An Architecture for Big Data Analysis in the Context of Social Media: the case of Twitter

Belesti Melesse
HiLCoE, Computer Science Programme, Ethiopia
belestimm@yahoo.com

Tibebe Beshah
HiLCoE, Ethiopia
School of Information Science, Addis Ababa University
tibebe.beshah@gmail.com

Abstract

Big Data analytics puts lots of pressure on ICT providers for developing new tools and technology to manage complex data. Challenges include storing and processing of huge volume of unstructured data, handling high-velocity data streams, cleansing noise and abnormality in the data, analyzing the data and finding value or meaningful results. Current tools and technologies are incapable to store, process and analyze huge amount of diverse data. In this paper, we proposed an architecture that enables an effective storage and analysis of unstructured data and developed a prototype to evaluate and test it. This paper presents the work of investigating and designing a Big Data analysis solution using a MapReduce platform named Hadoop and a data warehouse infrastructure built on top of Hadoop called Hive which enables the analysis of unstructured data. The proposed architecture is validated through the development of a prototype that can analyze unstructured data using Hadoop MapReduce, HDFS (Hadoop File System), and Hive. We also evaluated whether this architecture is achieving its goals and objectives. The evaluation is conducted through streaming Twitter data, storing, processing, finally fetching and performing sentiment analysis. Twitter, one of the largest social media sites, is used as a data source in our experiment. Hence, this study provided an architecture that helps to address problems related to Big Data analysis and furthermore it is novel due to the use of MapReduce and Hive in a unique way for Twitter data analysis. The results of this work can be applied by enterprises in sentiment analysis to understand how their customers feel about a particular product or service and to track how those opinions change over time, and also to get information regarding the relative performances of their competitors.

Keywords: Big Data frameworks; Big Data architecture; Big Data Analytics

1. Introduction

The emerging Big Data paradigm, owing to its broader impact, has profoundly transformed our society and will continue to attract diverse attentions from both technological experts and the public in general [1]. Big Data encloses large volume of complex structured, semi-structured, and unstructured data, which is beyond the processing powers of conventional databases [2].

As far back as 2001, Doug Laney (Industry Analyst) expressed the now conventional definition of Big Data as the 3Vs of Big Data like volume, velocity and variety [3]. According to [1], there are five attributes to classify Big Data, as listed below.

- Volume: the size of the data to be processed [2].
- Velocity: the speed at which the data is being produced or the frequency with which it is delivered [4].
- Variety: the data form, i.e., structured, semi-structured, and unstructured [1].
- Veracity: it includes data quality issues, in particular issues such as their incompleteness, errors, noise and other attributes affecting quality [5].
- Value: it refers to data usability and the practical possibilities of their use in decision-making and, consequently, to the possibility of

generating important value for an organization [5].

Therefore Big Data refers to datasets which for one of the many reasons (Volume, Velocity, Variety, Veracity, or Value) do not fit into a conventional relational database [3]. Unstructured data refers to information that either does not have a pre-defined data model or does not fit well into relational tables [6].

2. Literature Review and Related Work

Selected resource materials were thoroughly analyzed and investigated to understand the state of the art for the body of knowledge under review and explore empirical studies made.

2.1 Big Data Analytics Frameworks

Different types of frameworks are required to run different types of analytics. A variety of workloads are present in a large-scale data processing enterprise. In order to attain a business goal, it needs to see a blend of workloads deployed: batch-oriented processing, OLTP (Online Transaction Processing), stream processing, and interactive ad-hoc query [7].

- Apache Hadoop is a framework that allows handling distributed processing of Big Data across clusters of computers using simple programming models. Hadoop is designed to scan large data sets to produce results through a distributed and highly scalable batch processing systems [8].
- In order to carry out rigorous real-time analysis, Storm (Hadoop for real-time) has been developed. Storm is a distributed real-time computation system developed and released as open source by Twitter [2].
- Apache Drill is a distributed system for interactive ad-hoc analysis of large-scale datasets. Many times it occurs that a human sits in front of a business application and needs to execute ad-hoc queries as per business needs. Apache drill will provide the solution for all these issues [7].

Apache Hadoop is suited for workload where time is not a critical feature whereas Storm is well suited for data stream analysis in which analysis is made in real time and Apache drill is best for interactive and ad-hoc analysis [7]. In line with this, Apache Hadoop has been preferred for our study.

