DEVELOPING A REAL-TIME DATA ANALYTICS FRAMEWORK FOR TWITTER STREAMING DATA

by

Babak Yadranjiaghdam

December, 2016

Director of Thesis: Nasseh Tabrizi

Major Department: Department of Computer Science

Twitter is an online social networking service with more than 300 million users, generating a huge amount of information every day. Twitter's most important characteristic is its ability for users to tweet about events, situations, feelings, opinions, or even something totally new, in real time. Currently there are different workflows offering real-time data analysis for Twitter, presenting general processing over streaming data. This study will attempt to develop an analytical framework with the ability of in-memory processing to extract and analyze structured and unstructured Twitter data. The proposed framework includes data ingestion and stream processing and data visualization components with the Apache Kafka messaging system that is used to perform data ingestion task. Furthermore, Spark makes it possible to perform sophisticated data processing and machine learning algorithms in real time. We have conducted a case study on tweets about the earthquake in Japan and the reactions of people around the world with analysis on the time and origin of the tweets.

DEVELOPING A REAL-TIME DATA ANALYTICS FRAMEWORK FOR TWITTER STREAMING DATA

A Thesis

Presented To The Faculty of the Department of Computer Science

East Carolina University

In Partial Fulfillment of the Requirements for the Degree

Master of Science in Software Engineering

by

Babak Yadranjiaghdam

December, 2016

DEVELOPING A REAL-TIME DATA ANALYTICS FRAMEWORK FOR TWITTER STREAMING DATA

by

Babak Yadranjiaghdam

APPROVED BY:

DIRECTOR OF THESIS: _____
Nasseh Tabrizi, PhD

COMMITTEE MEMBER: _____
Sergiy Vilkomir, PhD

COMMITTEE MEMBER: _____
Junhua Ding, PhD

CHAIR OF THE DEPARTMENT OF COMPUTER SCIENCE: _____
Venkat Gudivada, PhD

DEAN OF THE GRADUATE SCHOOL: _____
Paul J. Gemperline, PhD

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

Currently Big Data is not a buzzword in a particular context; it is everywhere. With a huge amount of information generated every day and everywhere, it becomes a must for any organization and industry to deal with, as current hardware and software are unable to handle the vast amount of different types of data that is created at such a high speed. Sensors, log data of machines, data storages, public web, social media, business apps, media, archives, and numerous other types of technologies are creating and capturing data continuously in large quantities, offering great opportunities to manipulate and use it in a wide variety of applications. However, we face new technical challenges with respect to managing, organizing, processing, and analyzing this huge amount of data [1] and in how to enable businesses and companies to manipulate knowledge to upgrade the process of decision making and achieving higher performance [2].

Big Data differs from regular data in **a** few characteristics known as the 3 V's: Big Data volume, velocity, and variety. Also, some other characteristics of Big Data recently have been introduced as new V's such as value and veracity.

Volume: The size of digital data in 2011 was estimated at 1.8 Zettabytes, and at the current pace, one should expect to deal with 50 times more information by the year 2020 [3]. The spread of the Internet and the epidemic use of sensors, mobile devices and smartphones contributes remarkably in generating this huge data.

Velocity: Huge amounts of data are produced and Big Data sets are generated rapidly every second. Organizing, accessing, and processing the data as it is collected to be included in the decision making in real-time applications is usually the most important technical challenge [4].

Variety: There are a number of sources for collecting data that are usually in different formats. These sources may be messages, images, and videos posted to social networks, readings from sensors, business transactions, and economic and political news. Beyond the structured data, collected data includes semi-structured or unstructured data of all varieties, such as text, audio, video, web pages, log files, and more [5].

Value: There is usually unknown valuable information in the huge amount of data stored and unused. This characteristic of Big Data refers to recent large volumes of data that have been recorded but not exploited. By using Big Data technologies, value may be extracted from underdeveloped data [5].

Veracity: The important aspect of Big Data is the quality of captured data, which may vary greatly depending on accurate analysis. This is mostly dependent on data analytics methods used in Big Data.

Data scientists try to make use of all achieved data by drawing conclusions from them to potentially extract benefit for society. By analyzing data, knowledge may be extracted from mass amounts of factual information. When this happens, better decisions may be made, and technology may adapt to the trends discovered in that data. There are two big challenges in Big Data analytics: 1) traditional data management methods cannot handle large volumes of data well; and 2) facing unstructured data, which makes the bigger portion of received data [6]. For this reason, NoSQL databases were introduced to handle and utilize the storage and organization of masses of unstructured data. There are various ways to extract information for different types of unstructured data, where each method requires an organization of that data exclusive to that datatype [6]. Some of these methods include text, audio, video, social media, and predictive analytics. With the massive amounts

of data being accumulated from various sources, analysis of Big Data is vastly important for decision making of truly any kind—whether it is for businesses, scientific study, or the improvement of technology as a few examples. Moreover, real-time applications rely upon instantaneous input and fast analysis to arrive at a decision or action within a short and very specific timeline [4]. Originally, data analytics have been performed after storing data on hard disks, which eventually have a fair amount of access latency. Dealing with large amount of structured and unstructured data in real-time makes hard disks undesirable, and as a result, there has been a recent transition from hard disk drive storage to memory storage (RAM). In-memory processing significantly decreases the amount of access latency, which will have a crucial role when real-time analytics is performed.

Many data analytics applications may benefit from, and often even require, the extraction of intelligence from data in as quick and efficient manner. There are numerous ways to accomplish these tasks, as shown in the following sections. Some of the applications of the real-time data analytics are surveillance, environment, health care, business intelligence, marketing, visualization, cybersecurity, and social media. Analyzing data in real-time requires data ingestion and processing of the stream of data before the data storage step [7].

The basic difference between this study and other researches is that the proposed framework is able to not only perform the basic processing tasks, but makes an infrastructure for performing more sophisticated and complicated analytics on the streaming data. Current real-time methodologies use tools and technologies to process Twitter data which are using event processing and one-message-at-a-time analysis. This makes it possible to achieve real-time result, but lacks the ability of doing anything more

than plain processing. Reviewing related works in this field showed that there is a gap of capability of performing more complicated analytical tasks like machine learning algorithms. The proposed framework offers an infrastructure for real-time processing with the ability of extending the analytical capability.

CHAPTER 2: RELATED WORK

Due to fast growth of social networks and their role in the daily life of millions of people around the world, the amount of generated data in these media is increasing exponentially. As a result, more and more people tend to interact via these networks and share their opinions. There is usually unknown valuable information hidden within this data. By examining what is shared by the majority of people, their tastes, their opinions, and their inclinations may be extracted. Additionally, trends in a political situation or in commercial products can be identified. There are many other areas in which social media may give us a good insight about art, sporting events, health, and many other issues people deal with on a daily basis. The common point among all of these is that these streams of data usually are related to a specific time, location, and situation. So real-time applications rely upon instantaneous input and fast analysis to arrive at a decision or action within a short and very specific time line. In many cases, if a decision cannot be made within that timeline, it becomes obsolete [4].

Dealing with social media data, including many different data types such as text messages, photos, and videos and is arriving in a large volume in every second, needs a proper framework which does not rely upon storing data on hard disks and is able to process data in memory, as it arrives [8].

