Proceedings of the International Conference on Intelligent Sustainable Systems (ICISS 2017)
IEEE Xplore Compliant - Part Number:CFP17M19-ART, ISBN:978-1-5386-1959-9

Data Mining Techniques for IoT and Big Data –A Survey

A.SHOBANADEVI
Research Scholar
Department of Information & Technology
SRM University, Chennai, India
shobanak07@gmail.com

G.Maragatham
Research Supervisor
Department of Information & Technology
SRM University, Chennai, India
maragathamhaarish@gmail.com

***Abstract*—Data Mining is the discovery of "models" of data. Data dredging is a process of derogatory referring to attempts for extracting information that was not supported by the data. Today, data mining is more similar to machine learning and most techniques uses algorithms from machine learning in order to discover unusual events hidden within the large amounts of data. The recent advancements in communication technology, people and the things are becoming increasingly interconnected. The availability of the Internet makes it possible to connect various devices that can communicate with each other and share data. The Internet of Things (IoT) is a new concept that allows users to connect various sensors and smart devices to collect real-time data from the environment. Big Data is a vast amount of data collected from IoT environment and it applies to information that can't be processed or analyzed using traditional tools. Every organization is facing more and more challenges to access a wealth of information and how to get values out of large variety of data. As creation of data is much easier than analyzing it, there is a need for Novel approaches in data mining techniques to deal with huge data. From the perspective of software, the traditional mining algorithms are applicable only for small scale IoT data. This paper first focuses on a review of existing techniques and data mining algorithms that are used to process massive data of IoT and also the limitations are discussed. Second, the work and advancements related to data mining algorithms that are implemented with Hadoop and Spark technology are presented. Third, Hybrid data mining algorithms using MapReduce framework are reviewed. Finally, open research challenges and issues are presented as a conclusion.**

Keywords— Big Data, Data Mining,Hadoop, IoT, MapReduce.

I. INTRODUCTION

Data mining is the process of extracting useful information or patterns from the raw data. Suppose we have a certain amount of data, and we look for events of a certain type within that data. We can expect events of this type to occur, even if the data is completely random, and the number of occurrences of these events will grow as the size of the data grows. These occurrences are "bogus," in the sense that they have no cause other than that random data will always have some number of unusual features that look significant but aren't. A theorem of statistics, known as the Bonferroni correction gives a statistically sound way to avoid most of these bogus positive responses to a search through the data .Without going into the statistical details, we offer an informal version, Bonferroni's principle that helps us avoid treating random occurrences as if they were real. Calculate the expected number of occurrences of the events you are looking for, on the assumption that data is random. If the number is significantly larger than the number of real instances you hope to find, then we must expect almost anything found to be bogus, i.e., a statistical artifact rather than evidence of what you are looking for. This observation is the informal statement of Bonferroni's principle.

Data mining in internet of Things is used to manage the large amount of data which is received from the IoT devices. Data mining involves knowledge discovery and analysis from the massive set of data. The main purpose of data mining is to find useful patterns from large dataset received from Internet of Things (IoT) devices, sensors. Knowledge discovery, pattern analysis and information harvesting are the terms which are used for data mining in Internet of Things. The primary objective of data mining is to build an efficient and descriptive model which is best suitable for the data set. On the basis of definition of data mining and its functions, a typical data mining process includes these four steps, Data Collection, Data Preparation, Data Mining and Presentation.

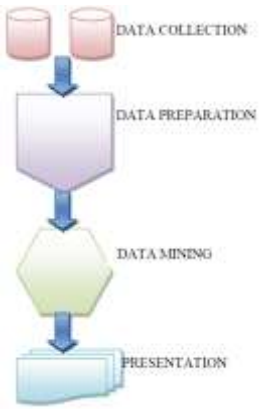

Fig.1. Data Mining System [4]

A. Data Mining with BigData

In traditional data mining systems, the mining process need intensive computing units for analyzing and comparing the data stored in systems. So, this computing platform works with at least two resources, the data and the processors of computing systems.

The data mining of small scale data needs only one desktop computer and it is more enough to process the data. Whereas, the data mining algorithms designed to process the medium scale data and may be the data is distributed to 'n'

number of systems will not fit into single main memory. So, the parallel computing (or) mining came into extinct. The typical data mining algorithms are redesigned in order to sample and collect data from different sources of system and it performs the parallel mining process. For example, parallel K-means algorithms, parallel classifier algorithm and parallel association rule mining algorithm are used to process the distributed data.

When the data scale is far beyond as well as large volume then, the single desktop computer cannot handle, so the data mining framework will use clusters of computing nodes and it is deployed by using some parallel programming tools like MapReduce. MapReduce is a programming framework for large number of data and it uses clusters of nodes to carry out the parallel processing. The large data i.e. some petabytes (or) terabytes of data are split into many small tasks (or) jobs, and each of them is placed on multiple computing nodes that is clusters and the mining of large datasets is performed.

Big Data systems are already designed and it is available in some industry which combines both hardware and software components. Many business intelligence companies like IBM, Microsoft, Tera data, and amazon and so on, have released their own products to service and help customer's data to find out the insights and hidden relationships of stored data.

To build intelligent learning databases system to handle large-scale data processing of Big Data, the first and most key step is to scale up the system exceptionally large volume of data and provide treatments for the characteristics featured by the HACE theorem [32].

The conceptual view of big data processing framework can be categorized into three sections by considering the inside out of data; they are (1) Data accessing and computing (2) Data privacy and domain knowledge (3) Big data mining algorithms [3].

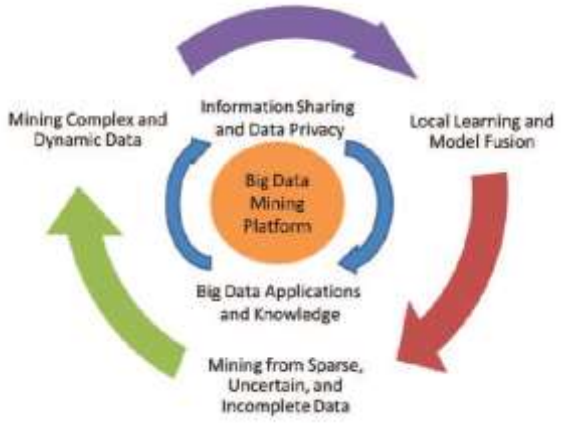

Fig.2. Big Data Processing Framework [3]

B. Big Data Methodologies:

The large volume of data has created more serious problems to data mining algorithms, techniques, statistics and machine learning, and so on due to the complexity of computing large datasets. The research and industrial areas number increases to find out new methodologies and frameworks for big data processing.

The methods that are introduced for big data are MapReduce paradigm, Apache Hadoop and Apache Spark that improves the performance of hadoop [4].

1) MapReduce:

The MapReduce framework and its algorithms are developed to process large-scale Data applications in 'n' number of clusters. MapReduce uses two kernel elements – mappers and reducers in the programming model. Map function will generate a set of temporary key/value pairs and the reduce function is used to merge/combine all the intermediate values of the key. The main idea of MapReduce algorithms is every job node of map and reduce is independent of other parallel job nodes that use different data and key to perform its operations [5] [6].

