data stream is to accept the data that meets a criterion and drop the other data.

Implicit filtering: This is when the system itself for the stability of the stream processing performs filtering to avoid overload implicitly by the system. Filtering also known as load-shedding is performed by Data Stream Management Systems. Due to the high volume and velocity of data streams the DSMS needs to employ various methods to reduce the load. The filtering is done in such a way as to minimize negative impact on the accuracy of queries, but this problem of inaccuracy still exists [18].

Explicit filtering: This is when filtering is performed by an evaluating procedure [18]. To avoid the negative impact of implicit filtering it is best to include the filtering condition in the query procedure itself. Depending on the filtering condition the filtering procedures analyse various characteristics of a stream. Bloom Filter algorithm is one such procedure which uses filtering and hashing techniques to eliminate 'undesirable' items.

Filtering is used to identify malicious activity. For example, a financial analyst monitoring commodity might be interested in a sudden rise in stock price or an increase in trading volume. A spike in credit card activity would be of concern to a consumer. A security camera capturing movements of objects, when finds a sudden movement in an object starts capturing subsequent frames for analysis. In places where human lives are at risk - mines, oil and gas drills, and such extreme conditions, machines record data and filtering could be used to provide anomalous movements.

3.4. Hashing

For streaming data, hashing is used as the principle technique in many algorithms. It is used to speed up sorting, searching, inserting, or deleting data. It maps data of arbitrary size to data of fixed size. A hash function reduces search runtime to O (1) when compared to a linear search runtime of O(n) and binary search runtime of O (Log n). Hash functions have the following properties [14]:

- it always returns a number for an object
- two equal objects will always have the same number
- two unequal objects not always have different numbers

A hash table is a collection of items stored in such a way that it makes searching and finding easier. Each slot (position of the hash table) holds an item and is denoted by an integer value starting at 0. A hash function is used to map items to its slots in the hash table. Below Fig 3 shows a hash table of size m=10.

Key	0	1	2	3	4	5	6	7	8	9
Value										

Fig. 3: Hash Table with M Slots Named 0 to 9.

Few simple hash functions are:

- Division-remainder method: An estimate of the number of items in the table is used as a divisor into the item value. This produces a quotient and a remainder. The remainder is considered the hashed value which is used as index to store the item value. This method is liable to produce collisions (addressed in our below example)
- Digit rearrangement method: As the name suggests this method takes digits in the original item and changes the order. For example, in an item the digits in positions 2 and 5 are reversed and the resulting sequence of digits is used as the hashed value or key.
- Folding method: This method works when the values are digits. In this method the value is divided into multiple parts, then added together, and any 4 digits used as the hashed value or key.

As an example, Fig 4 assume a set of integers 20, 34, 56, 88 and 95. The 'Division-Remainder method' is used to calculate the item's hash value as reminder of (h(item)=item/11). The items are input into the hash table at the assigned positions as below.

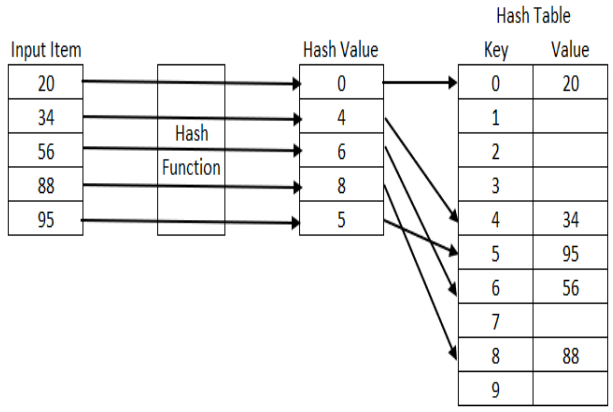

Fig. 4: The Process to Store an Object Using A Hash Function.

There is a possibility of a hash function mapping an object to an already occupied slot. This is called collision. For example, if the item '66' had been the next item in our input values, it would have a hash value of 6 (66%10==6). Since 56 already has hash value of 6 there is a collision.

Collision must be resolved using a collision handling technique like:

- Chaining: This is a simple technique which uses a linked list of records. Each cell in the hash table points to a linked list that has the same hash function value. This way the hash table never fills up and continues to add more elements to the chain. But it does require additional memory outside of the hash table thus if the chain becomes too long search time increases.
- Open Addressing: As mentioned, in open addressing all elements are stored in the hash table itself. When collision occurs, the next open slot is used to store the element. With this technique clustering or load factors need to be avoided. This addresses the low cache performance of linked lists (chaining technique) as 'open addressing' uses the same table.