There are many studies conducted in the field of real-time data analytics. Each of these makes some contribution to a specific category in daily life and uses different methodologies. Some of these areas of application have a high rate of sensitivity to react to factual data. The stock market is a tangible example of areas, which always have had a

heavy reliance upon fast and accurate analysis. A flying object in an unsafe situation which positions and find routes based on various data sensors is another example of necessity of real-time processing based on factual information [9]. Social media data on the other hand is usually the sort of data based upon opinions and feelings. In spite of this, there are still lots of useful hidden information, which may be extracted from this type of data. Analyzing posts on sites such as Facebook and Twitter may prove quite useful for drawing conclusions and making predictions about activities that occur in specific areas of the world at certain times [10]. Social media platforms can be quite informative through a crowdsourcing standpoint. Nguyen and Jung [11] offer a method of event detection through the behavioral analysis of Twitter users. By utilizing real-time data analytics on big social data, important events, even emergencies, may be predicted and detected. An architecture [12] was developed for analyzing social media text by filtering keywords, languages, and other informative aspects of large data set of tweets. This organized data is then used to process and draw conclusions.

Twitter data has a great potential for extracting trends and sensing communal feelings. Authors [13] presented a methodology for finding patterns related to the health events, where they collected and filtered data of five different diseases from three Australian cities. For this purpose, they offered a text analysis based upon classifying the list of words. They also used a scoring system to extract the relation between tweets and diseases. There are other studies in stream computing in healthcare applications [14], which focuses upon different sources of healthcare data varying from biomedical images and EHRs to social media data. Wachowicz et al. [15] developed a workflow for data ingestion and data management of Twitter streaming data, where they retrieved space-time activities

from geotagged tweets and stored them in a single cluster of MongoDB. Other studies [16] search for trends in Twitter posts using complex event-processing (CEP) that may determine important events from a large influx of updates and events and may act upon them in real-time. While processing and analyzing social media data may be very useful in a number of ways, Twitter services themselves make use of real-time data analytics query suggestion and spelling correction [17].

For data management, many of the methodologies use Apache Storm [19] (that is explained completely in chapter 3). It processes an event at a time and provides general primitives to do real-time computation. It also simplifies working with queues and workers while offering a scalable and fault-tolerant basis. In so doing, methodologies that use Storm for their data processing usually are able to perform general computational tasks, and if they want to use more complicated processes, they often do it after storing data on a database or passing the results to another real-time data analytics tool. In this situation, Storm has the role of data filtering and regular processing in real-time. However, using event-processing decrease its latency to sub-second order (almost no latency) but it is not able to use online machine learning [14, 15, 20].

Spark is a Big Data processing framework built to offer speed beside sophisticated analytics. Spark runs streaming computation as a series of very small, deterministic batch jobs [21]. The size of these batches may be as low as half a second with a latency of about one second. Although the latency in Spark is a little bit more than event processing frameworks, but in many cases this may be considered to be real-time [22]. This is why the presented framework uses Spark's capabilities of in-memory processing and its abilities of complex analytics as opposed to Storm's no-latency feature.

There are studies showing the power of social media in monitoring the impacts of earthquakes. Authors [23] analyze the ability of Twitter in contribution of information in terms of location and time of the earthquake happened on the East Coast of the United States on August 23, 2011. They compared results with the data gathered by U.S. Geological Survey and show that social media data may complement other sources of data and may help to improve our understanding of this type of events.

This is not the ultimate border of social media application. Earle [24] considers Twitter to be first-hand accounts of earthquake which by analyzing their content and geographical location, within a very short time may offer a supplement for instrument-based estimates of the location and magnitude of earthquakes. Authors [25] constructed an earthquake reporting system, which is using Twitter as a social sensor for detecting an event in real-time. They were able to detect almost 96% of earthquakes with the seismic intensity scales of higher than three only by monitoring the tweets. Their system was able to detect an earthquake and send emails to registered users, much faster than broadcasting the event.

The U.S. Geological Survey (USGS) is investigating how Twitter may help in augmenting earthquake response products and may improve the delivery of hazard information. The authors [26] show that this fast way of monitoring the earthquakes in 75% of the samples may detect the event within two minutes of origin time, which is much faster than seismographic detections in poorly instrumented areas.

CHAPTER 3: SURVEY OF REAL-TIME BIG DATA APPLICATION AND TOOLS

Users have taken advantage of Big Data technology to offer new services. In this section, we review the dominant tools and technologies that are being used in Big Data and real-time analytics. Although Hadoop is used successfully as a base for data storage and distributed processing, but, for stream processing, Spark is the most dominant tool which enables analyzing data in real-time. Also, for real-time data analytics there will be a need for data ingestion tools like Kafka, Storm, and Flume to import data in a way that is ready to be analyzed on the clusters.

3.1 Hadoop

The Apache Hadoop framework allows for the distribution processing of large data sets across clusters of computers. This happens while it uses simple programming models. Instead of using a single server it distributes the tasks over many machines, each of them having their own storage and processing units with no dependency on hardware to achieve high-availability [42]. The Hadoop library is designed to detect and handle failures at the software layer, so it does not need to rely upon high performance hardware [27], and thus, a highly available service on a cluster of computers, is delivered by it. Hadoop provides a framework in the field of Big Data analytics, which is a base for many of the other tools. It is a distributed processing framework based upon Java, effective in data-intensive analytics [28]. As a sample, all of the methods proposed in [29] use Apache Hadoop for data processing, analytics, and storage; specifically, they make use of Hadoop for the processing and analysis of millions of images but also utilize different storage systems that are searchable.

Many works in recent papers make use of Hadoop's specific projects, such a Map Reduce, for processing and storage. There is a fact about Hadoop MapReduce, however, is limited to batch processing of one job at a time. Additionally, it offers parallel computing that contributes to high performance and efficiency for large data analytics projects [30]. At the same time Hadoop MapReduce persists back to the disk after a map action or a reduce action thereby Hadoop MapReduce faces an access latency. The authors [31] propose the implementation of real-time analytics as a provided service. The architecture is made up of three service components: backend training system, service wrapper for easy machine learning accessibility, and service user interfaces to make implementation of real-time analytics simple for non-programmers. In the situation of behavior analytics, all of the relevant data that may influence student behavior is not always readily available. However, with the data that the school may collect, Hadoop, along with several of its included resources (Hive, Pig, Hadoop Distributed File System, and MapReduce), was the tool that authors [32] chose to collect, analyze, and store their data. While Batarseh and Latif [33] are sure to make use of several health care specific data analytics tools, Hadoop is implemented in their framework to handle storage and querying of health care data received.