The main functions of MapReduce are Map (), Combining/Shuffling () and Reduce ().

a) The Input Data: Once the program is received by a Hadoop cluster node, it breaks the program into pieces called tasks and the job is executed. In the master node, once the job is received, it is divided and made to run parallel with other slave nodes. JobTracker node is used to communicate with Namenode to find out its corresponding datanode and will send its tasks.
b) Map function: Map() function is applied to local data of each map or worker node and produces the output data in the temporary storage with key/value pairs such as (k1,v1),(k2,v2)……. Master node is used to arrange and combines all output key values. So, only one data is processed at a time.
c) Shuffle function: The output of map function is sent to the reduce step and assigning new key value to the working node with all associated map node data. Therefore, the data with same key are moved to one worker node.
d) Reduce function: This function is used to process each node data in parallel and to perform the specified reduce job and the function runs only once for each K2 value of map step.
e) Final output: The MapReduce paradigm combines all reduce function outputs and using the key value it is sorted and the output is produced to the user.

This MapReduce application is very useful, if the algorithm is divided into mappers and reducers to work with large data sets. Some parallel algorithms can work on small dataset, but it can't be used for large datasets using this MapReduce methodology [7] [8].

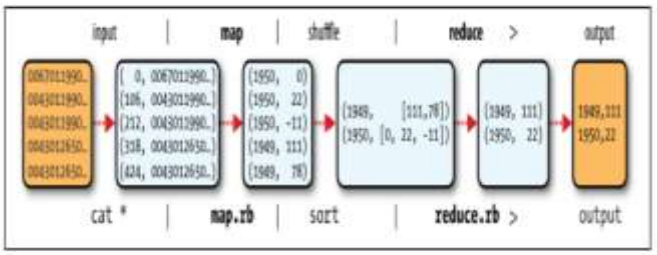

Fig.3. MapReduce Input and Output Format [37]

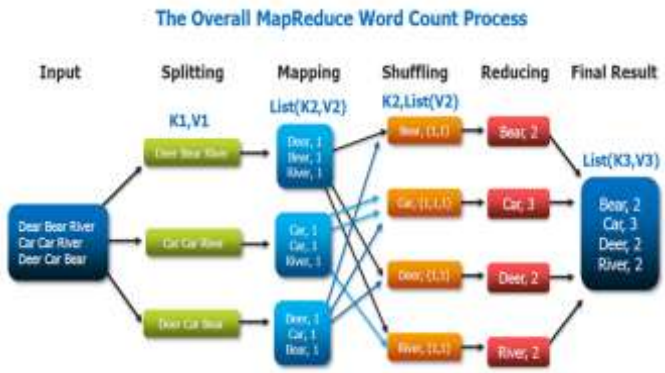

Fig.4. Flow of MapReduce Word Count Process [37]

2) Apache Hadoop:

Hadoop is developed as a top-level apache project by apache software foundation and it is written in java. Hadoop is designed specifically to handle very large-scale data operations and its computing environment is built on top of distributed clustered file system. MapReduce is used to manipulate data which is stored on cluster of servers and the work is broken down into mapper and reducer jobs to achieve massive parallelism [9].

For a wider set of use cases and data sets, hadoop has made it practical and applied for more applications. Unlike traditional transactional systems, hadoop was developed and designed to scan through large data sets and produce its results using highly scalable, distributed batch processing system. Hadoop works as a function to data model and not as a data to function model, since it handles so much of data for analysis.

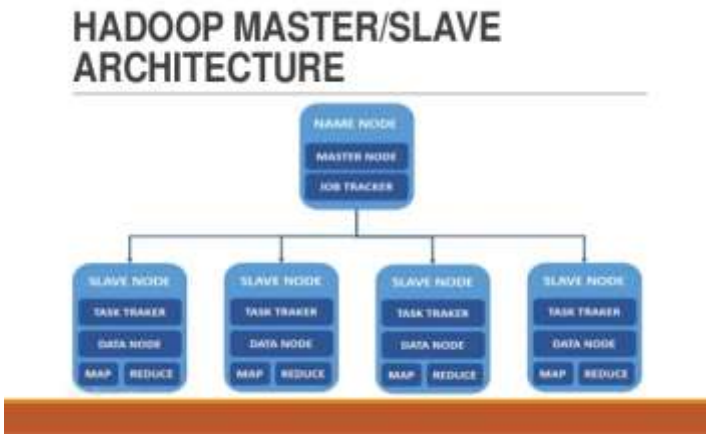

Fig.5. Master/Slave architecture of Hadoop [37]

Hadoop has two fundamental parts, a hadoop distributed file system (HDFS) and a MapReduce programming framework [10].

a) HDFS:

HDFS file system is mainly used to, scale up the hadoop cluster to hundreds and thousands of nodes. The large data in cluster node is split into number of small pieces named as blocks and it is distributed over other nodes of cluster. The mapper and reducer functions will be executed on the smaller blocks of subsets of data to improve the scalability of big data processing.

Hadoop usually uses available servers in a very large cluster and that server has inexpensive internal disk drives and MapReduce is used to process the data stored on these servers. The storage area network (SAN) is used for hadoop environment; it may leads to data locality problem due to extra network communication overhead such as bottlenecks for larger clusters. To overcome the redundancy problem of data, hadoop allows the cluster to break data into smaller chunks and execute those jobson all servers explicitly over the cluster.

The given input data file is divided into blocks and each block default size 64MB, whereas in on-disk file size 512bytes for traditional file system and for relational databases the block size is 4KB to 32KB [11] [12].

b) Name Node:

Name Node is a special server to manage the hadoop's data placement logic. It keeps track all data files in DFS and produces information such as location of block storages and it is stored memory. For any storage manipulation (or) read requests, it also provides quick response times. Name node server should be more robust than the other servers in the hadoop cluster, to avoid single point of failure problem. Latest version of hadoop includes Backup node to store the backup data of name node server.

c) Job Tracker:

Job tracker performs the job in a specific node of a hadoop cluster once the application submits the job. It always communicates with the Namenode, to provide the data required for the existing job over the cluster. It also splits the job into map and reduces tasks for each node across the cluster. Jobtracker will schedule the task on the nodes where data exists in the cluster.

d) TaskTracker:

It is used to monitor a set of tasks that are continually running in a hadoop cluster and provide status of its each task. If the status of the task is failed, then task tracker will report failure status to job tracker to reschedule that task to some other node across the cluster.

e) MapReduce:

MapRedue paradigm serves as the heart of the hadoop cluster. The multiple map and reduce tasks is used to increase the parallelism and improve the overall performance of the executing job. The programs of MapReduce are written in java and its jar file is distributed over various hadoop cluster nodes using jobtracker to execute the map and reduce functions.

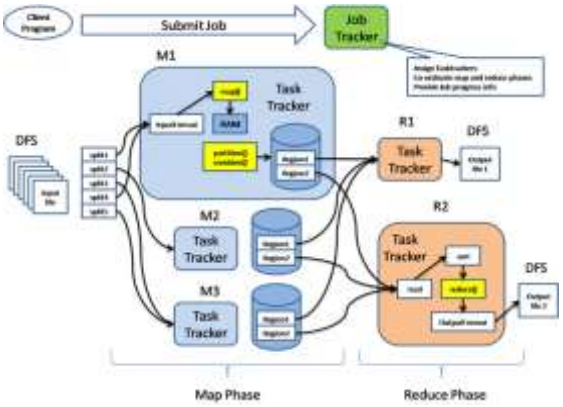

Fig.6. Architecture of Hadoop [37]

3) Apache SPARK:

Apache Spark is a framework that is developed by Apache to process large-scale data and is built as a part of Hadoop architecture. Spark serves as the first distributed framework that allows the use of general purpose programming language to process and compute big data on cluster nodes. It also allows using many iterative machine learning algorithms and some mining algorithms that can reuse the data over many parallel operations and it also uses interactive data analysis tools. It retains the fault tolerance and scalability of MapReduce. Three main components introduced in Spark are resilient distributed datasets (RDD), parallel operations for data, shared variables and spark context [16] [19].

a) SparkContext:

SparkContext is defined as a driver program for the main program and it is mainly used for memory management, scheduling, if there is any fault recovery of the parallel tasks and interacting with storage systems. It can be built using several cluster managers such as yarn, Mesos, or it usesits own standalone cluster manager. First, the cluster manager establishes a connection and Spark will attempt to find the executors of all nodes in the cluster. Once, the executors are found the application code from spark is retrieved by the entire executor and the code is run in parallel using all nodes of cluster.

b) Resilient Distributed Datasets:

Resilient distributed datasets (RDD) is termed as the datasets abstraction introduced in Spark. An RDD is a large collection of objects that are partitioned and distributed over Hadoop cluster nodes by Spark. Since, RDD is read-only datasets, the tasks that are run will not change the elements of RDD. The abstraction of an RDD has enough information to compute the elements of RDD and so all the elements of RDD need not be stored in physical storage. The languages such as Scala, Java, or Python can be used to represent the RDD object of Spark. An RDD object can be developed in three ways:

- By storing in Hadoop Distributed File System (HDFS), to read the file stored in a shared file system.
- By using SparkContext.Parallelize() method in the driver program, the object of collection can be parallelized and the spark is used to divide the object and distribute the partitions over the clusters nodes.
- An existing RDD is transformed with a user defined function by using flatMap or mapcomputing transformation operations. Using flatMap or map function a dataset of one type can be transformed into a new dataset of some other type.

The RDD storage level can be shown using RDD persistence status object and many storage level of RDD are available, such as DISK_ONLY, MEMORY_AND_DISK, MEMORY_ONLY_SER,MEMORY_ONLY,and MEMORY_ONLY_2. Spark RDD will be recomputed by default every time if any action runs on that RDD and it can be reused for the next performing action. In order to improve the performance of algorithm, the method persists () can be used on the RDD to change the persistence level [19].

c) Parallel Operations:

Spark performs many parallel operations on RDD such as Transformations and Actions. Transformations are computed to create new RDD by passing each and every element of an RDD to a user defined function by performing an operation using an existing RDD. The map, flatMap, filter etc. are the several transformation functions provided by Spark. Actions are performed as an operation to compute a desired result using an RDD and return the result to the main driver program. One example is first () is an action operation provided by Spark that mainly return the first element of RDD. The two operations Action and transform on RDD is computed in parallel by running a number of parallel tasks on cluster nodes. The number of RDD partitions should be equal to the total number of tasks or jobs performed. Fig.2 shows the architecture of Spark [19].

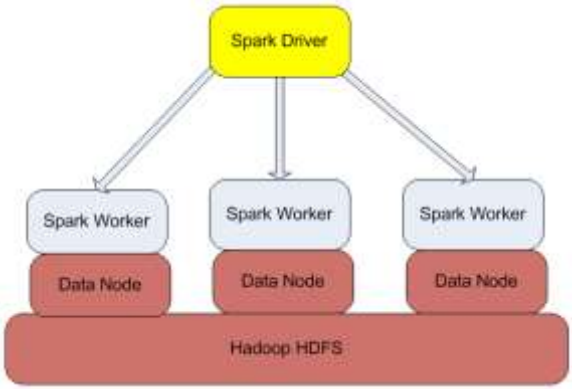

Fig.7. High-Level Architecture of Spark [37]

C. DATA MINING FOR IoT

The relationships among the big data, data mining and KDD for IoT are discussed in the following section. A basic working model to identify the applicable data mining technologies is also given. A basic introduction to the applied data mining techniques for IoT is also discussed, with a data mining unified framework.

1) Idea of using Data Mining for IoT:

The creation of data is much easier than to analyzing the data. The major explosion in the growth of data will make a serious problem for IoT. Till now, varies studies has been introduced and tried to solve the problem of processing IoT big data. There are not enough efficient analysis tools, and so all the systems, will definitely be merged by this large amount of data [1]. From the view of hardware, if KDD is applied to IoT, cloud computing and some other distributed technologies will become the solutions for big data. From the view of software, most of the available mining techniques are designed and developed to run on one system. In the big data era, the most KDD systems available today and traditional data mining techniques cannot be used directly to process this huge amount of IoT data [1].

In general, the preprocessing methods of KDD or the data mining techniques should be redesigned to process a large amount of data which produced by IoT devices. On the other case, the existing data mining algorithms can be used to small scale IoT system and it could produce a very small amount of data.

A data mining module with high performance for IoT is developed using the following three vital concerns in the KDD technology for the problem that has to be solved—the objective, characteristics of data, and mining algorithm [1].

• Objective (*O*): The problem wants to define precisely if the same has to be solved, where the assumptions, limitations, and measurements of the problem needs to be specified first. With that information, the objective of the problem can be made very clearly.

• Data (*D*): The size, distribution, and representation are the characteristics of data and those are the most important aspect of data mining. Different types of data need to be processed differently. The data may come from different applications and that may be related to each other, that data could be analyzed differently if the data has different meanings.

• Mining algorithm (*A*): The data mining algorithm can be easily identified, if the above mentioned objective and data are clearly specified. A new mining algorithm is to be developed and that cannot be easily acceptable by using the above three concerns. For example, based on the data characteristics, if the volume of data goes beyond the competence of a system and if there is no reasonable solution to moderate the complexity of the data, then an innovative data mining technique should be absolutely established [1].

The other concern is associated to the property and objective of the problem. To improve the performance of a system, a novel mining algorithm is required. One illustration related to this is the clustering technique for a wireless sensor network, which needs load of computation to taken into account, but several traditional clustering algorithms purely discount this issue. The objective of the problem is decided, the features of the input data are understood, and the specific goals of mining and the algorithms related to that are selected to process the data [2].

To develop a great performed data mining structure of KDD for an IoT system the following three points need to be considered to select the suitable mining technology, and they are:

• First essential thing is to understand the definition of the problem, their limitations and required information about data and so on.

• Secondly, the major concept would be to understand what kind of data is to be required like the data representation, data size, processing and computation of different data etc.

• Third thing to consider on the basis of the abovementioned considerations, a suitable data mining algorithm is to be chosen to bring out sensible and required information from the provided raw data [2].

2) Data Mining for IoT Applications

There is a rapid growth in the IoT devices and sensors connected over the internet; we can find many applications in this field. Some of the successful applications related to data mining are listed below.

a) Smart City

In order to make the system better and smarter, the various IoT systems in a smart city are given below relating it to the appropriate data mining functionality.

- Traffic Control:

IoT devices used in traffic control are GPS, smart phones, vehicle sensors deployed across the city can provide data points such as travel time, frequency of heavy and light vehicles, accident prone zones and construction areas. The data points collected from sensors will provide the insights to the reason behind congestion in the targeted area. In this scenario, the classification algorithm can be used to solve traffic congestion problem. Depending upon the high, medium, low probability of occurrence of traffic jam in a particular area, the targeted areas can be classified. After developing the classification model, the model can be used to predict the future time of the day where the traffic congestion will be at peak. Based on this prediction, the vehicle can choose the alternative route to arrive at the destination. The traffic will be distributed and the congestion problem is avoided [2].

- Residential E-meters:

Smart meters are nowadays being replaced rapidly with smart meters and the smart meters will provide real time data about the energy consumption through email or on smart phones in a digital format. Time series analysis data mining technique can be used for this time series data in which data is automatically gathered at different intervals for the whole day and can be used to predict the energy consumption and if anomaly is detected in energy consumption it provides notifications by many ways. Synthetic data can also be generated using this smart meters from available real data, which can be used for forecasting data [2].

