International Journal of Computer Architecture and Mobility
(ISSN 2319-9229) Volume 4-Issue 4, April 2016

Evaluation of Big Data Tools (EBDT)

Shubham Gupta, Ritesh Jindal
CDGI, Indore
shubham02gupta@gmail.com, jritesh95@gmail.com

Abstract

Big data is a term for massive data sets having large, more varied and complex structure with the difficulties of storing, analyzing and visualizing for further processes or results. Big Data is driving radical changes in traditional data analysis platforms. Our analysis illustrates that the Big Data analytics is a fast-growing, influential practice and a key enabler for the social business. The insights gained from the user generated online contents and collaboration with customers is critical for success in the age of social media. The process of research into massive amounts of data to reveal hidden patterns and secret correlations named as big data analytics. These useful information's for companies or organizations with the help of gaining richer and deeper insights and getting an advantage over the competition. For this reason, big data implementations need to be analyzed and executed as accurately as possible. This paper presents an overview of big data's content, scope, samples, methods, advantages and challenges and discusses privacy concern on it.

Keywords: Big Data, massive data, data analytics.

Introduction

We are Awash in a flood of data today. In a broad range of application areas, data is being collected at unprecedented scale. Decisions that previously were based on guesswork, or on painstakingly constructed models of reality, can now be made based on the data itself. Such Big Data analysis now drives nearly every aspect of our modern society, including mobile services, retail, manufacturing, financial services, life sciences, and physical sciences.The explosion of Big Data has prompted much research to develop systems to support ultra-low latency service and real-time data analytics. Existing disk-based systems can no longer offer timely response due to the high access latency to hard disks. The unacceptable performance was initially encountered by Internet companies such as Amazon, Google, Facebook and Twitter, but is now also becoming an obstacle for other companies/organizations which desire to provide a meaningful real-time service (e.g., real-time bidding, advertising, social gaming). For instance, trading companies need to detect a sudden change in the trading prices and react instantly (in several milliseconds), which is impossible to achieve using traditional disk-based processing/storage systems. To meet the strict real-time requirements for analyzing mass amounts of data and servicing requests within milliseconds, an in-memory system/ database that keeps the data in the random access memory (RAM) all the time is necessary.

Apache Hadoop is an open source software framework for storage and large scale processing of data-sets on clusters of commodity hardware. Hadoop is an Apache top-level project being built and used by a global community of contributors and users. It is licensed under the Apache License 2.0.

Hadoop was created by Doug Cutting and Mike Cafarella in 2005. It was originally developed to support distribution for the Nutch search engine project. Doug, who was working at Yahoo! at the time and is now Chief Architect of Cloudera,

named the project after his son's toy elephant. Cutting's son was 2 years old at the time and just beginning to talk. He called his beloved stuffed yellow elephant "Hadoop".

Big data is all About 3 V's. As shown in fig 1.

Fig.1 Explaing 3 V's of Big data

The Apache Hadoop framework is composed of the following modules

1. Hadoop Common: contains libraries and utilities needed by other Hadoop modules
2. Hadoop Distributed File System (HDFS): a distributed file-system that stores data on the commodity machines, providing very high aggregate bandwidth across the cluster
3. Hadoop YARN: a resource-management platform responsible for managing compute resources in clusters and using them for scheduling of users' applications
4. Hadoop MapReduce: a programming model for large scale data processing

All the modules in Hadoop are designed with a fundamental assumption that hardware failures (of individual machines, or racks of machines) are common and thus should be automatically handled in software by the framework. Apache Hadoop's MapReduce and HDFS components originally derived respectively from Google's MapReduce and Google File System (GFS) papers. Beyond HDFS, YARN and MapReduce, the entire Apache Hadoop "platform" is now commonly considered to consist of a number of related projects as well: Apache Pig, Apache Hive, Apache HBase, and others.

For the end-users, though MapReduce Java code is common, any programming language can be used with "Hadoop Streaming" to implement the "map" and "reduce" parts of the user's program. The Hadoop framework itself is mostly written in the Java programming language, with some native code in C and command line utilities written as shell-scripts.

