International Conference on Systems, Science, Control, Communication, Engineering and Technology 2016 [ICSSCCET 2016]

ISBN	978-81-929866-6-1		**VOL**	02
Website	icssccet.org		**eMail**	icssccet@asdf.res.in
Received	25 – February – 2016		**Accepted**	10 - March – 2016
Article ID	ICSSCCET145		**eAID**	ICSSCCET.2016.145

A Big Success of Big Data

M Thillainayaki[1], **M Hemalatha**[2]
[1]Research Scholar, Bharathiar University, Coimbatore.
[2]Head of Computer Science, Karpagam University, Coimbatore

Abstract- *The age of big data is now coming. But the traditional data analytics may not be able to handle such large quantities of data. The question that arises now is, how to develop a high performance platform to efficiently analyze big data and how to design an appropriate mining algorithm to find the useful things from big data. With an ever-increasing amount of options, the task of selecting machine learning tools for big data can be difficult. The available tools have advantages and drawbacks, and many have overlapping uses. The world's data is growing rapidly, and traditional tools for machine learning are becoming insufficient as we move towards distributed and real-time processing. This paper is intended to aid the researcher or professional who understands machine learning but is inexperienced with big data. In order to evaluate tools, one should have a thorough understanding of what to look for. To that end, this paper provides a list of criteria for making selections along with an analysis of the advantages and drawbacks of each.*

INTRODUCTION

The goal of machine learning is to enable a system to learn from the past or present and use that knowledge to make predictions or decisions regarding unknown future events. In the most general terms, the workflow for a supervised machine learning task consists of three phases: build the model, evaluate and tune the model, and then put the model into production. In response to the problems of analyzing *large-scale data*, quite a few efficient methods, such as sampling, data condensation, density-based approaches, grid-based approaches, divide and conquer, incremental learning, and distributed computing, have been presented. Of course, these methods are constantly used to improve the performance of the operators of data analytics process. The results of these methods illustrate that with the efficient methods at hand, we may be able to analyze the large-scale data in a reasonable time. The dimensional reduction method (e.g., principal components analysis; PCA) is a typical example that is aimed at reducing the input data volume to accelerate the process of data analytics. Another reduction method that reduces the data computations of data clustering is sampling, which can also be used to speed up the computation time of data analytics.

At the heart of machine learning is the data that powers the models, and the new era of Big Data is catapulting machine learning to the forefront of research and industry applications. The meaning of the term "big data" is still the subject of some disagreement, but it generally refers to data that is too big or too complex to process on a single machine. We live in an age where data is growing orders of magnitude faster than ever before. According to International Data Corporation's annual Digital Universe study, the amount of data on our planet is set to reach 44 zettabytes (4.4 × 1022 bytes) by 2020 which would be ten times larger than it was in 2013. While no single entity is working with data at this magnitude, many industries are still generating data too large to be processed efficiently using traditional techniques. Ancestry.com, for example, stores billions of records totalling about 10 petabytes of data. With such a growth rate in data production, the challenge faced by the machine learning community is how to best efficiently process and learn from big data. Popular machine learning toolkits such as R or Weka were not built for these kinds of workloads. Although Weka has distributed implementations of some algorithms available, it is not on the same level as tools that were initially designed and built for terabyte-scale. Hadoop, a popular framework for working with big data, helps to solve this scalability problem by offering distributed storage and processing solutions. While Hadoop is just a framework for processing data, it provides a very extensible platform that allows for many machine learning projects and applications; the focus of this paper is to present those tools.

McKinsey & Company observed how big data created values after in-depth research on the U.S. healthcare, the EU public sector administration, the U.S. retail, the global manufacturing, and the global personal location data. Through research on the five core industries that represent the global economy, the McKinsey report pointed out that big data may give a full play to the economic function, improve the productivity and competitiveness of enterprises and public sectors, and create huge benefits for consumers. In McKinsey summarized the values that big data could create: if big data could be creatively and effectively utilized to improve efficiency and quality, the potential value of the U.S medical industry gained through data may surpass USD 300 billion, thus reducing the expenditure for the U.S. healthcare by over 8 %; retailers that fully utilize big data may improve their profit by more than 60 %; big data may also be utilized to improve the efficiency of government operations, such that the developed economies in Europe could save over EUR 100 billion (which excludes the effect of reduced frauds, errors, and tax difference).