- Pipeline Leak Detection:

Maintaining water pipe leaks for municipal corporations is a very tedious job. Especially while using the old pipes, sensors can be used to pass the sound of water passing through the pipe can be analyzed using data mining techniques. The outlier detection algorithm is used to identify leaks in the pipes. By implementing this technique in pipeline leak, taxing job of detecting water leaks can be simplified and in addition to this, cost of maintenance can be reduced to the half as compared to the conventional detection method [2].

b) Home Automation

In home automation system, many IoT devices are used and more data generated by different sensors can be mined to generate meaningful information and patterns. The discovered patterns can be mainly used to predict the future events and will provide more effective automated interaction with the user. The classification and time series analysis data mining models can be used for home automation system. Classification is used by classifying the interactive devices that are connected closely together and based upon their usage. Time series analysis data mining algorithm is used for the data that are generated by these devices with their corresponding time stamps and using this future event can be predicted for a particular time using linear regression [2].

c) Health Care

The growth of the health care industry is evidently seen due to the usage and advancements of IoT systems in this medical field. The IoT systems used in the medical field offer innumerable services for users to check on their health such as medication adherence systems, calorie burnt, blood pressure, blood glucose, heart rate, weight measuring devices and pulse

meters and store the data on some cloud based platforms maintained by required hospitals. In order to integrate these heterogeneous data and give accurate information about the patient an intelligent a system should be developed. The data can be text mined using the patient doctor specific prescriptions and medical history and from this we can draw important conclusions about the present condition of the patient and chances of survival of the patient.

Clustering data mining technique can be used for the better treatment and care of the patient. Outlier analysis can also be done to identify any unusual patterns which will be easy in detection of any fraud [2].

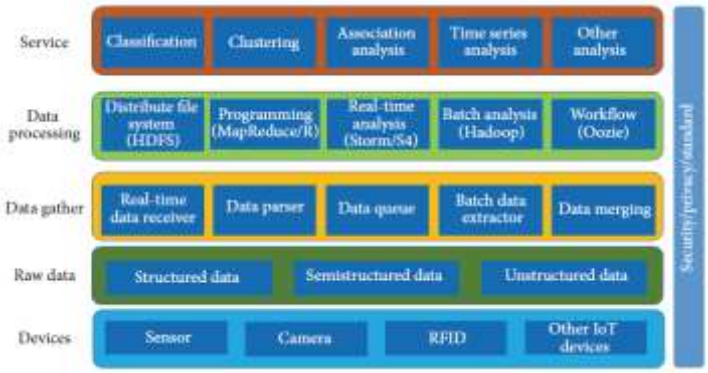

Fig.8. Data Mining IoT Process [1]

II. Data Mining Techniques with Hadoop and MapReduce

A. Frequent Itemset Mining (FIM) -FIDOOP

Association rule mining algorithm is the one of the best data mining algorithm and it is used to find the frequent itemsets and association rules from the given dataset. The frequent itemsets are identified first and then the association rules generated with the confidence larger than or equal to the specified minimum confidence. Apriori algorithm and FP-growth algorithm are the two categories of frequent itemset mining algorithm. Automatic parallelization, load balancing, data distribution, and fault tolerance on large clusters are the major drawbacks of existing parallel mining algorithms for frequent itemsets. A parallel frequent itemsets mining algorithm called FiDoop using the MapReduce programming model is designed as a solution to this problem. FiDoop uses the frequent items ultrametric tree, rather than normal FP tree, in order to avoid building conditional pattern bases and to achieve compressed storage.

FiDoop implements three MapReduce jobs to complete its data mining task, in that the MapReduce job uses mappers to decompose itemsets independently, and reducers will construct small ultrametric trees to combine the operations of mapper jobs. The whole mining tasks of these trees are performed separately. In this paper, FiDoop is implemented using +in-house Hadoop cluster and the cluster is very sensitive to data distribution and dimensions, since the different lengths of itemset will propose different decomposition and construction costs. Workload balance metric is developed to measure the loading balance over the computing nodes of cluster. In order to speed up the mining performance of high-dimensional data analysis, an extension of FiDoop named FiDoop-HD is developed and experimented with real-world celestial spectral data itemsets. The procedure used in their experiment is from the given M datasets, a list of (M-1)-itemsets are taken and it's again decomposed into (M-2)-itemsets and finally it is union into its original (M-2)-itemsets and this decomposition process repeated to accomplish the frequent itemsets. The results of their experiment show that, FiDoop performs best for large low-dimensional datasets and FiDoop-HD performs better for high-dimensional data processing. FiDoop-HD uses decomposing of long itemsets to improve the performance of FiDoop and to measure the load balance of FiDoop a metric tree is introduced.

Future work of this paper includes, applying this metric to investigate the advanced load balance strategies and to implement the data-aware load balancing scheme in the context of FiDoop and integrate data placement mechanism with FiDoop on heterogeneous clusters. To balance the amount of data stored in every heterogeneous node and to improve the data-processing performance, the data placement scheme is used. This parallel mining of frequent itemsets should be investigated on heterogeneous clusters with data placement mechanism. [15]

B. FIM on Hadoop

Valuable business and economic insights can be provided by the frequent item sets and it is very much useful for research purposes. While mining with big data, privacy also plays an important role with the unstructured non-numeric data. So, how to handle and secure the privacy of datasets that contain sensitive information is a big issue. Different privacy preserving mechanisms provided to protect this sensitive information data. In this paper, author introduced a proposed system that uses frequent itemset mining (FIM) technique on MapReduce platform and it performs data processing using non-numeric dataset. FP-growth algorithm is implemented using MapReduce to achieve parallelism of data processing. The proposed system is experimented with four datasets and results with high speed up and better mining efficiency and it can be used for mining the massive small files datasets. A private frequent itemset mining is proposed in this paper using map reduce programming model for privately mining large dataset in order to minimize the required time. FP-growth is mostly used for numeric dataset but the proposed system is implemented to improve the privacy mechanism using non-numeric dataset. [17]

C. K-Nearest Neighbor

The K-nearest neighbor is a frequently used classification method in data mining. This method computes a KNN join for a given set of query points R and a set of reference points S and it discovers the k nearest neighbors in S, for each point in R. The main aim of this algorithm is to find the KNN join of the testing dataset with its training dataset. From the given input unlabeled data, a new class label is discovered for new data points and class membership is computed by performing KNN query on the training set. The KNN computation can also be applied with other methods in many fields like medical, social network analysis, and bio-information and time series analysis. The two main difficulties

of KNN join are data volume and data dimensionality. Indexes can be used to reduce the number of distances that is needed to be calculated and projections can be used to reduce the data dimension. Still the limitations are increased while processing KNN on single machine with large data. KNN can be implemented using a scalable parallel and distributed programming with the Hadoop framework, an open source implementation for large scale data processing.

In this paper, the author surveys existing methods for computing KNN on MapReduce both theoretically and experimentally. Three generic steps of KNN computation on map reduce such as data pre-processing, data portioning and computation are used to compare solutions. Each step of experiment is analyzed for load balancing, accuracy and complexity aspects. The author uses a variety of datasets and analyzes the impact of data volume, data dimension and time and space complexity of k and its accuracy. First section of this paper pointed out all possible solutions to follow three main steps to compute KNN over map reduce are preprocessing of data, portioning and actual computation.

In each step, the pros and cons of other algorithms also discussed. In the second part of this paper, they performed extensive experiments in the same environment and compared the performance, disk usage and accuracy of all these algorithms. For all algorithms two real datasets with high dimensions are used and it was the first published experiment. For each algorithm, a fine analysis and outlining was performed to find out the importance and difficulty of fine tuning some parameters to obtain the best performance.