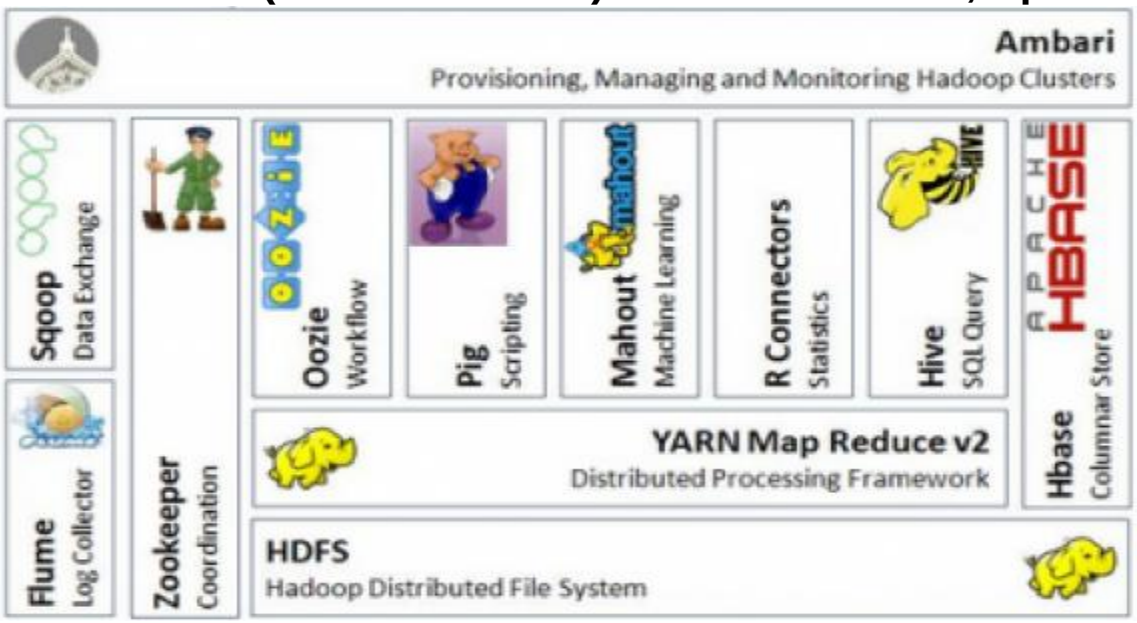

Fig.2 Apache Hadoop Ecosystem

HDFS and MapReduce

There are two primary components at the core of Apache Hadoop 1.x: the Hadoop Distributed File System (HDFS) and the MapReduce parallel processing framework. These are both open source projects, inspired by technologies created inside Google.

MapReduce is a processing technique and a program model for distributed computing based on java. The MapReduce algorithm contains two important tasks, namely Map and Reduce. Map takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value pairs). Secondly, reduce task, which takes the output from a map as an input and combines those data tuples into a smaller set of tuples. As the sequence of the name MapReduce implies, the reduce task is always performed after the map job. The major advantage of MapReduce is that it is easy to scale data processing over multiple computing nodes. Under the MapReduce model, the data processing primitives are called mappers and reducers. The Algorithm 1. Generally MapReduce paradigm is based on sending the computer to where the data resides.

MapReduce

2. MapReduce program executes in three stages, namely map stage, shuffle stage, and reduce stage.

a). Map stage: The map or mapper's job is to process the input data. Generally the input data is in the form of file or directory and is stored in the Hadoop file system (HDFS). The input file is passed to the mapper function line by line. The mapper processes the data and creates several small chunks of data.

b) Reduce stage: This stage is the combination of the Shuffle stage and the Reduce stage. The Reducer's job is to process the data that comes from the mapper. After processing, it produces a new set of output, which will be stored in the HDFS.

3. During a MapReduce job, Hadoop sends the Map and Reduce tasks to the appropriate servers in the cluster.

4. The framework manages all the details of data-passing such as issuing tasks, verifying task completion, and copying data around the cluster between the nodes.

5. Most of the computing takes place on nodes with data on local disks that reduces the network traffic.

6. After completion of the given tasks, the cluster collects and reduces the data to form an appropriate result, and sends it back to the Hadoop server.