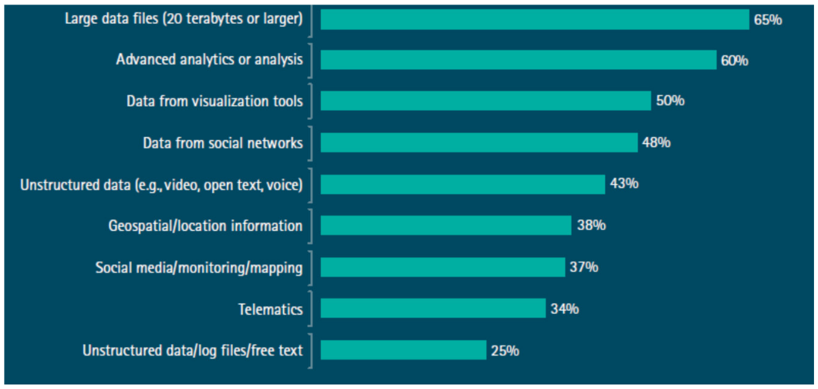
II. Big Data Relationships

The sharply increasing data deluge in the big data era brings about huge challenges on data acquisition, storage, management and analysis. Traditional data management and analysis systems are based on the relational database management system (RDBMS). However, such RDBMSs only apply to structured data, other than semi-structured or unstructured data. In addition, RDBMSs are increasingly utilizing more and more expensive hardware. It is apparently that the traditional RDBMSs could not handle the huge volume and heterogeneity of big data. The research community has proposed some solutions from different perspectives. For example, cloud computing is utilized to meet the requirements on infrastructure for big data, e.g., cost efficiency, elasticity, and smooth upgrading/downgrading. For solutions of permanent storage and management of large-scale disordered datasets, distributed file systems and NoSQL databases are good choices. Such programming frameworks have achieved great success in processing clustered tasks, especially for webpage ranking. Various big data applications can be developed based on these innovative technologies or platforms. Moreover, it is non-trivial to deploy the big data analysis systems. Some literature discusses obstacles in the development of big data applications. The key challenges are listed as follows:

- Data representation:
- Redundancy reduction and data compression
- Data life cycle management
- Analytical mechanism
- Data confidentiality
- Energy management
- Expendability and scalability
- Cooperation

The features of Big Data are:

- It is huge in size.
- The data keep on changing time to time.
- Its data sources are from different phases.
- It is free from the influence, guidance, or control of anyone.
- It is too much complex in nature, thus hard to handle.



It's huge in nature because, there is the collection of data from various sources together. If we consider the example of Facebook, lots of numbers of people are uploading their data in various types such as text, images or videos. The people also keep their data changing continuously. This tremendous and instantaneously, time to time changing stock of the data is stored in a warehouse. This large storage of data requires large area for actual implementation. As the size is too large, no one is capable to control it oneself. The Big Data needs to be controlled by dividing it in groups. Due to largeness in size, decentralized control and different data sources with different types the Big Data becomes much complex and harder to handle. We cannot manage them with the local tools those we use for managing the regular data in real time. For major Big Data-related applications, such as Google, Flicker, Facebook, a large number of server farms are deployed all over the world to ensure nonstop services and quick responses for local markets.

III Data Analytics in Big Data

However, many challenges on big data arose. With the development of Internet services, indexes and queried contents were rapidly growing. Therefore, search engine companies had to face the challenges of handling such big data. Google created GFS and Map Reduce programming models to cope with the challenges brought about by data management and analysis at the Internet scale. In addition, contents generated by users, sensors, and other ubiquitous data sources also filled the overwhelming data flows, which required a fundamental change on the computing architecture and large-scale data processing mechanism.