2.2 The Hadoop Framework

Hadoop analytics takes a fundamental approach to a single large workload, mapping it into smaller sub-workloads. These smaller workloads are then merged to obtain the end result. Hadoop manages this workload by assigning a large cluster of low cost nodes built with commodity hardware. Hadoop clusters usually run MapReduce jobs in parallel, in addition to background activities such as importing/exporting data from Hadoop File System (HDFS) [6]. Apache Hadoop provides a reliable shared storage (storage provided by HDFS) and analysis (analysis provided by MapReduce) system for large scale data processing [3].

a. Hadoop Distributed File System (HDFS)

The Apache Hadoop project defines HDFS such as primary storage system used by Hadoop applications [3]. An HDFS cluster has two types of nodes operating in a master-worker pattern: a NameNode (the master) and a number of DataNodes (workers). The NameNode handles the file system tree and the metadata for all the files and directories in the tree. DataNodes are the workhorses of the file system [9].

b. MapReduce

MapReduce is a technique for distributing a task across multiple nodes. Each node that processes data stored on that node consists of two phases: map and reduce [3]. Map phase divides the workload into smaller sub-workloads and allocates tasks to mapper, which processes each unit block of data. Reduce phase analyzes and merges the input from mappers to produce the final output. The final output is written to the HDFS in the cluster [6]. The MapReduce framework comprises of a single master JobTracker and one slave Task Tracker per cluster-node. The master is responsible for scheduling the jobs'

component tasks on the slaves, monitoring them and re-executing the failed tasks. The slaves perform the tasks as directed by the master [10].

c. HBase

It is a NoSQL database that sits on top of HDFS based on Google's Big Table. HBase cannot directly operate SQL [2]. HBase is highly configurable, providing a great deal of flexibility to address large volumes of data efficiently [11].

d. Hive

Hive is a batch-oriented, data warehousing layer built on the core elements of Hadoop (HDFS and MapReduce). It provides users who know SQL with a simple SQL-like operation called Hive Query Language (HiveQL). With Hive, one can get the best of both worlds: SQL-like access to structured data and advanced Big Data analysis with MapReduce. Unlike most data warehouses, Hive is not designed for quick responses to queries [11]. Hive converts HiveQL query into a series of MapReduce jobs for implementation on a Hadoop cluster. It arranges data into tables, which provide a means for attaching structure to data stored in HDFS [12].

e. Hue

Hue is an open source web-based interface for making it easier to use Apache Hadoop and its ecosystem. Hue aggregates the most common Apache Hadoop components into a single interface and targets the user experience. Its main goal is to have users just use Hadoop without worrying about the underlying complexity or using a command line. It features a file browser for HDFS, a job designer/browser for MapReduce, Hive, Pig editors and more [13, 14].

2.3 Related Work

Nowadays, most of the information saved in enterprises is unstructured. Retrieval and extraction of the information is essential and significant in semantic web areas. Text mining and natural language processing are two mechanisms with their methods for knowledge discovery from textual context in documents. Das and Kumar [15] developed a framework for analyzing unstructured data. Their proposed approach consists of acquiring unstructured data from public tweets of Twitter, storing the data in a NoSQL database like HBase, and retrieving and analyzing the data. They investigated and tested the first phase, which is acquiring unstructured data from public tweets of Twitter. The processes used in the first phase are registering the application with the Twitter development, authenticating the application, sending the requests with Java coding, and parsing the result as an XML and the result obtained needs to be parsed for filtering [15]. However, there are many limitations in the paper. Data acquired from Twitter were stored in HBase after sentiment analysis. Furthermore, MapReduce was not used either for processing or analyzing sentiments even though they are the core components of apache Hadoop. Therefore, parallel processing of data during analysis is impossible. Since HBase data requires Java coding, business analysts who are supposed to use this solution need to have Java knowledge and skills.

Processing large datasets obtained from diverse sources is a challenging task as it requires tremendous storing and processing capabilities. Distributing the data across multiple processing units and parallel processing units produces linear improved processing speeds. Boja *et al.* [24] investigated the problem of storing, processing and retrieving meaningful insight from petabytes of data. They developed a Distributed Parallel Architecture for Big Data that can be used to solve the analyzed problem. When distributing the data is critical, each processing unit is assigned the same number of records and that all the related data sets reside on the same processing unit. Using a multi-layer architecture to acquire, transform, load and analyze the data, ensures that each layer can use the best of breed for its specific task. But the authors focused only on the volume aspect of Big Data while Big Data has five attributes.