There are some sources that do not explicitly use Hadoop for their data analytics, but some use it to evaluate the performances of their systems [11]. Hadoop's MapReduce feature is appealing to multiple researchers aiming to efficiently process large datasets and to attain meaningful information from them [12, 34]. Many argue that an optimal solution to a Big Data problem may be achieved through the use of Hadoop and its several provided methods [34] by performing a couple different experiments on different Big Data sets. The

creators of Facebook Messaging [35], utilize Hadoop for their application because of its elasticity, fault isolation, low latency, and consistency semantics. Other researchers, however, make use of only part Hadoop's functionality [36] by applying MapReduce for the processing of data, but the cloud for data storage. However, authors [37] argue that while Hadoop is well suited for experimentation in data analytics, it is not ideal when aiming for real-time processing. In fact, the architects behind Twitter [17] have attempted to supply query suggestion and spell checking services with Hadoop's processing but have changed their method because the access latency was much higher. Introduction of the methodology [38] for real-time image retrieval with in-memory vocabulary tree makes use of MapReduce for the training and retrieval of images in real-time.

3.2 Spark

Spark is a framework for parallel processing of Big Data. Spark is designed to use Hadoop MapReduce with some modifications that enable it to perform more efficiently. Apache Spark has its own streaming API and independent processes for continuous micro-batch processing across intervals with varying, but short time duration. Spark runs up to 100 times faster than Hadoop in certain circumstances; however, it still uses Hadoop distributed file system. This is the reason why most of the Big Data projects install Apache Spark on Hadoop so that the advanced analytical applications may be run on Spark by using Hadoop distributed file system. So, one may consider Spark as an extension of Apache Hadoop, which has some features for real-time analytics and is supportive of applications such as machine learning, stream processing, and graph computation [43]. Authors [31] have implemented Spark for real-time data analytics as a service. It is able to

support both stream and batch processing while Hadoop is made mostly for batch processing. Spark provides many real-time processing and evaluation options that Hadoop alone cannot. Therefore, to manage the data for their architecture, they utilize Spark specifically. Though in [39], authors are making use of a graph database, Neo4J, to store datasets. Use of Spark in the graph processing system being used allows analysis of the waste amount of data possible.

The research [40] on distributed computing engines shows that Spark has consistent scalability for large datasets as a new Big Data analytics platform that supports more than map/reduce parallel execution mode with good scalability and fault tolerance. They [41] show that Spark is scalable to process seismic data with its in-memory computation and data locality features. They have used a few typical seismic data processing algorithms to study the performance and productivity of Spark.

3.3 Storm

Storm is another real-time computation system. It is a task parallel distributed computing system, which may process unlimited streams of importing data. Storm utilizes Zookeeper, a service for maintaining configuration information and for providing distributed coordination, instead of running on Hadoop clusters. The main use of Storm is real-time analytics. Many of the explored resources make use of Storm with their new contributions to real-time data analytics [44]. Storm, unlike Hadoop alone, can continue to analyze data as it arrives (dissimilar to batch processing). Additionally, Storm is a CEP system that has the ability to detect important event occurrences. For that reason, Storm is

the processing system that Jones [16] utilizes to detect crucial events through the processing of Twitter feeds.

3.4 Flume

Data ingestion tools have a very important role in real-time analytics. Flume is one of the tools that offer a distributed and available service for importing data. Collecting and bringing in large amount of data based on streaming data flows, makes it possible for Big Data frameworks to ingest data in a way that makes it easy for processing tools to reach data. Flume is one of the data processing frameworks that has the ability to be applied to real-time data analytics acts similarly to that of Storm [45]. Makeshwar et al. [46] proposed a framework to receive and store huge data from a sensor network and to analyze the received data. They explored the suitability of Apache Flume and Apache Mahout to deliver high performance computational scalability of Hadoop Distributed File System.

3.5 Kafka

The other powerful tool for data ingestion is Kafka. It is a platform for distributed streaming and a message brokering system which provides a high-throughput data feeds. Another characteristic of Kafka, which makes it different, is a low-latency platform for real-time processing. It is, in fact, a scalable message queue which by distributing transaction log, makes it highly useful for infrastructures to process streaming data [6]. Others including [47] have used multiple streams of messages that are generated from Apache Kafka's producers and processed at Storm, then stored in a distributed storage NoSQL Cassandra system. Their framework may be applied for many real-time analytics as well as prediction and recommendation systems in healthcare.

There are other tools being used in data analytics; S4 is a popular framework for in-memory stream processing. It is able to handle data continuously as it arrives without terminating [48]. The sources discussed in this paper make use of common tools, but each framework utilizes the tools in various ways. There are several ways to conduct data analytics in real-time. A new data analytics methodology, FAST, proposes a much faster way to search Big Data for relevant files using semantic correlation [29]. In their article [37], authors introduced Radoop as a new method for data analytics that is meant to be a hybrid Hadoop and RapidMiner. It makes it possible to analyze data beyond the boundaries of main memory. Medusa and Mars are applications specifically for visualization of data and cluster analysis. According to He [49], these tools are useful for real-time data analytics visualization specifically with GPGPUs. Others [50] have adopted WebGL to perform GPU-based data processing and computation.

3.6 Comparing Processing Tools

Hadoop, Spark and Storm are all open source processing systems and may be used for Big Data analytics. They offer fault tolerance and scalability while needing a simple implementation methodology. Hadoop, Spark, and Storm are implemented in JVM based programming languages- Java, Scala and Clojure. The basic difference between these processing tools may be identified in two categories: 1) the processing models they are using; and 2) their performance.

- Hadoop MapReduce is very suitable for batch processing. As previously discussed, Big Data applications require platforms to perform real-time tasks. Apache Spark is designed to do more than simple data processing and it is able to run machine

learning libraries and process graphs. Apache Spark is capable of performing both batch processing and real-time processing; it makes it possible to use a single platform for everything rather than splitting the tasks on different systems. Micro-batching is a special kind of batch processing wherein the batch size is much smaller. Windowing is much easier with micro-batching, due to its stateful computation of data. Storm is a complete stream processing engine that supports event processing whereas Spark is a batch processing engine that micro-batches.

- Spark processes in-memory data whereas Hadoop MapReduce goes back to the disk after a map action or a reduce action; thereby, Hadoop MapReduce has much more latency compared to Spark in this aspect. Spark requires huge memory, as it loads the process into the memory and stores it for caching. However, if Spark has the ability to run on top of YARN with various other resources demanding services, but the performance will be much reduced. In the case of Hadoop MapReduce, the process is killed when the job is completed, making it possible to run along with other resource demanding services with a very little difference in performance. Spark and Storm both provide fault tolerance and scalability but differ in the processing architecture. Spark streams events in small batches that come in short time windows before it processes them, whereas Storm processes one event at a time. Thus, Spark in total, has a latency of a few seconds, whereas Storm processes an event only with millisecond order latency. Spark has good performance on clusters when the data is in the memory, whereas Hadoop can only perform well when data is in its distributed file system on the hard disks.

Table I categorizes different studies based on the technology they used and area of application of Big Data analytics of each of them.