Hadoop Ecosystem

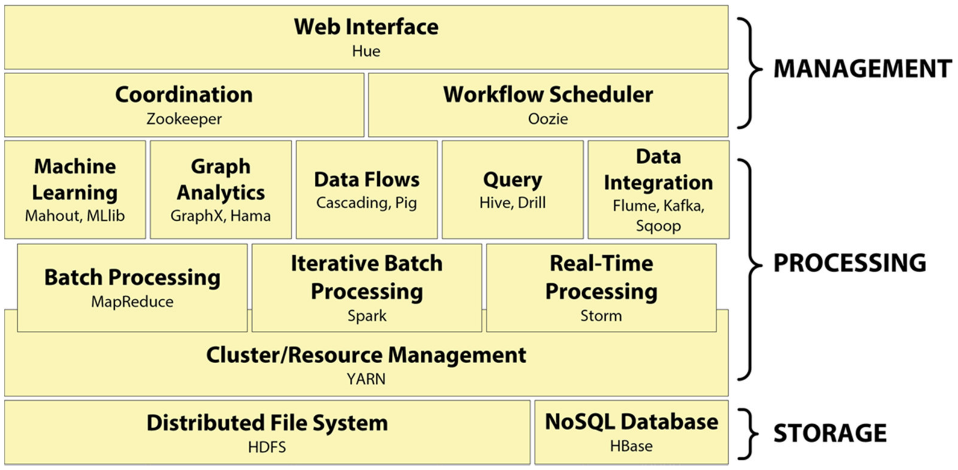
Many people consider the terms *Hadoop* and *MapReduce* to be interchangeable, but this is not entirely accurate. Hadoop was initially introduced in 2007 as an open source implementation of the MapReduce processing engine linked with a distributed file system, but it has since evolved into a vast web of projects related to every step of a big data workflow, including data collection, storage, processing, and much more. The amount of projects that have been developed to either complement or replace these original elements has made the current definition of Hadoop unclear. For this reason, we often hear reference to the *Hadoop Ecosystem* instead, which encompasses these related projects and products. To fully understand Hadoop, one must look at both the project itself and the ecosystem that surrounds it. The Hadoop project itself currently consists of four modules:

•• *Hadoop distributed file system (HDFS)* A file system designed to store large amounts of data across multiple nodes of commodity hardware. HDFS has a master–slave architecture made up of data nodes which each store blocks of the data, retrieve data on demand, and report back to the name node with inventory. The name node keeps records of this inventory (references to file locations and metadata) and directs traffic to the data nodes upon client requests. This system has built-in fault tolerance, typically keeping three or more copies of each data block in case of disk failure. Additionally, there are controls in case of name node failure as well, in which a system will either have a secondary name node, or will write backups of metadata to multiple file systems.

•• *MapReduce* Data processing engine. A MapReduce job consists of two parts, a map phase, which takes raw data and organizes it into key/value pairs, and a reduce phase which processes data in parallel. A detailed discussion of this processing approach can be found in the following section.

•• *YARN ("Yet Another Resource Negotiator")* Prior the addition of YARN to the Hadoop project in version 2.0, Hadoop and MapReduce were tightly coupled, with MapReduce responsible for both cluster resource management and data processing. YARN has now taken over the resource management duties, allowing a separation between that infrastructure and the programming model. With YARN, if an application wants to run, its client has to request the launch of an application manager process from the resource manager, which then finds a node manager. The node manager then launches a container which executes the application process. For any readers who are familiar with previous versions of Hadoop, the job tracker responsibilities from MapReduce are now YARN, split between the resource manager, application master, and timeline server (which stores application history), while the old task tracker responsibilities are handled by the node managers. This change has improved upon many of the deficiencies present in the old MapReduce. YARN is able to run on larger clusters, more than doubling the amount of jobs and tasks it can handle before running into bottlenecks. Finally, YARN allows for a more generalized Hadoop which makes MapReduce just one type of YARN application. This means it can be left out altogether in favour of a different processing engine.