The increasing gap between data processing and input/output capacity on High End Computing (HEC) machines makes a severe bottleneck for data analysis. Instead of transferring data from its source to the output storage, in-situ analytics processes output data

while simulations are running. Since different data processing techniques have different impact on performance and cost, there is a consequent need for flexibility in the location of data analytics.

Zou *et al.* [25] proposed a flexible data analytics framework and developed flexible data analytics prototype system for analytics placement. This flexible placement strategy merges data compression and visualization query – two algorithms, which can be dynamically selected and switched to achieve the best I/O performance with features profiling and real-time system resource status monitoring. The experiment showed that the data reduction ratio and available processors are two most important factors to impact the potential location of the analysis. However, much more could have been done to perform more investigation on these factors.

3. The Proposed Architecture

Recognizing the limitations in the framework by Das and Kumar [15], we proposed a three layer architecture named data acquisition layer, data storage and processing layer, and user access layer. Hence, the novelty of our work lies in the storage and processing layer of the architecture by incorporating HDFS, MapReduce and Hive instead of HBase. Components of the architecture include Twitter API, HDFS storage, MapReduce, Hive, Hue and a front end application tool as it is shown in Figure 1. The proposed architecture enables processing and analyzing Twitter datasets while addressing the limitations mentioned above and it has three layers. Twitter is chosen because Twitter postings are easier to analyze due to the length limit as the authors are usually straight to the point. Thus, it is often easier to attain high sentiment analysis accuracy. Reviews are also easier because they are highly focused with little irrelevant information [16].

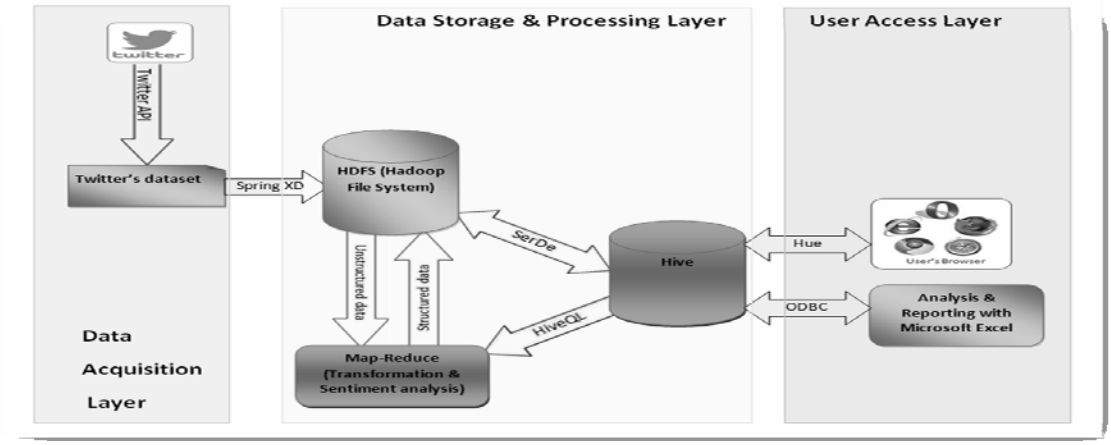

Figure 1: The Proposed Architecture

3.1 Data Acquisition Layer

The first layer gets data from Twitter and makes the datasets ready to be taken by HDFS. This layer includes Twitter API and Twitter's datasets.

This API employs the push strategy for data retrieval. Once a request for information is made, the streaming APIs provide a continuous stream of updates with no further input from the user. The limitation of Twitter APIs is that it can be too restrictive for certain types of applications. The APIs only grant access to a 1% sample of the Twitter data, and there are concerns about the sampling strategy and the quality of Twitter data obtained via the API. However, Twitter API works well for a lot of individuals that just want to access Twitter data for light analytics or statistical analysis and they are free to use [17]. Therefore, Twitter API is used to collect and download public tweets of Twitter; one is for cost reason and the other is data samples obtained with Twitter APIs are sufficient for experimenting with the

proposed architecture [17]. Once an authorized API request is sent, Twitter database provides the data in JSON (Java Script Object Notation) format.