This paper provides a clear and detailed view of the algorithms for processing KNN on MapReduce and it also gives the limits of each algorithm practically and where they can best perform. Overall, this paper can be used a guideline to select the appropriate method for KNN join operation on MapReduce for a particular dataset. Future work of this paper are First, finding the optimal parameters to reduce the number of replications and Second, to reduce the cost of repartitioning for dynamic queries. MapReduce through its Hadoop implementation is still remains to be an open issue for streaming data and it is well suited for static batch data. [21]

D. Clustering for Big Data

The data is growing as huge data by frequent swiftness in size (volume) and in different formats (variety). This large volume of data can be from various sources namely, media, communication devices, internet, business etc. and while handling it, everyone faces many difficulties and challenges. Data mining is the process that is mostly used in analytical data, typically business or market associated data – also termed as "Big Data". Several data mining techniques are available such as classification, clustering, outlier analysis and association rule mining. In this paper, author discussed several applications and importance of clustering technique. Clustering algorithms provide a powerful meta-learning tool to examine the huge volume of data. They used numerous clustering techniques including traditional and recently developed in this paper and experimented in reference to large data sets. The experimental results with each method's benefits and limitations are being presented in this paper. [14]

E. MapReduce for classification

Classification is one of the most widely used data mining technique and it is the process of developing a model that assigns records in a collection to predefined classes. To develop a classification model, various algorithms like K Nearest Neighbor, C4.5, Support Vector Machine and Naive Bayes have been implemented. The main goal of a classifier model is to accurately predict the target class for a given input data set. The classification process begins with the model creation by taking into consideration an input data as a training data set. MapReduce is a style of programming model that is developed in order to process large datasets on distributed clusters of commodity machines. This MapReduce Programming model can be used to implement a classifier model. In this paper, author first studied the classification algorithms implemented on MapReduce and later they introduced Naïve Bayes classification technique based on MapReduce programming. [22] [23]

F. MapReduce using fuzzy

It became more troublesome to perform efficient analysis using the current traditional techniques, due to the huge increase in the size of data. The several characteristics of big data like volume, velocity, variety, variability, value and complexity are put forward a lot of open issues and research challenges. Today there is a necessity for efficient data mining techniques to process large volume of data but in addition to that there is also a need to meet the computational requirements to process such huge volume of data.

The main goal of this research paper is to implement a map reduce framework using fuzzy and crisp techniques, and a comparative study is provided between the results of the proposed systems and the methods that are reviewed in the literature. In this paper, author proposed four proposed systems that are implemented using the map reduce programming model to process on big data. In first system, the fuzzy k-nearest neighbor method as a fuzzy technique and the support vector machine as non-fuzzy technique are the two techniques used in the mapper. In second system, the mode, the fuzzy soft labels and Gaussian fuzzy membership function are the three techniques used in the reducer. Using the fuzzy KNN in the mapper and the mode in the reducer is the first proposed system and Using the SVM in the mapper and the mode in the reducer is the second proposed system and Using the SVM in the mapper and the soft labels in the reducer is the third proposed system, and Using the SVM in the mapper and fuzzy Gaussian membership function in the reducer is the fourth proposed system. The experimental results using different data sets show that the fuzzy proposed methods give a better performance than the crisp proposed method and the method reviewed in the literature.

In this paper, author introduced a comparative study of classification techniques on big data. Two classification techniques are used with MapReduce paradigm; the fuzzy k-nearest neighbor and the support vector machine. The proposed four algorithms have two parts, the mapper and the reducer.

In mapper part, the data sets are divided into chunks over the computing clusters and produced a set of intermediate records by the map function in the form of a "(key, value)"

pair. The individual nodes mapper execute the computing process and the results are sent to the reduce function. In the reducer part, the individual computations results are received and combined them together to obtain the final result. The reducer algorithm uses three functions; the mode, soft labels, and fuzzy Gaussian. The fuzzy methods show good accuracy of performance than using crisp methods. Future work of this paper will be the further implementation of other classification techniques using parallel algorithms to improve the usage efficiency and accuracy of computing resources and reduce the execution time. [13]

G. A Hybrid Data Mining Algorithm

Data mining has been achieved as an active area of research for the past couple of year decades. In data mining, classification is an important technique that will assign a data instance one of the several predefined categories. To solve the problems of the classification various successful methods have already been suggested and tested. In this paper, author introduced a new hybrid classifier by combining evolutionary and non-evolutionary algorithms. Genetic Programming and Decision Tree algorithms are merged together to improve the accuracy, comprehensibility and timing of the classification technique. The Hybrid algorithm proposed in this paper provides better performance than the decision tree and genetic programming used individually. By using Feature Selection, the accuracy is improved and by shrinking the number of attributes, the comprehensibility is increased. The future work of this paper is to compare the other hybrid algorithms available and then improving the proposed hybrid algorithm for better performance. [18][20]

III. DATA MINING TECHNIQUES WITH SPARK:

A. Nearest Neighbor Classification Algorithm

The main contemporary challenges in machine learning are mining massive and high-speed data streams. The methods should be redesigned with ability to continuously update their structure and handle big number of instances. In this paper, author present a new incremental and distributed classifier based nearest neighbor algorithm named as DS-RNGE for a demanding scenario. The proposed method is implemented in Apache Spark and it uses a distributed metric space ordering to perform faster searches. The author additionally proposed an incremental instance selection method for massive data streams that used to update and remove outdated examples continuously from the case base. DS-RNGE is the first lazy learning method designed for large-scale, high-speed and streaming data problems. The proposed model will organizes the instances by building a distributed metric tree, which consists of a top-level tree that routes the queries to leaf nodes and it also has a set of distributed sub trees to perform the searches in parallel.

The DS-RNGE also has an instance selection technique that allows the insertion of correct examples and removing outdated ones constantly that improves the performance and effectiveness of the learner. The system is able to respond quickly to the continuous stream of data, since all the phases of DS-RNGE perform its computations locally. The experimental analysis results DS-RNGE gives high accuracy by combining significantly reduced processing time and memory consumption. For a resource-efficient mining of massive dynamic data collections this model can be used. DS-RNGE is compared with other algorithms, it give better time results for prediction phase whereas other algorithms are faster in updating the case-base. Future work of this paper is on adding a concentration technique to control the ever-growing size of case-base and by removing redundancy the cost of time derived from will be reduced. They also plan to extend their model with drift detection module to propose time and memory efficient solutions for drifting data streams. The algorithm can be modified in future that will make it suitable for mining massive and imbalanced data streams.[25]

B. Intelligent K-Means Algorithm

The conventional environment cannot be used to compute when the amount of data is big and the growth of data has brought to the big data generation. Many computational environments had been developed to compute this big data, they are Hadoop and Spark. Hadoop uses Distributed File System and MapReduce framework and Spark is a new parallel and distributed framework that can be combined and run on top of Hadoop. In this paper, author designed a intelligent K-means algorithm implemented on Spark that run on Hadoop environment for big data clustering. In this design, batch of data is used instead of using Resilient Distributed Dataset (RDD) and compared the results with the implementation that using original RDD of data. The Experiment result shows that implementation using batch of data is faster than using original RDD. So, this design can speed up the computational time in big data platform.[24]

C. A Parallel Random Forest Algorithm

The emergence of big data and the research issue of how to extract valuable knowledge from big dataset efficiently and accurately have attracted more in both academia and industry. In this paper, Parallel Random Forest (PRF) algorithm is used on the Apache Spark platform. A hybrid approach by combining data parallel and task parallel optimization is used to optimize the PRF algorithm. From the view of data-parallel optimization, to reduce the data communication cost effectively a vertical data-portioning method is performed and to reuse the training dataset and to diminish the volume of data, a data-multiplexing method is performed.