Cryptography hash functions are used in digital signatures and other authentication applications. Hash functions MD2, MD4, and MD5, reduce digital signatures into a shorter value called a message-digest. The Secure Hash Algorithm (SHA) makes a larger (60-bit) message digest [10].

A good hash function should minimize the number of collisions be easy to compute, and evenly distribute the items in the hash table [14].

3.5. Sketching

The term sketch, as in an artist's sketch, is used to describe algorithms that can extract information from a stream of data or batch data in a single pass (sometimes called "one-touch" processing) using various randomization techniques. A sketch is different from a sample; the latter considers only a portion of the entire data, while the former is computed over all the data. Sketching is mainly used in large-scale computing environments which handle big data such as internet traffic analysis, observing contents of massive databases and such. Sketching query results are an approximation within a user-defined error boundary. The trade-off though is sketch size versus accuracy.

Consider the below example, to find the missing number in a consecutive order of numbers, that has random input. Input Numbers 3,1,6,9,10,2,4,8,7

Using the regular method all numbers are input into an array (n), in sequence and the answer is determined (below)

1	2	3	4		6	7	8	9	10

Answer: 5

Using sketching technique, first the total sum of the dataset (50) is calculated in one-pass and stored in a single cell. Next, using the formula n(n+1)/2 the sum of consecutive numbers is calculated (55). Subtracting the two the missing number is obtained. This way storage is minimized, and performance is faster.

50	Answer: 55 – 50 = 5

To further improve the time performance of a sketch, a sketch computation on a sample of the stream can be performed thus combining sampling and sketching techniques. Rusu & Dobra [19] compare their experimental results of sketching on samples from Bernoulli sampling, sampling with replacement and sampling without replacement with the results of a basic sketch. They show that the accuracy of a sketch computed from a sample is close to the basic sketch and the speed factor is 10 times greater.

There are many frequency-based sketches that solve problems related to estimating functions of frequencies [5].

- Count-Min sketch - counts a group of items and produces an estimate with the minimum of various counts.
- Count-Sketch - like the Count-Min sketch provides an estimate for the value of any individual frequency. The main difference is the accuracy guarantee provided for the estimate.
- Heavy-Hitters - most significant items which have high frequencies. Used in computing most searched query in Google, most sold product on amazon and such.

Open problem in sketching is the usage of different algorithms for different functions. Each sketch is defined to support a certain query. To find out frequently purchased items an algorithm is used and stores the sketch. Again, to find out which pair of items were purchased together another algorithm is used. This way, a query specific procedure is performed on the sketch to obtain the answer.

3.6. Clustering

Clustering is a useful tool for unsupervised classification that helps in grouping similar objects into clusters. Each object in a cluster shares common properties with other objects in the cluster and different properties in different clusters according to a defined distance measure. This is useful to organize streaming data. It helps to understand hidden structures and to represent high-dimensional data in a low-dimensional space [16].

At a high-level clustering can be classified into 3 types: Partitioning, Hierarchical and Density-based.

- Partitioning: Dividing data objects into subsets (clusters) such that all objects in one cluster have similar properties and each data object belongs to exactly one cluster
- Hierarchical: A set of nested clusters organized in the form of a tree
- Density-Based: Grouping of points that are closely packed together as high-density and marking outliers that lie alone as those in low-density regions

Of the various clustering techniques, the partitioning methodology – K-Means is the most popular algorithm. Much has been written and experimented on this method. There exist various variations of k-means algorithms such as k-means++, k-median, k-mods, or k-medoids. Lee, Althoff.et.al. in their paper [12] provide a hardware friendly, multilevel streaming clustering algorithm that can handle big, multidimensional datasets. Using a hardware/software design they achieve high throughput and low resource utilization.

The Traditional K-means algorithm is very simple with the following steps, as described by Shiudkar and Takmare [11]:

1) Select the value of K i.e. Initial centroids
2) Repeat step 3 and 4 for all data points in dataset
3) Find the nearest point from that centroids in the Dataset
4) Form K cluster by assigning each point to its closest centroid
5) Calculate the new global centroid for each cluster

The working of the K-Means Algorithm can be explained better with the help of an example. Fig 5 shows the graphical representation of the working of a K-means algorithm. There are 500 data points with 3 clusters for which the centroids (in red) are determined. The process of determining the centroid repeats until the

best clusters (no change in centroid value) are achieved. In this example it took 8 iterations to reach the final stage. According to the centroid the clusters are formed which gives different clusters to the dataset.