3.2 Data Storage and Processing Layer

This layer stores and processes the Twitter datasets. This layer includes HDFS, MapReduce, and Hive as described below.

a. HDFS Storage

HDFS cluster is the storage structure in the architecture and hence it stores all the data in the system. It stores both the raw data and the data after processed and analyzed. The HDFS cluster has two types of nodes operating in a master-worker pattern: a NameNode (the master) and a number of DataNodes (slaves). The NameNode is responsible for maintaining the directory structure of the file system and tracking where each file data is kept in DataNodes of the Hadoop cluster. So it is the NameNode which knows what blocks on which data nodes make up the complete file where as the DataNodes are where data are actually stored.

b. Mapping and Reducing

The MapReduce cluster processes data sets in a distributed fashion. It works based on the concept of breaking the data processing task into two smaller tasks of mapping and reduction. It also works in master slave architecture: a JobTracker (the master) and a number of TaskTrackers (slaves). The MapReduce jobs break the input data set into independent blocks. Then the master is responsible for dispatching jobs to TaskTrackers, keeping track of the job progress, re-executing the failed tasks, and returning the result to the storage. Then slaves perform the tasks as directed by the master.

c. Hive

In our architecture, we preferred Hive to HBase for many reasons. Hive converts SQL query into a series of MapReduce jobs for implementation on a MapReduce cluster [9]. Systems that require real-data processing can't tolerate latency since they have to detect patterns and attempt to identify either opportunities or threats so as to take immediate actions. Some of them are real-time bidding systems in stock market, credit (debit) card fraud detection system, heart rate monitoring system in hospitals, air craft control system, and traffic light system [18]. On the other hand, many business firms can handle analysis of customers' tweets with Hive at least every one or two hours while this latency is not affecting their business. In addition, HBase requires writing MapReduce codes with Java which is not an issue in Hive [19].

d. Spring XD

Spring XD is a unified, distributed, and extensible service for data ingestion and data export. It offers high throughput data ingestion from a variety of input sources into big data store such as HDFS, and high throughput data export from HDFS to Hive or HBase [20].

e. JSON SerDe (Serializer/Deserializer)

Hive uses the SerDe interface for input/output. The interface manages both serialization and de-serialization and also interpreting the results of serialization as individual fields for processing. JSON SerDe enables Hive to read from and write into HDFS in JSON format [21].

3.3 User Access Layer

This layer provides access to processed data and is used to manage analysis and report requests. It uses MS Excel and web browsers as front end application tools. It also incorporates Open Database Connectivity (ODBC) and Hue web server to create connection between front end applications and Hive.

4. Results and Discussion

The validity of the architecture is warranted through the development of a prototype. The prototype has been done in a two node cluster environment in master slave architecture. The cluster runs Hadoop, MapReduce, HDFS, and Hive. Breen's algorithm has been used in performing sentiment analysis in our research. The general idea is to calculate a sentiment score for each tweet so we can know how positive or negative is the posted message.

There are different ways to calculate such scores. We used a very simple yet useful approach [22].

Sentiment Score = Sum of Sentiment Scores of words in a tweet

Score of positive words is considered to be 1, score of negative words is considered to be -1, and score of neutral words is considered to be 0. Then the total score for a tweet is calculated by adding sentiment scores of words in a tweet. If Score is greater than 0, this means that the sentence has an overall 'positive opinion'. If Score is less than 0, this means that the sentence has an overall 'negative opinion'. If Score is equal to 0, then the sentence is considered to be a 'neutral opinion' [22].

Subsequently sentiment dictionary and time-zone-map files have been downloaded [23] and stored together with the raw sentiment data. Then, a Hive script that can refine, transform, and analyze the raw sentiment data has been written in HiveQL (Hive Query Language). This script should be executed to clean the raw data, transform to structured data and compute sentiment score for each tweet as shown in Figure 2.