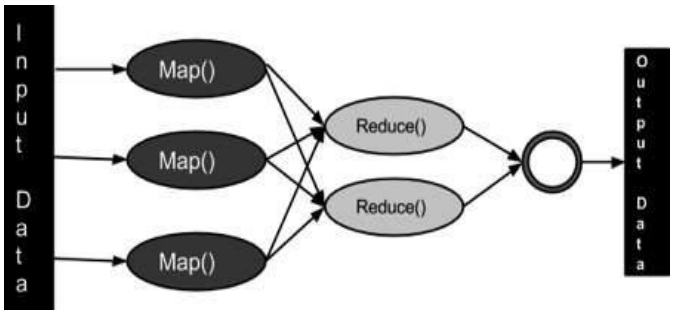

Fig.3 MapReduce Architecture

Hadoop Distributed File System (HDFS)

Hadoop File System was developed using distributed file system design. It is run on commodity hardware. Unlike other distributed systems, HDFS is highly fault tolerant and designed using low-cost hardware. HDFS holds very large amount of data and provides easier access. To store such huge data, the files are stored across multiple machines. These files are stored in redundant fashion to rescue the system from possible data losses in case of failure. HDFS also makes applications available to parallel processing.

Features of HDFS

- It is suitable for the distributed storage and processing.Hadoop provides a command interface to interact with HDFS.
- The built-in servers of namenode and datanode help users to easily check the status of cluster.
- Streaming access to file system data.
- HDFS provides file permissions and authentication.

Given below is the architecture of a Hadoop File System.

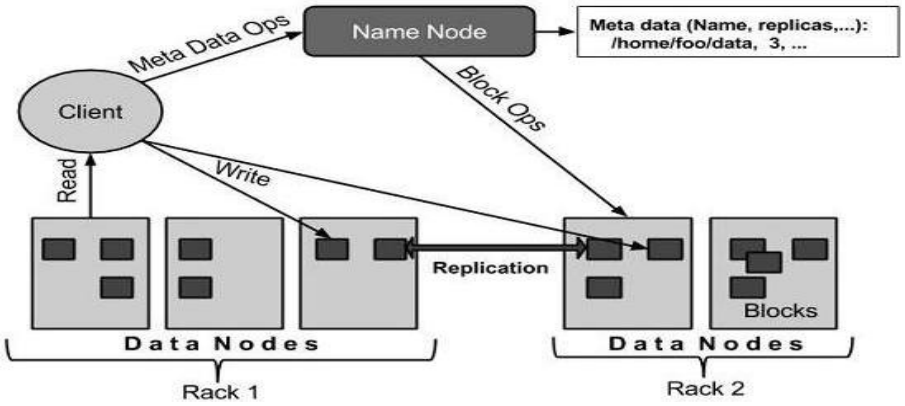

Fig.4 HDFS Architecture

Namenode

The namenode is the commodity hardware that contains the GNU/Linux operating system and the namenode software. It is a software that can be run on commodity hardware. The system having the namenode acts as the master server and it does the following tasks:

- Manages the file system namespace.
- Regulates client's access to files.
- It also executes file system operations such as renaming, closing, and opening files and directories.

Datanode

The datanode is a commodity hardware having the GNU/Linux operating system and datanode software. For every node (Commodity hardware/System) in a cluster, there will be a datanode. These nodes manage the data storage of their system.

- Datanodes perform read-write operations on the file systems, as per client request.
- They also perform operations such as block creation, deletion, and replication according to the instructions of the namenode.

Block

Generally the user data is stored in the files of HDFS. The file in a file system will be divided into one or more segments and/or stored in individual data nodes. These file segments are called as blocks. In other words, the minimum amount of data that HDFS can read or write is called a Block. The default block size is 64MB, but it can be increased as per the need to change in HDFS configuration.

Goals of HDFS

- ***Fault detection and recovery:*** Since HDFS includes a large number of commodity hardware, failure of components is frequent. Therefore HDFS should have mechanisms for quick and automatic fault detection and recovery.

- ***Huge datasets***: HDFS should have hundreds of nodes per cluster to manage the applications having huge datasets.

- ***Hardware at data:*** A requested task can be done efficiently, when the computation takes place near the data. Especially where huge datasets are involved, it reduces the network traffic and increases the throughput.

Known limitations of this approach in Hadoop 1.x

The allocation of work to TaskTrackers is very simple. Every TaskTracker has a number of available slots (such as "4 slots"). Every active map or reduce task takes up one slot. The Job Tracker allocates work to the tracker nearest to the data with an available slot. There is no consideration of the current system load of the allocated machine, and hence its actual availability. If one TaskTracker is very slow, it can delay the entire MapReduce job—especially towards the end of a job, where everything can end up waiting for the slowest task. With speculative execution enabled, however, a single task can be executed on multiple slave nodes.