From the view of task-parallel optimization, in the training process of RF a dual parallel approach is carried out and according to the parallel training process of PRF and the dependence of RDD objects, a task Directed Acyclic graph is created. For the tasks in the DAG, different task schedulers are invoked. The author performed a dimension reduction approach in the training process in order to improve the accuracy of algorithm for a large, high-dimensional, and noisy data. A weighted voting approach is used in this algorithm in the prediction process prior to parallelization.

Extensive experiments are conducted and results show that the PRF algorithm has superiority and notable advantages over other algorithms implemented using Spark MLlib in terms of accuracy, performance and scalability.

Future work of this paper is implementing incremental random forest algorithm using stream data in cloud environment and the improvement of task scheduling mechanism and data allocation on a distributed and parallel environment for the algorithm.[26]

D. Best of Breed Solution for Clustering

The unsupervised algorithm clustering is the process of assigning entities into groups based on the similarities among the entities. The crucial step of mining satellite images is the Image clustering. Since satellite images are generated at a higher rate there should be better solutions in terms of accuracy and performance. In this paper, author proposed the solution using big data platform Apache Spark that performs the image clustering using different methods namely scalable K-means++, Bisecting K-means and Gaussian Mixture. Best of Breed approach of validating the number of clusters using Simple Silhouette Index algorithm is used to find out number of clusters, since the number of clusters will not be known in advance in any of the above mentioned methods and thus to provide the best clustering method.

In this experiment, they suggest a solution to choose the best clustering algorithm to a particular satellite image and also the number of clusters needed for image processing using Simple Silhouette Index. The proposed Best of Breed solution is scalable for images with 200MB in size. The future work is to add more clustering algorithms that are not present in Apache Spark and evaluate it on the basis of their scalability and performance.[27]

E. A Fast Heuristic Attribute Reduction Algorithm

The energy data with energy consumption statistics and other related data in green data centers are mostly used and grow dramatically. But many attributes of energy data are redundant and unnecessary. Many existing attribute reduction algorithms are used to reduce the attributes of energy data but the time consumption is high for computation. In this paper, author addresses these issues by extending the methodology of rough data sets to construct the energy data center. An attribute reduction algorithm with good advantage of in-memory computing using Spark is proposed in this paper.

This algorithm uses a heuristic formula to measure the significance of attribute to reduce search space. This efficient algorithm also used to simplify the energy consumption decision table that further used to improve the computation efficiency. Experiments are conducted using this proposed algorithm and the results show that the speed of this algorithm has 0.28X performance improvement compared to other traditional attribute reduction algorithm using Spark. [28]

F. Distributed Data AugmentedSupport Vector Machine

Support vector machine (SVM) remains a widely used method for classification in data mining and machine learning tasks and the traditional SVM can be applied from small to medium datasets. The need to scale up the SVM with data size and to develop new implementation method to improve the performance of SVM is the most recent research attention. Many distributed SVMs are developed recently but the distributed SVM implementation with data augmentation has not been developed. In this Paper, author introduced a new distributed data augmentation implementation of SVM using Spark. Apache Spark is the most popular and widely used parallel platform for distributed computing in research as well as industry.

The proposed method is termed as sparkling vector machine (SkVM) that supports both classification and regression methods by scanning the data exactly only once. In addition to this, the author developed a framework to solve label-drift classification problem, to handle the data with new classes that are arrived using online classification setting where new data points can have labels that are not previously used. The scalability of the proposed method is demonstrated using large-scale datasets which has more than one hundred million data points. The results of experiment show that the predictive performance of proposed method is better than other methods of that already implemented on Spark with respect to execution time and order of magnitude. The sparkling SVM is mainly used to handle the multiclass classification, regression, label drift classification and the distributed algorithm.[29]

G. Performance Modeling for Spark Using SVM

In many organizations and enterprises, Spark is widely used when compared to Hadoop. For some applications Spark performs much faster than Hadoop, the configuration parameters of spark have a great impact in its performance due to its large parameters, interactions between applications and characteristics of applications. The research to predict the performance of Spark is very less based on its configuration sets. In this paper, the author used the machine learning technique Support Vector Machine (SVM) in order to build the performance models for Spark. Running the Spark application using randomly modified and combined parameter values and the input of configuration sets are collected. By this way, they determined the range of each property and gained a deeper understanding about how the properties are work in Spark. The author also uses Artificial Neural Network method to model the performance of Spark using three workloads from HiBench.

Two major phases are used to build the performance model of Spark: (1) Data collecting and (2) Modeling. In the first phase, three workloads from HiBench suite with different characteristics are selected. The configuration parameters and execution time are collected after each time of running workload and stored as a vector to use later as a sample data. In second step, the performance models of Spark are built using the input matrix of configuration parameter values and execution times from the stored vector. The models built using SVM are compared with models built by ANN method. They found that ANN error rate is on average of 1.98 times more when compared with SVM. The results show that predicting the performance of Spark using SVM is better than ANN. The machine learning algorithms are more effective to build the performance model of Spark and like this, many other methods like least square and Bayesian can also be used to predict the performance and optimal configuration parameters of Spark.[30] [31]

H. R-Apriori: An Efficient Apriori based Algorithm

Association rule mining is an effective and very popular method to find possible associations between the items and to extract meaningful information from large transaction based datasets. Frequent patterns could be generated in order to create the associations. The "Apriori" algorithm is the most preferred choice of frequent pattern generation algorithm along with its set of improved variants and it provides ease of implementation and its natural tendency of parallelization. Many Apriori methods are available for single machine but the data available today is far beyond the capacity of a single machine. To meet the demands of the ever growing data, there is need of scalability across multiple machines. MapReduce became more popular programming and fault-tolerant framework for distributed applications. Many systems with heavy disk I/O implements MapReduce framework using efficient iterative data mining algorithms like Apriori. To overcome the disk I/O bottle necks in MapReduce, a new in-memory distributed dataflow platform called Spark is proposed. Spark represents an ideal platform for distributed mining algorithms.

In Apriori algorithm implementation, the candidate sets generation having all possible pairs for single frequent items and comparing each pair with each transaction record is the most computationally expensive job. Here, the author proposed a new approach that eliminates the candidate generation step which in turn dramatically reduces this computational complexity and avoiding costly comparisons. The author conducted various in-depth experiments to gain insight into the effectiveness, efficiency and scalability of their research. Since, the conventional Apriori takes more time and space for the iteration of frequent item generation, a new version of Apriori is implemented in order to mine the frequent patterns of large datasets with various attributes. For example, the 106 singleton frequent items will generate nearly 110 candidate set for second iteration and it is very huge and infeasible. The R-Apriori algorithm results a better and improved performance when the size of the data and the number of items increases. R-Apriori is implemented on spark, which is high parallel computational platform as compared to others. The comparison of R-Apriori with classical Apriori implementation on Spark platform using different standard datasets results the R-Apriori outperforms well with existing Apriori algorithms.[32] [33]

I. PAMAE: Parallel k-Medoids Clustering Algorithm

The best known data mining clustering algorithm is k-medoids algorithm, but it is not used widely due to its high computational complexity. Compared to this, k-means algorithm is used very much for big data analytics.Many research studies and methods are proposed to solve the computational problem of k-medoids algorithm, but the efficiency improved based on the expense of accuracy. In this paper, author proposed a new parallel k-medoids algorithm called as PAMAE with high accuracy and high efficiency. They identify main two factors "global search" and "entire data" which are most essential to achieve high accuracy and very less time consuming if it is used simultaneously.