Figure 2: Executing Hive Query

As it can be seen in Figure 3, with our sample data, the result shows that 17.86 percent of the tweets have positive opinion, 78.57 percent of the tweets have neutral opinions, and 3.57 percent of the tweets have negative opinions about Samsung Galaxy S6 mobile. Therefore, business analysts of Samsung Company can take actions so as to respond to the customers' needs. It is also possible to make sentiment analysis of public tweets for other similar products in competitor companies. This comparison will provide the status of Samsung Galaxy S6 mobile in the eyes of customers among other products like iPhone and Huawei smart phones.

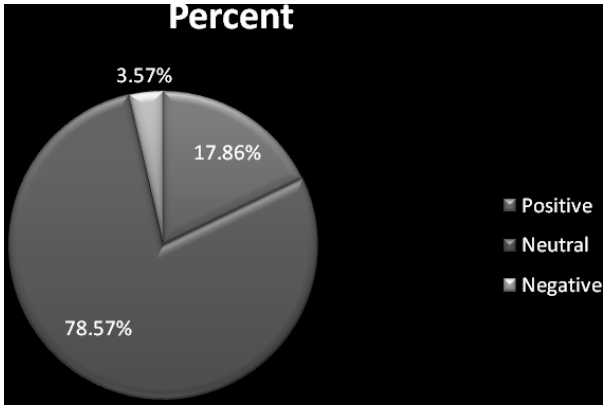

Figure 3: Sentiment Summary

Our results have suggested that the proposed Big Data solution is effective and it therefore added enhancement to the effectiveness of the framework

proposed by Das and Kumar [15]. This is exhibited through the following points:

- It involved Hive, MapReduce, and HDFS.
  - The advantage of using MapReduce here is its ability in processing large sets of data in parallel, by distributing the jobs among the task tracker nodes in the cluster. A MapReduce job usually splits the input data set into independent chunks which are processed by the map tasks in a completely parallel manner [11].
  - In using HDFS we also get multiple advantages. HDFS can store all kinds of data (both structured and unstructured) without prior organization. In addition, HDFS stores the replicas of each data blocks on different DataNodes to provide both high speed data transfer and high availability of data [9].
  - Using Hive instead of HBase is another new feature of our architecture. Analyzing Hive data doesn't require Java coding; rather it is done with HiveQL scripts, and HiveQL is already known by many business analysts since anyone familiar with SQL can interact with Hive with HiveQL [19].
- It allows storing unstructured data. As a Big Data analysis architecture, we should be able to store both structured and unstructured data in our system. This is made possible with the use of HDFS before and after data transformation and analysis.
- Sentiment analysis is done using sentiment analysis algorithm in MapReduce. Managing the sentiment analysis task with MapReduce provides performance gain as MapReduce can do it with parallel processing using multiple task tracker nodes in the cluster.
- HiveQL is used to write or execute MapReduce jobs. Instead of writing MapReduce jobs in Java, which demands serious software development skills, it is possible to write MapReduce jobs in HiveQL since Hive allows using HiveQL scripts [19].

Hence, our architecture is much better than the works of Das and Kumar [15]. This is related to the use of HDFS and MapReduce in the Hadoop system and the use of Hive instead of HBase. In addition, we have shown that a business analyst of an enterprise can get public Tweets about concerned products and services to make analysis and suggest appropriate actions for customers' reaction effectively.

5. Conclusion and Future Work

We have proposed an architecture for Big Data analysis and we have also showed how the proposed solution could work by streaming public tweets of Twitter, ingesting the raw data in to Hadoop system, and doing sentiment analysis during the experiment. We have also showed that raw Twitter data sets can be stored in HDFS for future use and sentiment analysis can be done with MapReduce. Furthermore, a business analyst can easily be familiar to write HiveQL scripts to perform sentiment analysis on public tweets tracked from Twitter based on a given keyword. Therefore, we believe that this proves the effectiveness of the proposed architecture.

This study will help business institutions in sentiment analysis to understand how their customers feel about a particular product or service and to track how those opinions change over time, and also to get information on how their competitors are doing in comparison. Hence, business organizations can deploy and use the result of this study to continuously monitor the feedbacks of their customers and take timely actions in response.

There are a lot of works that can be done in the future. The Hadoop and Hive newer versions have lots of bug fixes and enhancements. We will upgrade and implement it with newer versions to see if there are any improvements. Adjusting some configurations in Hadoop and Hive for a better performance and efficiency will also be done. Moreover, this research can be extended to other social media sources of Big Data like face book and LinkedIn.