Apache Hadoop NextGen MapReduce (YARN)

MapReduce has undergone a complete overhaul in hadoop-0.23 and we now have, what we call, MapReduce 2.0 (MRv2) or YARN. Apache Hadoop YARN is a sub-project of Hadoop at the Apache Software Foundation introduced in Hadoop 2.0 that separates the resource management and processing components. YARN was born of a need to enable a broader array of interaction patterns for data stored in HDFS beyond MapReduce. The YARN-based architecture of Hadoop 2.0 provides a more general processing platform that is not constrained to MapReduce.

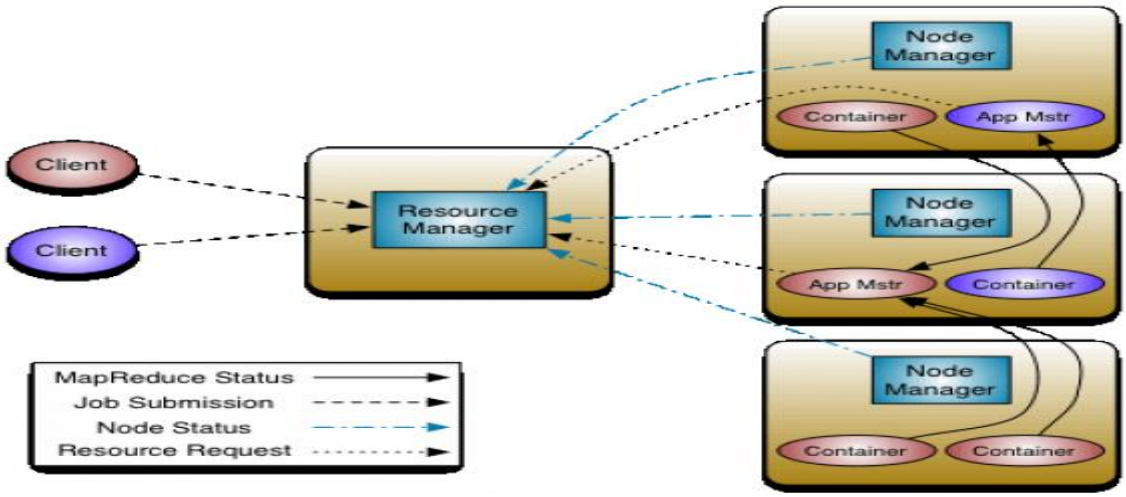

Fig.5 working of resource manager with node manager.

The fundamental idea of MRv2 is to split up the two major functionalities of the JobTracker, resource management and job scheduling/monitoring, into separate daemons. The idea is to have a global ResourceManager (RM) and per-application ApplicationMaster (AM). An application is either a single job in the classical sense of Map-Reduce jobs or a DAG of jobs.

The ResourceManager and per-node slave, the NodeManager (NM), form the data-computation framework. The ResourceManager is the ultimate

authority that arbitrates resources among all the applications in the system.

The per-application ApplicationMaster is, in effect, a framework specific library and is tasked with negotiating resources from the ResourceManager and working with the NodeManager(s) to execute and monitor the tasks.

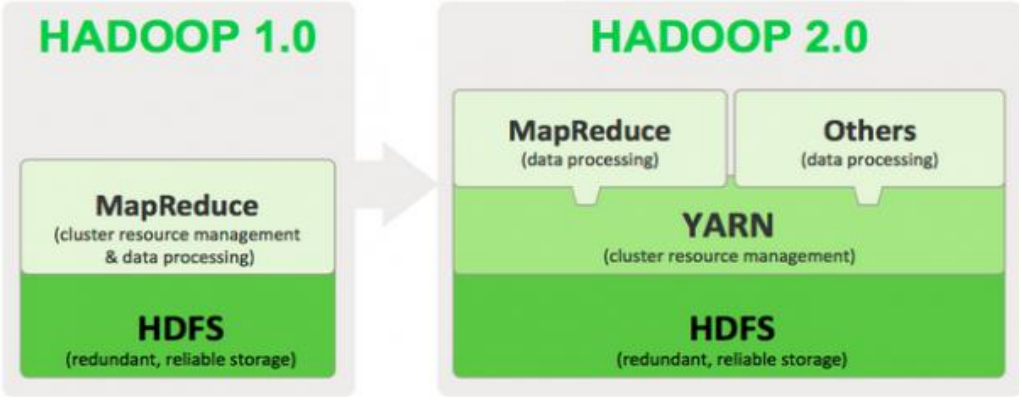

Fig.6 Hadoop 1.0 vs Hadoop 2.0

As part of Hadoop 2.0, YARN takes the resource management capabilities that were in MapReduce and packages them so they can be used by new engines. This also streamlines MapReduce to do what it does best, process data. With YARN, you can now run multiple applications in Hadoop, all sharing a common resource management. Many organizations are already building applications on YARN in order to bring them IN to Hadoop.