The key idea of this paper is to apply the two factors individually in two phases which are named as parallel seeding and parallel refinement. In the first phase, global search is applied on sampled data and in the second phase local search is done on entire data. They analyzed theoretically that the execution of two phases serially will result to accurate solution by doing global search over entire data. The PAMAE algorithm is implemented both on Hadoop and Spark, in order to validate the results and the experiments are conducted using many real world data sets on 12 machines. Compared to other parallel algorithms, the results of PAMAE significantly outperform and it produces more clustering quality. Compared to other algorithms such as GREEDI and CLARA-MR, the PAMAE algorithm gives more accuracy and efficiency with decreased clustering error. The source code and data of this paper are available at https://github.com/jaegil/k-Medoid.[34]

J. Distributed Frequent Itemset Mining Algorithm

In the process of Association rule mining, frequent itemset mining is an essential step. In the big data era, many conventional methods for mining frequent itemsets encounter significant research challenges when computing power and memory space are limited. In this paper, author proposed an efficient distributed frequent itemset mining algorithm (DFIMA) that can reduce the amount of candidate itemsets by using a matrix-based pruning approach. The proposed algorithm is implemented using Spark to improve the efficiency of iterative computation. The author conducted a numeric experiment using standard benchmark datasets and compared the proposed algorithm DFIMA with the existing algorithm parallel FP-growth running on spark.

The results show that DFIMA method has better performance with its efficiency and scalability. Additionally, a case study has been performed to validate the feasibility of DFIMA. The major problems of Apriori algorithm and iterative computation on MapReduce framework are performance bottleneck due to repeated database scanning that makes less use frequent itemset mining for massive data. They noticed in experiments that the proposed method performs especially well in occasions with a relatively high support degree. To improve DFIMA and make it suitable for more mining cases, a further optimization is to be considered.[35]

IV. Conclusion and Future Work

In this paper, we have reviewed studies on applying data mining technologies with the big data. Since, the growth of data is growing dramatically, the open issues and challenges of big data and data mining will gain more attraction to researchers and industries. First part of this paper is about the basic concepts of data mining, big data characteristics and big data methodologies are discussed. In the second part, the proposed algorithms of existing data mining techniques which are implemented using big data methodologies like Hadoop, MapReduce and Spark are presented. We did a literature survey on the past five year's papers and given a guideline to know about the current trends and algorithms that can be considered to process large-scale data. Hadoop is the first proposed big data methodology to process massive data in a distributed clusters using MapReduce paradigm. The data mining techniques such as classification, clustering and association rule mining are implemented using Hadoop to perform the parallel and

distributed large-scale processing. Apache Spark is the newly introduced framework of big data that uses Resilient distributed datasets (RDD) to perform parallel operation on data. Hadoop and Spark are both big data frameworks, but they are not same and they really don't serve same process. They differ in many ways,

- Hadoop uses distributed data infrastructure and it distributes the massive data across multiple nodes within the cluster of commodity servers. On the other hand Spark doesn't do distributed storage; it is data processing tool that operates on those distributed data collections.
- HDFS is used as a storage component and MapReduce is used a processing component in Hadoop. So, Hadoop don't need Spark to get its processing to be done. Spark doesn't have its own file management system. Spark can be implemented using Hadoop and uses HDFS file system. Sometimes, Spark may not also used with hadoop; it can be used with some other file systems.
- In the way of processing data, spark is generally a lot faster than MapReduce. Because, MapReduce operates in steps, it reads data from cluster and performs operation and write results to cluster and again read updated data from cluster etc. In Spark, it uses one step to read data from cluster perform all required operations on it and write results back to cluster. Spark can be 10 times faster than MapReduce for batch data processing and 100 times faster for in-memory data operations.
- MapReduce programming style can be applied to static and batch data. But to do analytics for streaming data, data from sensors and multiple operation applications require Spark platform to perform faster.
- In hadoop, data is written directly to disk after performing every operation, so it is naturally resilient to system faults. Spark uses RDD objects that will provide full recovery to system faults or failures. So, the data objects can be stored either on memory or on disks.

Recently, authors used Spark as implementation framework for big data and various new algorithms were proposed. Due to some disadvantages like single node failure, not suitable for real time data of hadoop over spark, recently many papers introduced the data mining algorithms with spark and achieved efficiency and scalability.

In this paper, we also reviewed the various works of IoT applications using data mining technologies, which consist of clustering, classification, and frequent patterns mining technologies, from the infrastructures perspective view and from the services perspective view. The survey analysis and implementations on the scale of each mining technology related to IoT applications are also included [1]. The development of IoT is still at the beginning stage of Nolan's stages of growth model. The main focus is on development of *effective* mining technologies to extract patterns and on the development of *efficient* preprocessing mechanisms to make the IoT system capable of handling big data and rules to describe the data of IoT [1].

To help the researchers of the paper to do *something* to proceed, the possible future work directions are given below:

- Big data is the future trend there is no doubt in that because most of the devices once they are connected to the internet will upload data to the internet. Classical data mining researches, namely, data fusion, data abstraction, and data summarization are used in IoT to inherit several signals processing problem. When using data mining technologies to analyze the data of IoT, to handle the flood of big data, sampling technologies, compression technologies, incremental learning technologies, and filtering technologies all will become more and more important [1].
- Ontology and semantic web technologies will give some possible solutions to these problems, but they cannot be used to fully solve these problems because these two technologies are not developed that mature enough. To enable a sensor to make decisions by itself, the fuzzy logic and multi-objective methods can be used. As a result, we do believe that these two technologies will be the trend of data mining technologies for the IoT [1].
- Many mobile app's are available today make it possible to use handheld devices to control our household appliances from anywhere. Two applications related to that are app to control the multimedia devices and app to control the energy of a home. That app applications are used to control devices are everywhere, but how to mine the data from these app's will become more interesting and potential research issue. The data of climate and power consumption are collected by a large number of sensors located almost everywhere to process the data [1].
- In the intelligent system and traditional data mining researches, feedback mechanisms are used. But this is not used in the household appliances and some other IoT applications. The interactive learning and feedback procedures can be used, by researchers while using data mining technologies for the IoT [1].
- The swarm intelligence algorithm fits perfectly to this IoT environment based on the computing and data flow characteristics, because particle swarm optimization and ant colony optimization can be put on the devices to make centralized computing and cooperative learning when the data is complex and large [1].

The Future work trends of this area are, from this paper we came to know that many new data mining algorithms has been proposed using big data methodologies for large-scale data processing. But those algorithms were experimented with the specified data sets for that particular application and they have not used any real time data or sensor data or multiple operations data.

That algorithms work well only with that specified datasets. So, finding an efficient mining algorithm that suits for any type of data with large size and volume is still a major research gap in this big data era. The algorithm can be proposed by modifying mathematical operations of the algorithm or by combining two or three algorithms (Hybrid approach) or by using the parallel algorithm on distributed platform like Hadoop and Spark. Thus this paper provides an overall mirror view of data mining techniques that were implemented on big data methodologies and provides a way for the researchers to explore their ideas.