References

[1] Hu, H., et al., “Towards Scalable Systems for Big Data Analytics: A Technology Tutorial”, IEEE 2014.

[2] Casado, R. and M. Younas, “Emerging Trends and Technologies in Big Data Processing. Concurrency and Computation: Practice and Experience”, 2014.

[3] Kumar, R., et al., “Apache Hadoop, NoSQL and NewSQL Solutions of Big Data”, International Journal of Advance Foundation and Research in Science & Engineering (IJAFRSE). 1(6): p. 28-36.

[4] Zadrozny, P. and R. Kodali, “Big Data and Splunk, in Big Data Analytics Using Splunk”, 2013, Springer. p. 1-7.

[5] Wieczorkowski, J. and P. Polak, “Big Data: Three-aspect Approach”, Online Journal of Applied Knowledge Management.

[6] Bakshi, K. “Considerations for Big Data: Architecture and Approach”, in Aerospace Conference, 2012 IEEE.

[7] Chandarana, P. and M. Vijayalakshmi, “Big Data Analytics Frameworks”, in Circuits, Systems, Communication and Information Technology Applications (CSCITA), 2014 IEEE International Conference.

[8] Akerkar, R., “Big Data Computing”, CRC Press, 2013.

[9] White, T., Hadoop, “The Definitive Guide”, O'Reilly Media, Inc., 2012.

[10] Hadoop, A. Hadoop Documentation, 2013 [cited 2015; Available at: http://hadoop.apache.org/docs/r1.2.1/mapred_tutorial.html#Example%3A+WordCount+v1.0.

[11] Nugent, A., F. Halper, and M. Kaufman, “Big Data for Dummies”, John Wiley & Sons, 2013.

[12] ActivSteps. Practical Data Science. 2013 [cited 2015; Available at: http://www.datascience-labs.com/hive/hiveql-data-definition/.

[13] Services, A.w. Amazon Elastic MapReduce. 2009 [cited 2015; Available at: http://docs.aws.amazon.com/ElasticMapReduce/latest/DeveloperGuide/what-is-hue.html.

[14] Cloudera. Hue Installation Guide. [cited 2015; Available at: http://cloudera.github.io/hue/docs-2.0.1/manual.html#_introduction.

[15] Das, T. and P.M. Kumar, “Big Data Analytics: A Framework for Unstructured Data Analysis”, International Journal of Engineering Science & Technology, 2013. 5(1): p. 153.

[16] Liu, B., “Sentiment Analysis and Opinion Mining. Synthesis”, Lectures on Human Language Technologies, 2012. 5(1): p. 1-167.

[17] Twitter. Documentation. [cited 2015; Available at: https://dev.twitter.com/overview/documentation.

[18] Teach-ict.com. Real time processing. [cited 2015; Available at: http://www.teach-ict.com/as_a2_ict_new/ocr/A2_G063/332_designing_systems/processing_methods/miniweb/pg4.htm#.

[19] TechTarget. Big data buzz gets louder with Apache Hadoop and Hive. 2015 [cited 2015; Available at: http://searchcloudcomputing.techtarget.com/tip/Big-data-buzz-gets-louder-with-Apache-Hadoop-and-Hive.

[20] Pivotal, S. B. Spring XD Guide. 2015 [cited 2015; Available at: http://docs.spring.io/spring-xd/docs/current/reference/html/.

[21] Atlassian. SerDe. [cited 2015; Available at: https://cwiki.apache.org/confluence/display/Hive/SerDe.

[22] Sanchez, G. Mininig Twitter with R. 2012 [cited 2015; Available at: https://sites.google.com/site/miningtwitter/questions/sentiment/analysis.

[23] Sentiments.rar. 2015 [cited 2015; Available at: https://drive.google.com/file/d/0B7wy3b65I3jiUzN4WHBkVXdFejA/edit.

[24] C. Boja, A. Pocovnicu, and L. Batagan, "Distributed Parallel Architecture for" Big Data," Informatica Economica, Vol. 16, pp. 116-127, 2012.

[25] H. Zou, Y. Yu, W. Tang, and H.-W. M. Chen, "Flexanalytics: A Flexible Data Analytics Framework for Big Data Applications with I/O Performance Improvement," Big Data Research, Vol. 1, pp. 4-13, 2014.