Table 1: Technologies used in different fields of Big Data

	Tools					
	Hadoop	**Spark**	**Storm**	**Kafka**	**Flume**	**Other**
Surveillance	[32], [29]					[51]
Visualization	[52]					[52], [50]
Environment	[30]	[39]				
Social Media	[35], [7], [17], [11], [12]	[48]	[7], [16], [48]	[6]	[45]	[48]
Health Care	[33], [36]		[47]	[47]		
Business Intelligence	[28], [34], [54]					[53]
Marketing	[55]	[55]				[56]
Cybersecurity	[18]					
Internet of Things	[46]		[44]		[46]	
General	[38], [37], [31]	[31]				[38]

CHAPTER 4: REAL-TIME DATA ANALYTICS FRAMEWORK

The purpose of this study is to develop a framework for real-time data analytics of Twitter data. This framework has some characteristics which distinguish it from traditional data analytics approaches. The main idea here is that there is a need for methods to analyze thousands of tweets coming each second, in a short time. Also, the framework should be independent of imported data volume; this is important because the volume of tweets is growing at a noticeable rate. Figure 1 shows a schematic of a scalable stream processing. The concept here is to collect event streams by different nodes and let multiple processing nodes to analyze data in parallel. So, the challenge here is how to manage streaming data and how to analyze it over the clusters.

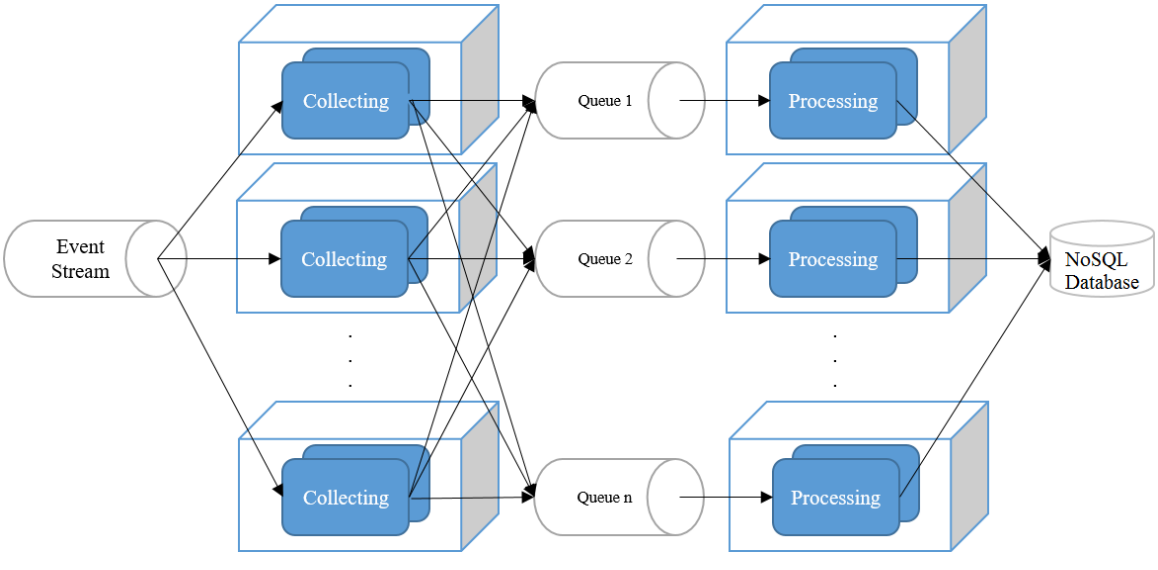

Figure 1: A Schematic Scalable Stream Processing

Data processing workflow usually connects computing resources to automate a sequence of tasks by processing large volumes of data. Different resources are connected

for automating different tasks. In the case of streaming data processing, a scalable and distributed platform is required for combining large volumes of historic and streaming data at the same time. The presented framework consists of three sections: 1) data ingestion; 2) data processing; and 3) data visualization. The data ingestion section, connects directly to Twitter streaming API and in a scalable manner import data to the framework. The data processing section with the ability of streaming processing over cluster accessses distributed imported data, analyzes data in-memory, and performs sophisticated processing tasks on data, and finally sends the results to be monitored. Figure 2 shows the real-time data analytics framework and its different components.

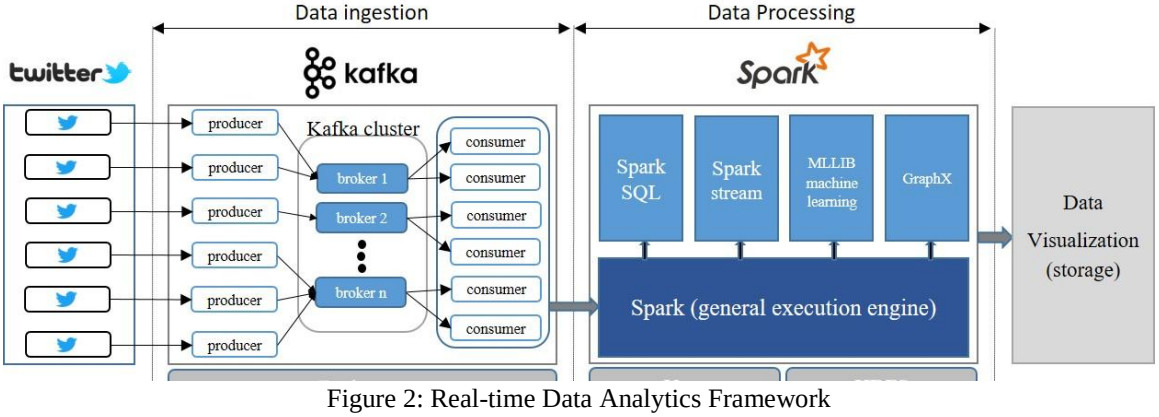

Figure 2: Real-time Data Analytics Framework

4.1 Data Ingestion

Apache Kafka is a distributed streaming platform that uses publish-subscribe messaging and is developed to be a distributed, partitioned, replicated service. Our framework uses this message brokering system. Data ingestion clusters typically consist of multiple brokers. To balance the incoming load, Topics are defined and each of these Topics is split into multiple partitions. In this system, each broker stores one or more of

those partitions. Another important aspect of this section of our framework should handle is the ability to accept multiple formats, varying from text, image, video and other formats. This is a basic need for Big Data systems to deal with unstructured data.

Kafka is suitable for building real-time streaming data routes that reliably pass data to systems or applications. Kafka does this by running on a cluster of servers. The Kafka cluster stores and categorize streams of records in Topics, while these records consists of key, value, and timestamp. Two main modules of Kafka are:

- The Producer: allows publishing a stream of records to Topics.
- The Consumer: allows subscribing to Topics.

The Figure 2 shows the role of each of this in the data ingestion section. In fact, Topics are the core abstraction which Kafka provides for a stream of records. A Topic is a category name to which records are published. Topics in Kafka are multi-subscriber and may have zero, one, or many consumers who may access to the data written to it. Partitions are sequence of records which are ordered, immutable, and may be continually appended to a commit log. This architecture allows Kafka's performance to be constant with respect to data size.

The Kafka cluster always retains all published records, without consideration of their consumption. However, in the proposed framework there is no need to store imported data in database, and the whole process will take place in memory to avoid access latency of hard disks.

Producers publish data to the desired Topics. The producer chooses which record to assign to which partition in the Topic; this may help to balance load. Each record

published to a Topic is accessed by one consumer. Consumers may be in separate processes or on separate machines.