Mining Techniques	Proposed Method	References
Frequent Itemset Mining Algorithm	i. Frequent Itemset Mining (FIM) using Hadoop–FIDOOP ii. FIM using Hadoop iii. Distributed Frequent Itemset Mining using Spark	[15], [17], [35]
Apriori Algorithm	i. R-Apriori: An Efficient Apriori using Spark	[32]
K-Nearest Neighbor Means Algorithm	i. Nearest Neighbor Means using Hadoop ii. Intelligent K-means using Spark iii. Nearest Neighbor Means using Spark	[21], [24], [25]
Support Vector Machine Algorithm	i. Distributed Data Augmented SVM using Spark ii. Performance Modeling for Spark Using SVM	[29], [31]
Parallel Random Forest Algorithm	i. A Parallel Random Forest using Spark	[26]
Clustering Algorithm	i. Clustering for Big Data using Hadoop ii. PAMAE: Parallel k-Medoids Clustering using Spark iii. Best of Breed Solution for Clustering using Spark	[14], [27], [34]
Hybrid Algorithm	i. A Hybrid Data Mining Algorithm using Hadoop (Genetic algorithm + Decision tree) ii. MapReduce for Classification using Hadoop iii. MapReduce with fuzzy and Classification algorithm using Hadoop iv. A Fast Heuristic Attribute Reduction using Spark	[18], [20], [22], [13] , [28]

Table.1. Comparison of Data Mining Techniques

References

[1] C. W. Tsai, C. F. Lai, M. C. Chiang, and L. T. Yang, "Data mining for internet of things: A survey," *IEEE Commun. Surv. Tutorials*, vol. 16, no. 1, pp. 77–97, 2014.

[2] K. Tapedia and A. Wagh, "Data Mining for Various Internets of Things Applications," *Natl. Conf. "NCPCI*, no. March, pp. 127–132, 2016.

[3] X. Wu, X. Zhu, G. Q. Wu, and W. Ding, "Data mining with big data," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 1, pp. 97–107, 2014.

[4] Shen Bin, Liu Yuan, and Wang Xiaoyi, "Research on data mining models for the internet of things," *2010 Int. Conf. Image Anal. Signal Process.*, pp. 127–132, 2010.

[5] M. Marjani *et al.*, "Big IoT Data Analytics: Architecture, Opportunities, and Open Research Challenges," *IEEE Access*, vol. PP, no. 99, p. 1, 2017.

[6] G. Bello-Orgaz, J. J. Jung, and D. Camacho, "Social big data: Recent achievements and new challenges," *Inf. Fusion*, vol. 28, pp. 45–59, 2016.

[7] J. Archenaa and E. A. M. Anita, "A survey of big data analytics in healthcare and government," *Procedia Comput. Sci.*, vol. 50, pp. 408–413, 2015.

[8] J. L. Reyes-Ortiz, L. Oneto, and D. Anguita, "Big data analytics in the cloud: Spark on Hadoop vs MPI/OpenMP on Beowulf," *Procedia Comput. Sci.*, vol. 53, no. 1, pp. 121–130, 2015.

[9] A. Gandomi and M. Haider, "Beyond the hype: Big data concepts, methods, and analytics," *Int. J. Inf. Manage.*, vol. 35, no. 2, pp. 137–144, 2015.

[10] C. Kacfah Emani, N. Cullot, and C. Nicolle, "Understandable Big Data: A survey," *Comput. Sci. Rev.*, vol. 17, pp. 70–81, 2015.

[11] Ishwarappa and J. Anuradha, "A Brief Introduction on Big Data 5Vs Characteristics and Hadoop Technology," *Procedia Comput. Sci.*, vol. 48, no. C, pp. 319–324, 2015.

[12] X. Ke, H. Jin, X. Xie, and J. Cao, "A distributed SVM method based on the iterative MapReduce," *Proc. 2015 IEEE 9th Int. Conf. Semant. Comput. IEEE ICSC 2015*, no. 4, pp. 116–119, 2015.

[13] M. El Bakry, S. Safwat, and O. Hegazy, "A MapReduce fuzzy techniques of big data classification," *Proc. 2016 SAI Comput. Conf. SAI 2016*, pp. 118–128, 2016.

[14] M. Dave and H. Gianey, "Different clustering algorithms for Big Data analytics: A review," *Proc. 5th Int. Conf. Syst. Model. Adv. Res. Trends, SMART 2016*, pp. 328–333, 2017.

[15] I. U. MapReduce, Y. Xun, J. Zhang, X. Qin, and S. Member, "FiDoop : Parallel Mining of Frequent," vol. 46, no. 3, pp. 313–325, 2016.

[16] N. Singhal and M. Ashraf, "Performance enhancement of classification scheme in data mining using hybrid algorithm," *Int. Conf. Comput. Commun. Autom.*, pp. 138–141, 2015.

[17] T. V Kenekar, "An Efficient Private FIM On Hadoop MapReduce," pp. 72–76, 2016.

[18] S. Sahay, S. Khetarpal, and T. Pradhan, "Hybrid Data Mining Algorithm in Cloud Computing using MapReduce Framework," no. 978, pp. 507–511, 2016.

[19] Z. Sun and G. Fox, "Study on Parallel SVM Based on MapReduce," *Int. Conf. Parallel Distrib. Process. Tech. Appl.*, pp. 16–19, 2012.

[20] G. Xu, C. Shen, M. Liu, F. Zhang, and W. Shen, "A user behavior prediction model based on parallel neural network and k-nearest neighbor algorithms," *Cluster Comput.*, vol. 20, no. 2, pp. 1703–1715, 2017.

[21] G. Song, J. Rochas, L. E. Beze, F. Huet, and F. Magoulès, "K Nearest Neighbour Joins for Big Data on MapReduce: A Theoretical and Experimental Analysis," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 9, pp. 2376–2392, 2016.

[22] A. Haldankar, "A MapReduce based approach for classification," 2016.

[23] O. Dridi, "Machine Learning With Spark CETIC Presentation," 2015.

[24] I. Kusuma, M. A. Ma'Sum, N. Habibie, W. Jatmiko, and H. Suhartanto,

"Design of intelligent k-means based on spark for big data clustering," *2016 Int. Work. Big Data Inf. Secur. IWBIS 2016*, pp. 89–95, 2017.

[25] S. Ramírez-gallego *et al.*, "Nearest Neighbor Classification for High-Speed Big Data Streams Using Spark," pp. 1–13, 2017.

[26] J. Chen *et al.*, "A parallel random forest algorithm for big data in a spark cloud computing environment," *IEEE Trans. Parallel Distrib. Syst.*, vol. PP, no. 99, pp. 919–933, 2016.

[27] T. Sharma, "Images Using Bigdata Platform Spark," no. Icicct, pp. 1–5, 2017.

[28] M. Chen, J. Yuan, L. Li, D. Liu, and T. Li, "A Fast Heuristic Attribute Reduction Algorithm Using Spark," *2017 IEEE 37th Int. Conf. Distrib. Comput. Syst.*, pp. 2393–2398, 2017.

[29] T. D. Nguyen, V. Nguyen, T. Le, and D. Phung, "Distributed Data Augmented Support Vector Machine on Spark," no. 498, pp. 498–503, 2016.

[30] S. Hong, W. Choi, and W. Jeong, "GPU in-memory processing using Spark for iterative computation," *Proc. 17th IEEE/ACM Int. Symp. Clust. Cloud Grid Comput.*, pp. 31–41, 2017.

[31] N. Luo, Z. Yu, Z. Bei, C. Xu, C. Jiang, and L. Lin, "Performance Modeling for Spark Using SVM," 2016.

[32] S. Rathee, M. Kaul, and A. Kashyap, "R-Apriori: An Efficient Apriori based Algorithm on Spark," *Acm*, pp. 27–34, 2015.

[33] S. Venkataraman *et al.*, "SparkR: Scaling R Programs with Spark," *SIGMOD Int. Conf. Manag. Data*, p. 4, 2016.

[34] H. Song, J.-G. Lee, and W.-S. Han, "PAMAE: Parallel k-Medoids Clustering with High Accuracy and Efficiency," *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, pp. 1087–1096, 2017.

[35] F. Zhang, M. Liu, F. Gui, W. Shen, A. Shami, and Y. Ma, "A distributed frequent itemset mining algorithm using spark for big data analytics," *Cluster Comput.*, vol. 18, no. 4, pp. 1493–1501, 2015.