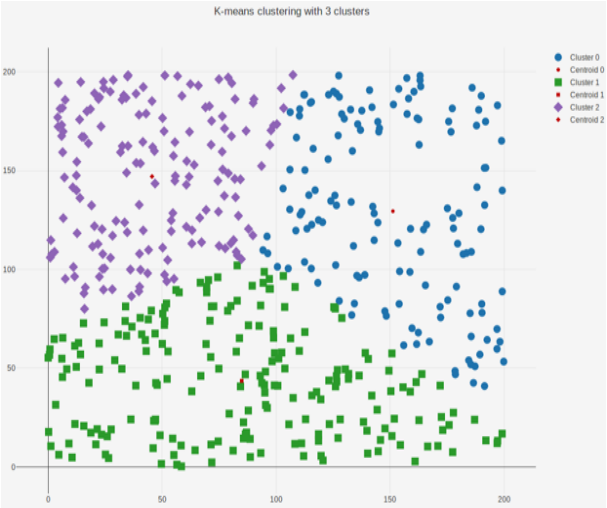

Fig. 5: K-Means Clusters.

Properties of k-means algorithm [11]:

- Efficient while processing large data set
- It works only on numeric values
- The shapes of clusters are convex

This partitioning algorithm is highly sensitive to the initialization phase, noise and outliers. This is overcome using the hierarchical or density-based clustering techniques.

Clustering is used in 'text-analytics' which involves extraction of hidden patterns from large data sets. The input data is cleaned, evaluated, and provided to users as output for interpretation. Other applications that use clustering methods are in pattern recognition, voice mining, image processing, weather report analysis and such.

3.7. Dimension reduction

Dimension Reduction is the process where a high-dimensional data set is reduced to fewer dimensions to improve conceptualization at the same time conveying the information clearly. Visualizing more than 3 dimensions becomes highly improbable to the human eye. Like other data reduction techniques reducing dimensions also helps in reducing noise and outliers and helps in data compression and storage. Analysts working in multi-dimensional fields use this method to remove highly correlated and unnecessary variables.

Suppose, the input data set has measurements of several objects in inches and cm. This could be represented in 2D. But the same information could be converted into 1D removing noise and unwanted data. Similarly, n dimensions of a data set can be reduced to k dimensions ($k < n$), using various dimension reducing techniques.

Feature selection and Feature extraction are two methodologies used in dimension reduction.

- Feature selection is the process where the initial data is reduced for analysis by finding the most relevant information.
- Feature extraction is where useful information is extracted from the initial dataset.

There are many supervised and unsupervised learning techniques in feature extraction like Linear Discriminant Analysis (LDA), Principal Components Analysis (PCA), Isomap and more.

Linear Discriminant Analysis is one of the oldest dimensionality reduction technique that uses a linear combination to categorize groups. Since LDA is a supervised learning technique it is being replaced by techniques based on unsupervised methods like Principal Components Analysis. But LDA is more accurate being a supervised method so it has been implemented in all major packages: R, Python, and MATLAB. Mao, Xue and Maria [23] propose a sliding window approach for the dimensionality reduction for linear discriminant analysis (LDA) on streaming data called Streaming LDA (SLDA). They present a dimension reduction algorithm using various sliding windows that they claim is an improvement over traditional LDA methods in terms of better performance on computational cost and classification accuracy.

One common application of the dimension reduction technique is image processing. Algorithms are used to detect significant features of an image and extract relevant information. There are a very high number of multi-dimensional pixels in an image. Deciding on which pixels to omit and which ones to include is an important decision that these techniques can help with.

3.8. Similarity mining

Similarity mining is a technique used in finding a match of objects. For example, a near similar web page could be considered as plagiarism. Similarity of sets can be defined looking at the relative size of their intersection [13] known as "Jaccard similarity". Data mining or knowledge discovery in databases (KDD) are based on some notion of similarity search - common patterns, clusters of data (similar objects), outliers (opposite of a cluster) and such. This technique is used in multimedia, medical imaging, Computer aided designs (CAD), etc.

Suppose there are two sets P and Q that have four elements in their intersection and a total of eleven elements that appear in P or Q or both. Hence, Jaccard similarity SIM (P, Q) = ⅜

Textual similarity (used to find out plagiarism) is an important feature of similarity mining as mentioned previously. This technique uses words at a character-level to figure out similarity in documents. Search engines use this technique to avoid showing similar pages (almost identical) within the first page of results thus improving their search result credibility.