In Kafka, a stream processor is the engine that takes continous streams of data from input Topics, and creates continual streams of data to output Topics. Kafka provides an integrated Streams API that allows building applications that do some processing that compute aggregations of streams. This may help execute tasks such as handling out-of-order data. It uses the producer and consumer for input, manipulate Kafka for stateful storage. The stateful computation is useful when we are using processing tools that utilize stateful computing.

4.2 Streaming Data Processing

Apache Spark is a very powerful engine to perform fast and large-scale in-memory data processing. In our framework, the data processing task uses Spark as shown in Figure 2. The core is the distributed execution engine and Additional libraries, built on the core, allow different workloads for streaming, SQL, and machine learning. Spark is designed for data science uses, which are more sophisticated processings. For example, machine learning algorithms are often iterative, and Spark's ability to cache the dataset in memory helps enormously speeds up iterative tasks. It consists of Spark Core and a set of libraries. Spark Streaming is an extension of the core Spark API that enables scalable, high-throughput, fault-tolerant stream processing of live data streams. Data may be ingested from many sources like Kafka, Flume, or TCP sockets, and may be processed using complex algorithms expressed with high-level functions such as map, reduce, join, and window. Finally, processed data may be pushed out to filesystems, databases, and live

dashboards. In fact, one may apply Spark's machine learning and graph processing algorithms on data streams.

Internally, it works as follows: Spark Streaming receives live input data streams and divides the data into batches, which are then processed by the Spark engine to generate the final stream of results in batches.

Spark Streaming provides a high-level abstraction called discretized stream or DStream, which represents a continuous stream of data. DStreams may be created either from input data streams from sources such as Kafka, Flume, and Kinesis, or by applying high-level operations on other DStreams. Internally, a DStream is represented as a sequence of RDDs. Each RDD in the sequence may be considered a "micro batch" of input data, therefore Spark Streaming performs batch processing on a continuous basis.

In addition to other Spark API libraries (such as Spark SQL, MLlib; machine learning, GraphX), Spark provides another major library called Spark Streaming. This library allows processing data streams (a continuous sequence of records) in near real time. There are two common approaches for stream processing: a) process each record individually as soon as it is arrived; or b) combine a set of records in mini-batches. Here, mini-batches may be created either by time or number of records in a batch [39]. Spark Streaming receives data from an input source such as file-based and network-based sources.

Spark can process Kafka using Receivers, but Spark also may be a direct consumer client of Kafka instead of using Receivers. The direct approach ensures Exactly Once processing of the Kafka data stream messages. Full end-to-end Exactly Once processing may be achieved provided that Spark's output processing is implemented as Exactly-Once.

There are two types of operations on DStreams: transformations and output operations. Spark application processes the DStream RDDs using Spark transformations such as map, reduce, and join, which create new RDDs. Any operation applied on a DStream translates to operations on the underlying RDDs, which in turn, applies the transformation to the elements of the RDD.

4.3 Data Visualization and Storage

Here the results as well as data streams may have to be stored or visualized. The storage should be on a NoSQL database since the tweets are in different formats, ranging from text to images to videos. Data stored in database may be used later for historical data analysis. However, the value of this type of data usually belongs to the current situation and may be much different in another time and circumstance.

4.4 Software Architecture of Real-Time Data Analytics Framework

The real-time data analytics framework is built based upon Kafka and Spark tools. This system consists of different software components, demonstrated in Figure 3, where the Create Producer module does the task of filtering incoming data and creating Kafka

producers. In this module, Topics are defined and the incoming data is sent to Kafka Brokers. This part has been developed in Java.

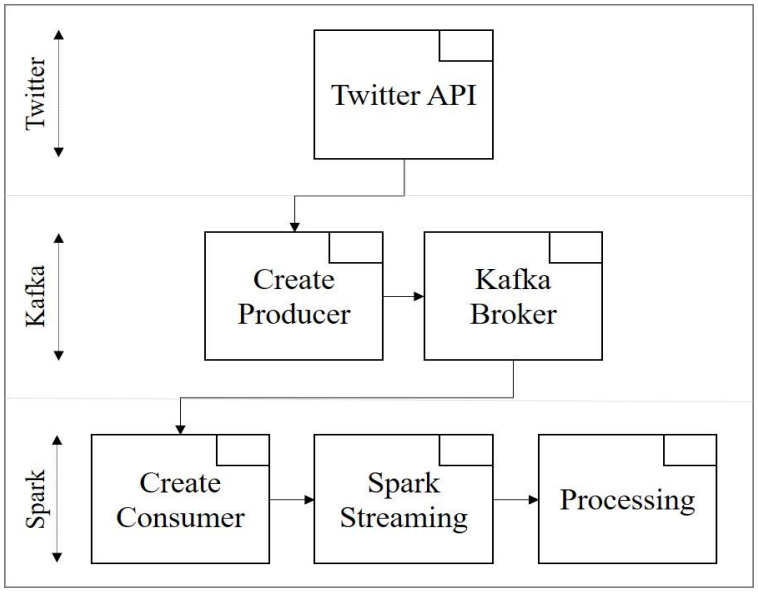

Figure 3: Software Architecture of Real-time Data Analytics Framework

Kafka Brokers distribute data and “Create Consumer” module which has been developed over Spark using Scala language reaches to distributed data and creates consumers to import data into Spark Streaming. When data is sent to this section, the Processing module starts the analytical tasks over imported data.

CHAPTER 5: CASE STUDY

Big Data analytics require a scalable hardware infrastructure with parallel processing capability. This system should have enough memory, bandwidth, and throughput, and be able to run multiple tasks simultaneously, and perform parallel processing of advanced analytics algorithms in matter of seconds. Since the main concept of Big Data computing is distributed processing, the framework is implemented over a cluster of servers. We have arranged a cluster of 16 servers which provide us with a powerful hardware base for Big data analytics tasks. Four of these servers act as administrative nodes and 12 servers work as worker nodes. Each of these 16 Servers has two Intel(R) Xeon(R) quad core CPU 5620 2.40 GHz processors, meaning there are eight real cores or 16 virtual cores on each server. These servers are equipped with 16 GB DDR3 RAM and a 1 TB hard disk. Operating system, we are using is a Linux Ubuntu server 14.04 64-bit. The switch is used is Juniper EX4200 which is a high-performance, low-latency one, providing 1 Gigabit Ethernet (GbE) access environment.

The infrastructure runs different software on administrative and worker nodes to provide the basis for Big Data analytics. The administrative nodes run HDFS primary NameNode, HDFS secondary NameNode, YARN ResourceManager, Kafka server, and Zookeeper server. The worker nodes run HDFS DataNode and YARN NodeManager. As previously described, in this framework, Kafka has the role of data ingestion and Spark provides a powerful basis for data analytics.

Since Twitter is social media with the characteristics of allowing people to react to an event just as it is occurring, we decided to monitor an event that is occurring at the

moment and it affects the whole world, or at least is from the point of interest of people in different countries. An earthquake happened in Japan on November 22nd, 2016. CNN reported: “A 6.9-magnitude earthquake struck off Japan's Honshu Island on Tuesday, triggering tsunami waves and bringing back traumatic memories for locals of the devastating 2011 Fukushima disaster.” The center of this earthquake was pretty close to the one in 2011. Figure 4 shows the 2011 earthquake center and the areas that were affected

Figure 4: 2011 Earthquake Center and the Areas Affected in Japan

in Japan.