Fig.7 YARN Architecture

As part of Hadoop 2.0, YARN takes the resource management capabilities that were in MapReduce and packages them so they can be used by new engines. This also streamlines MapReduce to do what it does best, process data. With YARN, you can now run multiple applications in Hadoop, all sharing a common resource management. Many organizations are already building applications on YARN in order to bring them IN to Hadoop. When enterprise data is made available in HDFS, it is important to have multiple ways to process that data. With Hadoop 2.0 and YARN organizations

can use Hadoop for streaming, interactive and a world of other Hadoop based applications.

What YARN does

YARN enhances the power of a Hadoop compute cluster in the following ways:

- Scalability: The processing power in data centers continues to grow quickly. Because YARN ResourceManager focuses exclusively on scheduling, it can manage those larger clusters much more easily.
- Compatibility with MapReduce: Existing MapReduce applications and users can run on top of YARN without disruption to their existing processes.
- Improved cluster utilization: The ResourceManager is a pure scheduler that optimizes cluster utilization according to criteria such as capacity guarantees, fairness, and SLAs. Also, unlike before, there are no named map and reduce slots, which helps to better utilize cluster resources.
- Support for workloads other than MapReduce: Additional programming models such as graph processing and iterative modeling are now possible for data processing. These added models allow enterprises to realize near real-time processing and increased ROI on their Hadoop investments.
- Agility: With MapReduce becoming a user-land library, it can evolve independently of the underlying resource manager layer and in a much more agile manner.

How YARN works

The fundamental idea of YARN is to split up the two major responsibilities of the JobTracker/TaskTracker into separate entities:

- a global ResourceManager
- a per-application ApplicationMaster
- a per-node slave NodeManager and
- a per-application container running on a NodeManager

The ResourceManager and the NodeManager form the new, and generic, system for managing applications in a distributed manner. The ResourceManager is the ultimate authority that arbitrates resources among all the applications in the system. The per-application ApplicationMaster is a framework-specific entity and is tasked with negotiating resources from the ResourceManager and working with the NodeManager(s) to execute and monitor the component tasks. The ResourceManager has a scheduler, which is responsible for allocating resources to the various running applications, according to constraints such as queue capacities, user-limits etc. The scheduler performs its scheduling function based on the resource requirements of the applications. The NodeManager is the per-machine slave, which is responsible for launching the applications' containers, monitoring their resource usage (cpu, memory, disk, network) and reporting the same to the ResourceManager. Each ApplicationMaster has the responsibility of negotiating appropriate resource containers from the scheduler, tracking their status, and monitoring their progress. From the system perspective, the Application Master runs as a normal container.

Real time processing

The Apache Hadoop ecosystem has become a preferred platform for enterprises seeking to process and understand large-scale data in real time. Technologies like Apache Kafka, Apache Flume, Apache Spark, Apache Storm, and Apache Samza are increasingly pushing the envelope on what is possible. It is often tempting to bucket large-scale streaming use cases together but in reality they tend to break down into a few different architectural patterns, with different components of the ecosystem better suited for different problems.

Streaming Patterns

The basic streaming patterns (often used in tandem) are:

- Stream ingestion: Involves low-latency persisting of events to HDFS, Apache HBase, and Apache Solr.
- Near Real-Time (NRT) Event Processing with External Context: Takes actions like alerting, flagging, transforming, and filtering of events as they arrive. Common use cases, such as NRT fraud detection and recommendation, often demand low latencies under 100 milliseconds.
- NRT Event Partitioned Processing: Similar to NRT event processing, but deriving benefits from partitioning the data—like storing more relevant external information in memory. This pattern also requires processing latencies under 100 milliseconds.