Collaborative filtering is another process where similarity of sets is very important. This is what online shopping websites use to recommend to shoppers' items that were liked by other shoppers with similar tastes. Two customers could be considered similar if their sets of purchased items have a high Jaccard similarity. But there is no significance in using just a high Jaccard similarity to denote similar tastes, a grouping (clustering) with similarity function would yield better results. Online movie rental company Netflix uses this technique in its customer movie ratings [13]

Shingling of documents is where a set of short strings is constructed from a document to identify lexically similar documents. This way, documents that share common elements in the set can be identified as similar. The most simple and common approach is k-shingle, A substring of length k is found within a document. Then the set of k-shingles is compared with each document to identify if the set appears one or more times within that document.

Fingerprint Matching is performed not using images but using a set of locations in which minutiae are located. A minutia is defined as a place point where a change happens like two ridges merging or a ridge ending. Grid squares are used to locate minutiae and can be compared just like any sets using Jaccard similarity or distance [13]. So, two similar fingerprints would have minutiae lying in exactly the same grid squares. Fingerprint comparison could be:

- Many to one - where a fingerprint is compared to many fingerprints in a database for a match. Used in forensic science - fingerprint found during a robbery.
- Many to many - where the entire database is compared to see if any pair matches an individual. Used for an individual's identification purpose.

Christian Böhm [3] states that the difference in traditional similarity search applications is that these applications do not only raise few, single similarity queries but rather a high number of such queries. So, even though many problems of similarity search have been solved and productive algorithms have been proposed the focus on efficiency is critical. He proposes to change such algorithms using feature transformation approach. The transformation extracts several characterizing properties from the objects and translates them into vectors of a multidimensional space using similarity mining (corresponding to a small distance) of the feature vectors. This

approach relies on distance-based queries (similarity queries) on the feature vectors.

4. Results and discussion

In data mining, streaming data algorithms have always been a challenge due to the large volume of data. Many existing data mining methods cannot be applied directly on data streams because the data needs to be mined in one pass. [1] This puts a strain on space and time and the trade-off is accuracy. All streaming algorithms need to consider the speed at which the data is arriving. In today's world when deciding on the ideal streaming technique for an application distributed data sources, computational resources and communication links need to be taken into consideration. A data stream processor which inculcates a combination of techniques performs stream processing at a rate in pace with the incoming data streams from multiple sources.

Ramos Rojas, Kery, et.al conducted a survey [17] in which 65% of the participants believed that using multiple sampling techniques could improve the quality of their insights and 71% thought it would decrease the time spent on data exploration. Overall, by using not just one streaming technique but a combination of different techniques a richer understanding of the dataset can be generated.

Using hashing and sketching an effective algorithm is the count-min sketch which is a hash-based sketching of the data stream. The count-min sketch counts groups of items and takes the minimum to produce a probabilistic estimate. This solves the problem of frequently accessed web pages, most bought items, heavily traded stocks, and more. The count distinct items algorithm which is used to find the distinct number of elements in a data stream has repetitive numbers (not unique) and uses two techniques - a hashing table with the sliding window technique. Few everyday scenarios where this question could be raised is unique visitors to a website, IP addresses that pass through a router, number of classes in a population, and such. Rusu & Dobra in their paper [19] improve the time performance of sketches by computing the sketch over a sample of the stream. Thus, using sketching and the sampling technique. Their results show that the accuracy of the sketch computed over a small sample is close to the accuracy over the entire data even when sample size is 10% or less of the data size. The basic K-means clustering algorithm itself maybe optimized using Locality Sensitive Hashing (LSH) techniques.

5. Conclusion

Today's data comes from networks, sensors monitoring our environment, the world wide web, transaction data from credit card purchases, stock prices and many more. All data is not relevant hence streaming techniques can be applied to obtain a reduced representation of the dataset that is much smaller in volume, yet closely maintains the integrity of the original data. Memory constraint is an important motivation behind many of the streaming techniques discussed in this paper. In conclusion after extensive research on streaming techniques, a combination of techniques provides more meaningful data for analysis rather than one technique alone, like sampling filtered data or sketching a cluster. Most of these streaming techniques focus only on the storage size of datasets but not on the knowledge hidden within them i.e. not knowledge-oriented reduction. The challenges with streaming data start from the data initialization phase (heterogeneous data) through the right cleaning and extraction techniques (described above) to analysis and interpretation. Future work will be focused on the algorithms used in streaming data and how to derive the maximum benefits out of the algorithms.

References

[1] C.C. Aggarwal, "Data Streams and Algorithms", Kluwer Academic Publishers, Boston, 2007.