Figure 5 shows that the earthquake happened in November 2016 and caused a tsunami alert and fear in Japan, Southeast Asia and even all around the entire world. The earthquake happened almost in the same place and the tsunami waves created an atmosphere of fear among the people living around the area.

Figure 5: Earthquake Near Japan in November 2016 Caused Tsunami Alert

The earthquake in 2011 triggered powerful tsunami waves that reached heights of up to 133 ft. Based upon the Japanese National Police Agency report 15,894 died at that time. The tsunami caused nuclear accidents, in the Fukushima Nuclear Power Plant Complex, threatening the lives of millions of people as well. All of these events have left a very bad memory and have made the people around the world sensitive to tsunami alerts.

As this news was broadcasted we started watching tweets. The flow of tweets was ingested in to our framework and then filtered, processed, and visualized. For this purpose, we started watching tweets including “Tsunami,” “Japan earthquake,” and “Fukushima.” In our framework Kafka connected to Twitter streaming API and for this brokered the incoming stream on the cluster to be ready to be analyzed. The filtering step happened here and the flow of tweets were categorized based upon their content.

In the next step Spark connected to distributed messages by Kafka and using the Spark Streaming process of analyzing data taking place. This step included analyzing the tweets based on their time, location of tweets, and the time zone from which they were

tweeted. This information was processing in memory and so we have a real-time processing over huge amounts of data coming into the system.

We have visualized the processed data on the world map in consecutive eight hours. These maps show that people all around the world reacted to this news. It shows that the most tweets about this earthquake happened in Southeastern Asia, the place most affected by the past tsunami. We analyzed a period of eight hours (11:00 PM, November 22nd – 7:00 AM, November 23rd) which was during day time in east Asia and night time in North America. The amount of tweeting changed during different hours in different countries. Figure 6 demonstrates the tweets about the earthquake and a possible tsunami in Japan right after the earthquake happened and the tweet scale for each of the countries in the world. This figure shows how our framework was able to import data, filter it, and analyze it in real time.

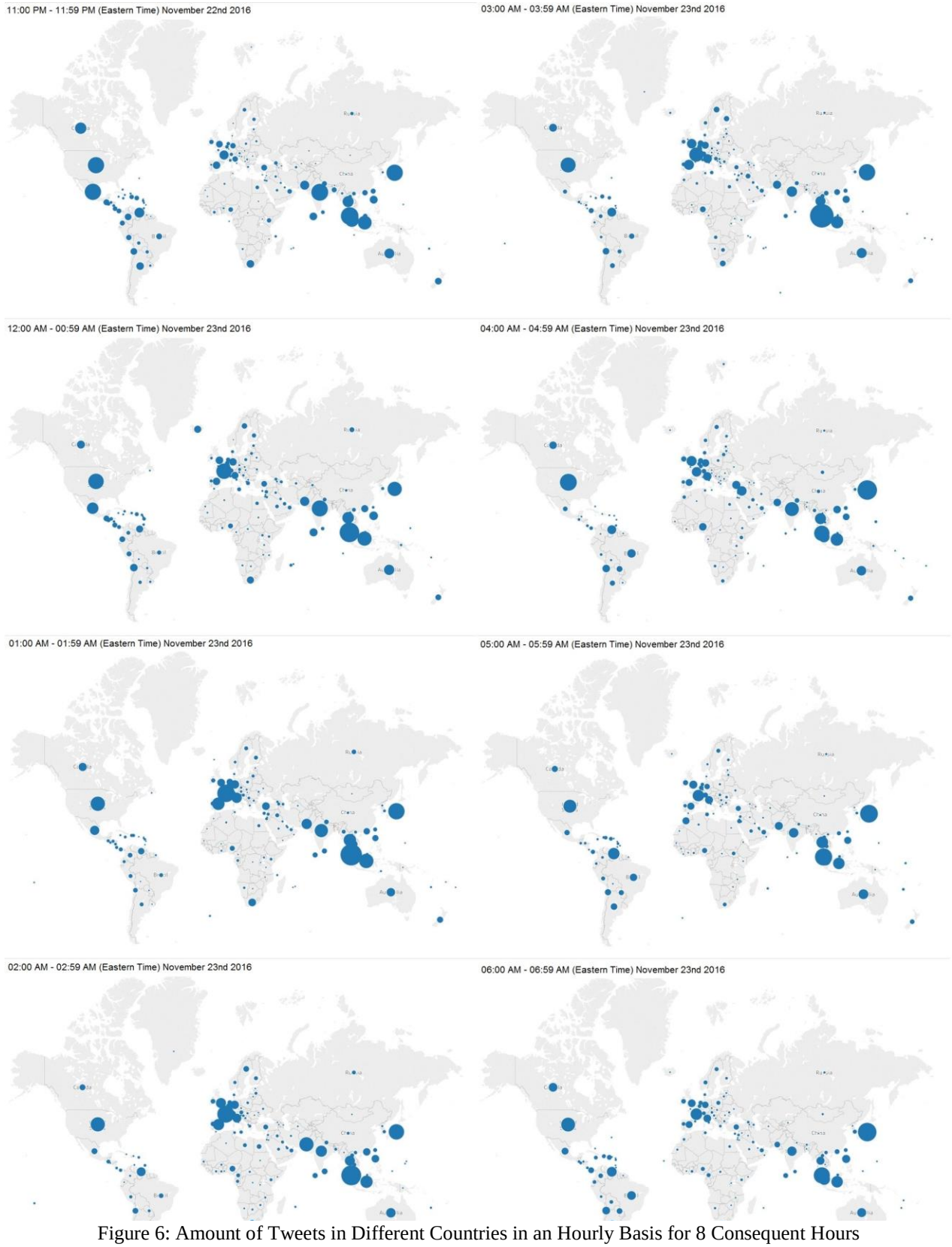

Figure 6: Amount of Tweets in Different Countries in an Hourly Basis for 8 Consequent Hours

We started receiving tweets only hours after the occurrence of the earthquake. During that eight hours we received over 50K tweets from all around the world about this incident. This means that we had more than 6,000 tweets per hour and more than 100 tweets per minute. This inordinate number of tweets shows the importance of social media for its users as a place to share their concerns and to even maybe use it as a communication tool and alerting system. Figure 7 shows the 20 countries with the most tweets about this issue.