Challenges

Big data analysis is the process of applying advanced analytics and visualization techniques to large data sets to uncover hidden patterns and unknown correlations for effective decision making. The analysis of Big Data involves multiple distinct phases which include data acquisition and recording, information extraction and cleaning, data integration, aggregation and representation, query processing, data modeling and analysis and Interpretation. Each of these phases introduces challenges.

1. Heterogeneity and Incompleteness

The difficulties of big data analysis derive from its large scale as well as the presence of mixed data based on different patterns or rules (heterogeneous mixture data) in the collected and stored data. In the case of complicated heterogeneous mixture data, the data has several patterns and rules and the properties of the patterns vary greatly. Data can be both structured and unstructured. 80% of the data generated by organizations are unstructured. They are highly dynamic and does not have particular format. It may exists in the form of email attachments, images, pdf documents, medical records, X rays, voice mails, graphics, video, audio etc. and they cannot be stored in row/ column format as structured data. Transforming this data to structured format for later analysis is a major challenge in big data mining. So new technologies have to be adopted for dealing with such data. Incomplete data creates uncertainties during data analysis and it must be managed during data analysis. Doing this correctly is also a challenge. Incomplete data refers to the missing of data field values for some samples. The missing values can be caused by different realities, such as the malfunction of a sensor node, or some systematic policies to intentionally skip some values.

2. Scale and complexity

Managing large and rapidly increasing volumes of data is a challenging issue. Traditional software tools are not enough for managing the increasing volumes of data. Data analysis, Computer Science & Information Technology (CS & IT) 135 organization, retrieval and modeling are also

challenges due to scalability and complexity of data that needs to be analysed.

3. Timeliness

As the size of the data sets to be processed increases, it will take more time to analyse. In some situations results of the analysis is required immediately. For example, if a fraudulent credit card transaction is suspected, it should ideally be flagged before the transaction is completed by preventing the transaction from taking place at all. Obviously a full analysis of a user's purchase history is not likely to be feasible in real time. So we need to develop partial results in advance so that a small amount of incremental computation with new data can be used to arrive at a quick determination. Given a large data set, it is often necessary to find elements in it that meet a specified criterion. In the course of data analysis, this sort of search is likely to occur repeatedly. Scanning the entire data set to find suitable elements is obviously impractical. In such cases Index structures are created in advance to permit finding qualifying elements quickly. The problem is that each index structure is designed to support only some classes of criteria.

Advantages

1. Cost reduction

For processing and storing vast quantities of new data in a data warehouse, for example, companies are using Hadoop clusters for that purpose, and moving data to enterprise warehouses as needed for production analytical applications. Well-established firms like Citi, Wells Fargo and USAA all have substantial Hadoop projects underway that exist alongside existing storage and processing capabilities for analytics. While the long-term role of these technologies in an enterprise architecture is unclear, it's likely that they will play a permanent and important role in helping companies manage big data.

2. Faster, better decision making

Analytics has always involved attempts to improve decision making, and big data doesn't change that. Large organizations are seeking both faster and better decisions with big data, and they're finding them. Driven by the speed of Hadoop and in-memory analytics, several companies I researched were focused on speeding up existing decisions. Some firms are more focused on making better decisions analyzing new sources of data. For example, health insurance giant United Healthcare is using "natural language processing" tools from SAS to better understand customer satisfaction and when to intervene to improve it. It starts by converting records of customer voice calls to its call center into text and searching for indications that the customer is dissatisfied. The company has already found that the text analysis improves its predictive capability for customer attrition models.

3. New products and services

Perhaps the most interesting use of big data analytics is to create new products and services for customers. Online companies have done this for a decade or so, but now predominantly offline firms are doing it too. Verizon Wireless is also pursuing new offerings based on its extensive mobile device data. In a business unit called Precision Market Insights, Verizon is selling information about how often mobile phone users are in certain locations, their activities and backgrounds. Customers thus far have included malls, stadium owners and billboard firms.

Application

The term 'Big Data' is a massive buzzword at the moment and many say big data is all talk and no action. This couldn't be further from the truth. With this post, I want to show how big data is used today to add real value. Eventually, every aspect of our lives will be affected by big data. However, there are some areas where big data is already making a real difference today.