•• *Common-* A set of common utilities needed by the other Hadoop modules. It has native shared libraries that include Java implementations for compression codecs, I/O utilities, and error detection. Also included are interfaces and tools for configuration of rack awareness, authorization of proxy users, authentication, service-level authorization, data confidentiality, and the Hadoop Key Management Server (KMS).



MapReduce

The MapReduce approach to machine learning performs batch learning, in which the training data set is read in its entirety to build a learning model. The biggest drawback to this batch model is a lack of efficiency in terms of speed and computational resources. In a typical batch-oriented workflow, the set of training data is read from the HDFS to the mapper as a set of key-value pairs. The output, a list of keys and their associated values, is written to disk. In a classification task, for example, the initial key-value pair might be a filename and a list of instances, and the intermediate output from the mapper would be a list of each instance with its associated class. This intermediate data is then read into one or more reducers to train a model based on this list. The final model is then once again written to disk.

Spark

Spark, which was initially developed at the University of California, Berkeley and is now an Apache top-level project, is based on MapReduce but addresses a number of the deficiencies described above. Like Hadoop, it supports iterative computation and it improves on speed and resource issues by utilizing in-memory computation. Spark's approach to processing has seen widespread adoption in both research and industry. The main abstractions used in this project are called Resilient Distributed Datasets (RDD), which store data in-memory and provide fault tolerance without replication. RDDs can be understood as read-only distributed shared memory.

Storm

Storm is used for processing data in real-time and was initially conceived to overcome deficiencies of other processors in collecting and analyzing social media streams. Development on Storm began at BackType, a social media analytics company and continued at Twitter after a 2011 acquisition. The project was open sourced and became an Apache top-level project in September 2014. The machine learning community has been placing growing importance on real-time processing, and as a result, Storm is seeing increased adoption both in production and in research environments. The Storm architecture consists of spouts and bolts. A spout is the input stream (e.g. Twitter streaming API), while bolts contain most of the computation logic, processing data in the form of tuples from either the spout or other bolts. Networks of spouts and bolts, which are represented as directed graphs, are known as topologies.

Flink

Flink was developed at the Technical University of Berlin under the name Stratosphere. It graduated the Apache incubation stage in January 2015 and is now a top level project. It offers capability for both batch and stream processing, thus allowing for the implementation of a Lambda Architecture as described above. It is a scalable, in memory option that has APIs for both Java and Scala. It has its own runtime, rather than being built on top of MapReduce. As such, it can be integrated with HDFS and YARN, or run completely independent from the Hadoop ecosystem. Flink's processing model applies transformations to parallel data collections. Such transformations generalize *map* and *reduce* functions, as well as functions such as join, group, and iterate.

H2O

H2O is an open source framework that provides a parallel processing engine, analytics, math, and machine learning libraries, along with data preprocessing and evaluation tools. Additionally, it offers a web-based user interface, making learning tasks more accessible to analysts and statisticians who may not have strong programming backgrounds. For those who wish to tweak the implementations, it offers support for Java, R, Python, and Scala.

IV Conclusion

The general approach used is Distributed Fork/Join, a divide-and-conquer technique, which is reliable and suitable for massively parallel tasks. This is a method which breaks up a job into smaller jobs which run in parallel, resulting in dynamic fine-grain load balancing for MapReduce jobs as well as graphs and streams. They claim to be the fastest execution engine, but as of the time of this writing, no academic studies have been published which verify or refute In this paper, we reviewed studies on the data analytics from the traditional data analysis to the recent big data analysis. From the system perspective, the KDD process is used as the framework for these studies and is summarized into three parts: input, analysis, and output. From the perspective of big data analytics framework and platform, the discussions are focused on the performance-oriented and results-oriented issues. From the perspective of data mining problem, this paper gives a brief introduction to the data and big data mining algorithms which consist of clustering, classification, and frequent patterns mining technologies. To better understand the changes brought about by the big data, this paper is focused on the data analysis of KDD from the platform/framework to data mining. The open issues on computation, quality of end result, security, and privacy are then discussed to explain which open issues we may face.