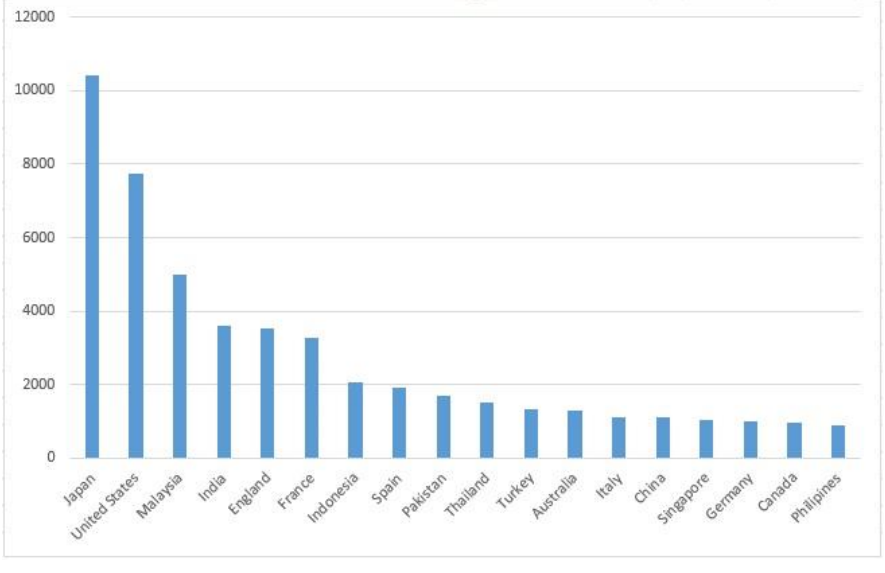

Figure 7: Countries with Most Tweets About Japan Earthquake.

This shows that Japan was the country most directly affected by this earthquake and a possible tsunami produced the most tweets. After Japan the countries in Southeast Asia had the most tweets. Also we have to consider that a country like USA has the highest rate of the tweeter users in the world, so it is not strange to see it in second place. Figure 8 demonstrates the countries with the highest Twitter users.

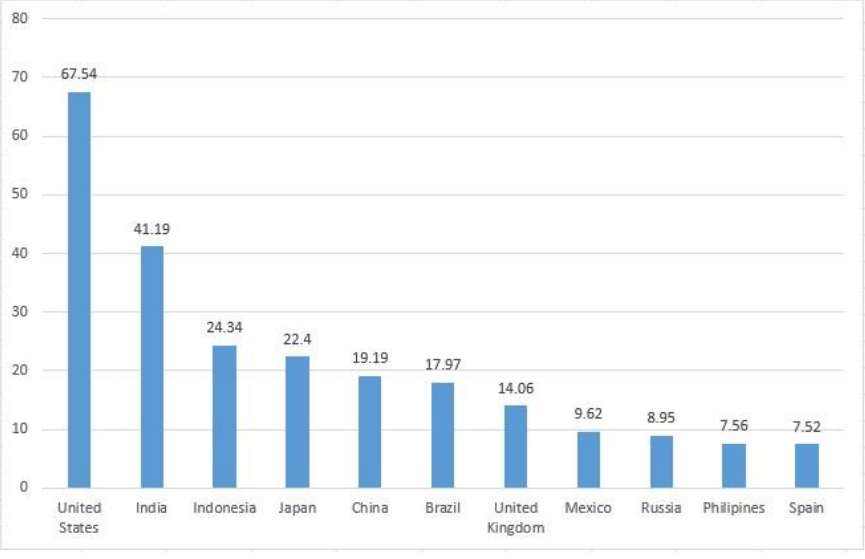

Figure 8: Countries with the Highest Twitter Users. © Statista 2016

The comparison between Figure 7 and Figure 8 shows that except the countries that were directly affected or located in the same region (Malaysia), countries with the highest Twitter users that tweeted the most. The United States, the leading country in Twitter users with more than 67 million, ranked second after Japan, even though the sampling period was during night time. This simply shows that there is a relation between the number of social media users and the number of reactions about a relatively important issue. India, Indonesia and United Kingdom are samples examples of this relationship.

Another interesting statistic here is the number of countries in the world that have reacted to this earthquake and its tsunami hazard. Almost all of the countries in the world except for a few in central Africa have at least shared at least a few tweets. Figure 9 shows how many tweets have been sent from different countries.

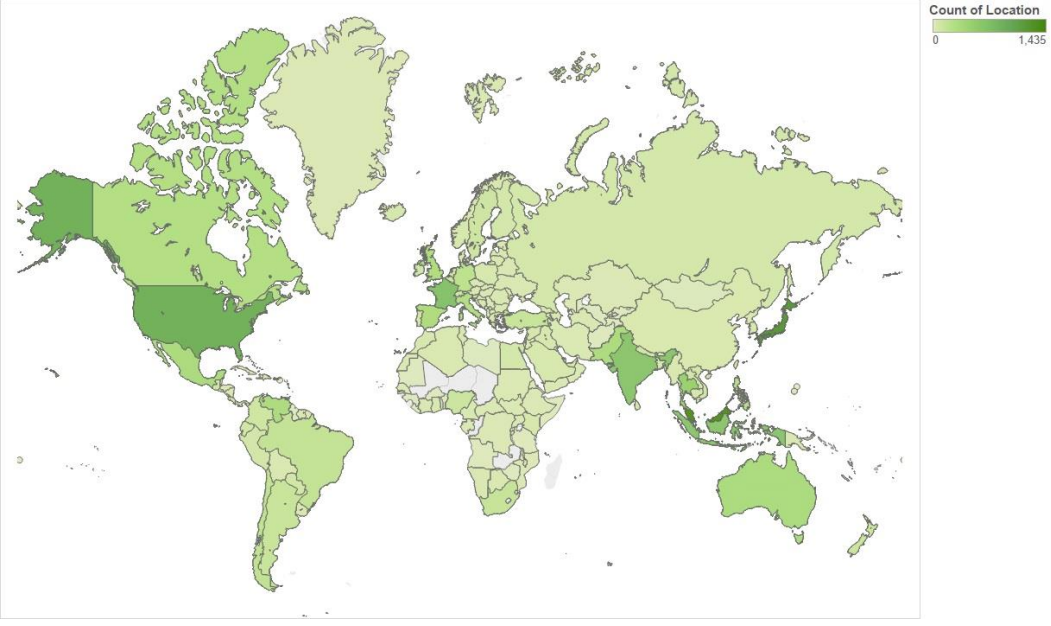

Figure 9: Amount of Tweets from Each Country, the Darker the Color, the More Tweets Are

5.1 Validating The Results

The social media data is not a 100% accurate data, since people usually reflect their opinions and feelings about an ongoing issue. It is almost impossible to rely on one or even few tweets to find out whether something is right or wrong; yet, dealing with a large amount of data Twitter and other social media networks may be a source to follow an event or detect some trends. In our case we had to validate our framework for two important aspects. The first one is whether or not it is able to detect tweets from different sources and locations. The second is checking our framework in terms of its capabilities of real-time processing capabilities. We conducted several test cases for this reason. First, a tweet posted with special hashtags such as #ilabbigdatainfrastructure (Innovation lab Big Data Infrastructure). The test results proved that we are able to detect any tweet with any content from different

locations. For the second part of testing we examined how long it took to retrieve the processing results after posting tweets. We are able to detect the details of tweets and their exact time to posting down to a manner of seconds. The total time for this process consists of the network connections' latency, ingestion time, processing time, and visualization time. Our estimate of our real-time framework was to import and process data in about one second; however, the Internet speed and network connectors could affect the total time. For this test we read data from Twitter every five seconds and started tweeting and retrieving processed data. Results show that our system was able to get the tweets in this range. This shows that our framework was successfully analyzed data in real-time (or near real-time). The last test was to tweet from different locations and various times with the sample hashtags and running spatial and temporal analyses. The result confirmed that our framework is able to accurately offer location and time based processing.