1. Understanding and Targeting Customers

This is one of the biggest and most publicized areas of big data use today. Here, big data is used to better understand customers and their behaviors and preferences. Companies are keen to expand their traditional data sets with social media data, browser logs as well as text analytics and sensor data to get a more complete picture of their customers. Using big data, Telecom companies can now better predict customer churn. Wal-Mart can predict what products will sell, and car insurance companies understand how well their customers actually drive. Even government election campaigns can be optimized using big data analytics. Some believe, Obama's win after the 2012 presidential election campaign was due to his team's superior ability to use big data analytics.

2. Understanding and Optimizing Business Processes

Big data is also increasingly used to optimize business processes. Retailers are able to optimize their stock based on predictions generated from social media data, web search trends and weather forecasts. One particular business process that is seeing a lot of big data analytics is supply chain or delivery route optimization. Here, geographic positioning and radio frequency identification sensors are used to track goods or delivery vehicles and optimize routes by integrating live traffic data, etc.

3. Improving Healthcare and Public Health

The computing power of big data analytics enables us to decode entire DNA strings in minutes and will allow us to find new cures and better understand and predict disease patterns. Just think of what happens when all the individual data from smart watches and wearable devices can be used to apply it to millions of people and their various diseases. The clinical trials of the future won't be limited by small sample sizes but could potentially include everyone! Big data techniques are already being used to monitor babies in a specialist premature and sick baby unit. By recording and analyzing every heart beat and breathing pattern of every baby, the unit was able to develop algorithms that can now predict infections 24 hours before any physical symptoms appear. That way, the team can intervene early and save fragile babies in an environment where every hour counts.

4. Improving Sports Performance

Most elite sports have now embraced big data analytics. We have the IBM SlamTracker tool for tennis tournaments; we use video analytics that track the performance of every player in a football or baseball game, and sensor technology in sports equipment such as basket balls or golf clubs allows us to get feedback (via smart phones and cloud servers) on our game and how to improve it.

5. Improving Science and Research

Science and research is currently being transformed by the new possibilities big data brings. Take, for example, CERN, the Swiss nuclear physics lab with its Large Hadron Collider,

the world's largest and most powerful particle accelerator. Experiments to unlock the secrets of our universe – how it started and works - generate huge amounts of data. The CERN data center has 65,000 processors to analyze its 30 petabytes of data. However, it uses the computing powers of thousands of computers distributed across 150 data centers worldwide to analyze the data. Such computing powers can be leveraged to transform so many other areas of science and research.

6. Optimizing Machine and Device Performance

Big data analytics help machines and devices become smarter and more autonomous. For example, big data tools are used to operate Google's self-driving car. The Toyota Prius is fitted with cameras, GPS as well as powerful computers and sensors to safely drive on the road without the intervention of human beings. Big data tools are also used to optimize energy grids using data from smart meters. We can even use big data tools to optimize the performance of computers and data warehouses.

7. Improving Security and Law Enforcement

Big data is applied heavily in improving security and enabling law enforcement. I am sure you are aware of the revelations that the National Security Agency (NSA) in the U.S. uses big data analytics to foil terrorist plots (and maybe spy on us). Others use big data techniques to detect and prevent cyber attacks. Police forces use big data tools to catch criminals and even predict criminal activity and credit card companies use big data use it to detect fraudulent transactions.

8. Improving and Optimizing Cities and Countries

Big data is used to improve many aspects of our cities and countries. For example, it allows cities to optimize traffic flows based on real time traffic information as well as social media and weather data. A number of cities are currently piloting big data analytics with the aim of turning themselves into Smart Cities, where the transport infrastructure and utility processes are all joined up. Where a bus would wait for a delayed train and where traffic signals predict traffic volumes and operate to minimize jams.

References

[1] https://en.wikipedia. org/wiki /Big_data

[2]http://www.sas.com/en_us/insights/big-data/what-is-big-data.html

[3]https://opensource.com/life/14/8/intro-apache-hadoop-big-data

[4]https://www.ibm.com/developerworks/vn/library/contest/dw-freebooks/Tim_Hieu_Big_Data/Understanding_BigData.PDF

[5]http://www.planet-data.eu/sites/default/files/presentations/Big_Data_Tutorial_part4.pdf

[6]https://hadoopecosystemtable.github.io/

[7]https://opensource.com/resources/big-data

[8]https://www.oreilly.com/ideas/what-is-big-data

[9]http://www.tutorialspoint.com/hadoop/hadoop_mapreduce.htm

[10]http://www.tutorialspoint.com/hadoop/hadoop_hdfs_overview.htm