References

1. G. Rigaut, A. Shevchenko, B. Rutz, M. Wilm, M. Mann, and B. S_eraphin, "A generic protein purification method for protein complex characterization and proteome exploration," Nature Biotechnol., vol. 17, no. 10, pp. 1030–1032, 1999.
2. Lyman P, Varian H. How much information 2003? Tech. Rep, 2004. [Online]. Available: http://www2.sims.berkeley.edu/research/projects/how-much-info-2003/printable_report.pdf.
3. Xu R, Wunsch D. Clustering. Hoboken: Wiley-IEEE Press; 2009.
4. Ding C, He X. K-means clustering via principal component analysis. In: Proceedings of the Twenty-first International Conference on Machine Learning, 2004, pp 1–9.
5. Kollios G, Gunopulos D, Koudas N, Berchtold S. Efficient biased sampling for approximate clustering and outlier detection in large data sets. IEEE Trans Knowl Data Eng. 2003; 15 (5):1170–87.
6. Fisher D, DeLine R, Czerwinski M, Drucker S. Interactions with big data analytics Interactions. 2012;19 (3) : 50 –9.
7. Laney D. 3D data management: controlling data volume, velocity, and variety, META Group, Tech. Rep. 2001. [Online]. Available: http://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf.
8. Van Rijmenam M. Why the 3v's are not sufficient to describe big data, BigData Startups, Tech. Rep. 2013. [Online]. Available: http://www.bigdata-startups.com/3vs-sufficient-describe-big-data/.
9. Borne K. Top 10 big data challenges a serious look at 10 big data v's, Tech. Rep. 2014. [Online]. Available: https://www.mapr.com/blog/top-10-big-data-challenges-look-10-big-data-v.
10. Press G. $16.1 billion big data market: 2014 predictions from IDC and IIA, Forbes, Tech. Rep. 2013. [Online]. Available: http://www.forbes.com/sites/ gilpress/2013/12/12/16-1-billion-big-data-market-2014-predictions-from-idc-and-iia/.
11. Big data and analytics—an IDC four pillar research area, IDC, Tech. Rep. 2013. [Online]. Available: http://www.idc.com/prodserv/FourPillars/bigData/index.jsp.
12. Taft DK. Big data market to reach $46.34 billion by 2018, EWEEK, Tech. Rep. 2013. [Online]. Available: http://www.eweek.com/database/big-data-market-to-reach-46.34-billion-by-2018.html.
13. Research A. Big data spending to reach $114 billion in 2018; look for machine learning to drive analytics, ABI Research, Tech. Rep. 2013. [Online]. Available: https://www.abiresearch.com/press/ big-data-spending-to-reach-114-billion-in-2018-l.
14. Furrier J. Big data market $50 billion by 2017—HP vertica comes out #1—according to wikibon research, SiliconANGLE, Tech. Rep. 2012. [Online]. Available: http://siliconangle.com/blog/2012/02/15/ big-data-market-15-billion-by-2017-hp-vertica-comes-out-1-according-to-wikibon-research/.
15. Kelly J, Vellante D, Floyer D. Big data market size and vendor revenues, Wikibon, Tech. Rep. 2014. [Online]. Available: http://wikibon.org/wiki/v/Big_Data_Market_Size_and_Vendor_Revenues.
16. Kogan J. Introduction to clustering large and high-dimensional data. Cambridge: Cambridge Univ Press; 2007.
17. Mitra S, Pal S, Mitra P. Data mining in soft computing framework: a survey. IEEE Trans Neural Netw. 2002; 13 (1):3–14.
18. Mehta M, Agrawal R, Rissanen J. SLIQ: a fast scalable classifier for data mining. In: Proceedings of the 5th International Conference on Extending Database Technology: Advances in Database Technology. 1996. pp 18–32.
19. Micó L, Oncina J, Carrasco RC. A fast branch and bound nearest neighbour classifier in metric spaces. Pattern Recogn Lett. 1996; 17 (7):731–9.
20. Djouadi A, Bouktache E. A fast algorithm for the nearest-neighbor classifier. IEEE Trans Pattern Anal Mach Intel. 1997; 19 (3):277–82.