[2] B. Babcock, S. Babu, M. Datar, R. Motwani and D. Thomas, "Operator scheduling in data stream systems." The VLDB Journal the International Journal on Very Large Data Bases, Vol. 13, Issue. 4, pp.333-353, 2004. https://doi.org/10.1007/s00778-004-0132-6.

[3] C. Böhm, "Similarity search and data mining: Database techniques supporting next decade's applications".

[4] V. Braverman, R. Ostrovsky, and C. Zaniolo, "Optimal sampling from sliding windows." Journal of Computer and System Sciences, Vol. 78, Issue. 1, pp. 260-272, 2012. https://doi.org/10.1016/j.jcss.2011.04.004.

[5] G. Cormode, "Sketch techniques for approximate query processing." Foundations and Trends in Databases. NOW publishers, 2011.

[6] G. Cugola, and A. Margara, "Processing flows of information: From data stream to complex event processing." ACM Computing Surveys (CSUR), Vol. 44, Issue. 3, Article No. 15, June 2012. https://doi.org/10.1145/2187671.2187677.

[7] L. Golab, "Sliding window query processing, over data streams.", University of Waterloo, 2006.

[8] P.J. Haas, "Data-stream sampling: basic techniques and results." In Data Stream Management, pp. 13-44. Springer, Berlin, Heidelberg, 2016. https://doi.org/10.1007/978-3-540-28608-0_2.

[9] G. Hebrail, "Data stream management and mining." Mining massive data sets for security, pp.89-102, 2008.

[10] A.K. Jain, R. Jones and P. Joshi, "Survey of Cryptographic Hashing Algorithms for Message Signing." IJCST, Vol. 8, Issue. 2, 2017.

[11] MS. Kavitha and S. Takmare, "Review of Existing Methods in K-means Clustering Algorithm.", Vol. 4, Issue. 2, 2017.

[12] D. Lee, A. Alric, R. Dustin, and K. Ryan, "A streaming clustering approach using a heterogeneous system for big data analysis." In Computer-Aided Design (ICCAD), 2017 IEEE/ACM International Conference on, pp. 699-706. IEEE, 2017. https://doi.org/10.1109/ICCAD.2017.8203845.

[13] J. Leskovec, A. Rajaraman, and J.D. Ullman, "Mining of massive datasets", Cambridge university press, chap. 3, 2014. https://doi.org/10.1017/CBO9781139924801.

[14] B.N. Miller, D.L. Bradley, "Problem Solving with Algorithms and Data Structures Using Python" SECOND EDITION. Franklin, Beedle & Associates Inc., 2011.

[15] E. Panigati, F.A. Schreiber, and C. Zaniolo. "Data streams and data stream management systems and languages." In Data Management in Pervasive Systems, pp. 93-111. Springer, Cham, 2015. https://doi.org/10.1007/978-3-319-20062-0_5.

[16] B. Ramesh, R. Nandhini, "Clustering Algorithms – A Literature Review", International Journal of Computer Sciences and Engineering, vol. 5, Issue 10, 2017. https://doi.org/10.26438/ijcse/v5i10.302306.

[17] J.A.R. Rojas, M.B. Kery, S. Rosenthal, A. Dey, "Sampling techniques to improve big data exploration." In 2017 IEEE 7th Symposium on Large Data Analysis and Visualization (LDAV), pp. 26-35. IEEE, 2017. https://doi.org/10.1109/LDAV.2017.8231848.

[18] I. Rozenbaum, "Filtering techniques for data streams". Rutgers The State University of New Jersey-New Brunswick, 2007.

[19] F. Rusu, and A. Dobra, "Sketching sampled data streams." In Data Engineering, 2009. ICDE'09. IEEE 25th International Conference on, pp. 381-392. IEEE, 2009. https://doi.org/10.1109/ICDE.2009.31.

[20] C. Tozzi, "Big Data 101: Dummy's Guide to Batch vs. Streaming Data" syncsort blog, July 25, 2017

[21] M.H. ur Rehman, C.W. Liew, A. Abbas, P.P. Jayaraman, T.Y. Wah, and S.U. Khan, "Big data reduction methods: a survey." Data Science and Engineering 1, no. 4, pp.265-284, 2016.

[22] J.S. Vitter, "Random sampling with a reservoir." ACM Transactions on Mathematical Software (TOMS) 11, no. 1, pp. 37-57, 1985. https://doi.org/10.1145/3147.3165.

[23] M. Ye, X. Li. and M.E. Orlowska, "Supervised dimensionality reduction on streaming data." In Fuzzy Systems and Knowledge Discovery, 2007. FSKD 2007. Fourth International Conference on, vol. 1, pp. 674-678. IEEE, 2007.