CHAPTER 6: CONCLUSION

In this study we have proposed a framework for real-time analysis of Twitter data. This framework is designed to collect, filter, and analyze streams of data and gives us a chance to sense what is popular during a specific time and condition. The framework consists of three main steps: data ingestion, stream processing, and data visualization. Data ingestion is done by Kafka, a powerful message brokering system to import tweets, and to distribute it based on Topics that it defines, and to make it available over consumers' nodes to be used by analytical tools. Apache Spark is used to access these consumers directly and analyze data by Spark Streaming. This allows not only general processing tasks but more sophisticated and high-level data analytics and machine learning algorithms. The case study that was conducted regarding the earthquake in Japan in November 2016, watched during an eight-hour span, and analyzed the imported data in terms of the origin of the tweets and the number of tweets within the few hours of news broadcasting. This provided us with very good insight about how people react to tragic or dangerous events in specific locations.

This study may and should be investigated more and more toward sentiment analytics or conducting intangible poll in election time. Another very interesting area of study could be medical and healthcare issues. This may help to find out about the spread of a disease even before reported samples of it become of the interest of the officials.

References

[1] Snijders, C.; Matzat, U.; Reips, U.-D. (2012). "'Big Data': Big gaps of knowledge in the field of Internet". International Journal of Internet Science. 7: 1–5.

[2] A. McAfee and E. Brynjolfsson. "Big Data: the management revolution." Harvard business review, Vol. 90, No. 10, pp. 60-68, 2012.

[3] Bakshi, Kapil, "Considerations for Big Data: Architecture and approach", Aerospace Conference, IEEE, 2012, pp. 1-7.

[4] N. Mohamed, J. Al-jaroodi, Real-Time Big Data Analytics: Applications and Challenges. International Conference on High Performance Computing & Simulation (HPCS), 2014.

[5] P. Gupta. N, Tyagi. An Approach Towards Big Data-A Review. International Conference on Computing, Communication and Automation (ICCCA2015).

[6] Datadog Engineering Blog. Monitoring Kafka performance metrics. 23 May 2016.

[7] S. Cha and M. Wachowicz. Developing a real-time data analytics framework using Hadoop. 2015 IEEE International Congress on Big Data, pages 657–660, June 2015.

[8] B. Yadranjiaghdam, N. Pool, N. Tabrizi, “A Survey on Real-time Big Data Analytics: Applications and Tools,” in progress of International Conference on Computational Science and Computational Intelligence, 2016.

[9] H. Hedayati, N. Tabrizi, “MRSL: Autonomous Neural Network-Based 3-D Positioning System," 2015 International Conference on Computational Science and Computational Intelligence, Pages: 170 - 174, doi: 10.1109/CSCI.2015.88.

[10] A. Bifet, “Mining Big Data in real time,” Informatica, 37(1), 2013, Pages 15 - 20.

[11] D. T. Nguyen and J. E. Jung. Real-time event detection for online behavioral analysis of big social data. Future Generation Computer Systems, 2016.

[12] D. Preotiuc-Pietro, S. Samangooei, T. Cohn, N. Gibbins, and M. Niranjan. Trendminer: An architecture for real time analysis of social media text. Proceedings of the workshop on real-time analysis and mining of social streams, 2012.

[13] J. Zaldumbide, R. O. Sinnott, "Identification and Validation of Real-Time Health Events through Social Media," 2015 IEEE International Conference on Data Science and Data Intensive Systems, Pages 9 – 16, doi 10.1109/DSDIS.2015.27.

[14] V. Ta, C. Liu, G.W. Nkabinde, "Big Data Stream Computing in Healthcare Real-Time Analytics", 2016, IEEE International Conference on Cloud Computing and Big Data Analysis, Pages: 37 - 42, doi: 10.1109/ICCCBDA.2016.7529531.

[15] M. Wachowicz, M.D. Artega, S. Cha, and Y. Bourgeois, "Developing a streaming data processing workflow for querying space–time activities from geotagged tweets" Computers, Environment and Urban Systems Journal. 2015.

[16] M. T. Jones. Process real-time Big Data with twitter Storm. IBM Technical Library, 2013.

[17] G. Mishne, J. Dalton, Z. Li, A. Sharma, and J. Lin. Fast data in the era of Big Data: Twitter's real-time related query suggestion architecture. In Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data, pages 1147–1158. ACM, 2013.

[18] T. Mahmood, U. Afzal. Security Analytics: Big Data analytics for cybersecurity. In 2nd National conference on Information Assurance(NCIA), 2013.

[19] A. Toshniwal, S. Taneja, A. Shukla, K. Ramasamy, JM. Patel, S. Kulkarni, J. Jackson, K. Gade, M. Fu, J. Donham, N. Bhagat, “Storm@ twitter”, InProceedings of the 2014 ACM SIGMOD international conference on Management of data 2014 Jun 18 (pp. 147-156). ACM.

[20] A. Sotsenko, M. Jansen, M. Milrad, J. Rana, “Using a Rich Context Model for Real-Time Big Data Analytics in Twitter”, 4th International Conference on Future Internet of Things and Cloud Workshops, 2014, Pages 228 -233, DOI 10.1109/W-FiCloud.2016.55.

[21] M. Z. Mosharaf Chowdhury and T. Das, “Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing,” in NSDI’12 Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation. San Jose, CA: USENIX Association Berkeley, Apr. 2012.

[22] Y. Yan, L. Huang, L. Yi, “Is Apache Spark Scalable to Seismic Data Analytics and Computations?”, IEEE International Conference on Big Data (Big Data), 2015, Pages: 2036 - 2045, doi: 10.1109/BigData.2015.7363985.

[23] A. Crooks, A. Croitoru, A. Stefanidis, J. Radzikowski, "#Earthquake: Twitter as a Distributed Sensor System", Transaction in GIS, 8 October 2012, doi: 10.1111/j.1467-9671.2012.01359. **x.**

[24] P. Earle, “Earthquake Twitter”, Nature Geoscience 3, 221 - 222 (2010), doi:10.1038/ngeo832.

[25] T. Sakaki, M. Okazaki, Y. Matsuo, "Earthquake shakes Twitter users: real-time event detection by social sensors", Proceedings of the 19th international conference on World wide web, Pages 851-860, 2010, doi: 10.1145/1772690.1772777.

[26] P.S. Earle, D.C. Bowden, M. Guy, "Twitter earthquake detection: earthquake monitoring in a social world", Annals of Geophisycs, vol 54, No 6, 2011, doi: 10.4401/ag-5364.

[27] http://Hadoop.apache.org

[28] H. Chen, R. H. Chiang, and V. C. Storey. Business intelligence and analytics: From Big Data to big impact. MIS quarterly, 36(4):1165–1188, 2012.

[29] Y. Hua, H. Jiang, and D. Feng. Fast: Near real-time searchable data analytics for the cloud. SC14: International Conference for High Performance Computing, Networking, Storage and Analysis, pages 754–765, Nov 2014.

[30] M. M. Rathore, A. Ahmad, A. Paul, and A. Daniel. Hadoop based real-time Big Data architecture for remote sensing earth observatory system. In 2015 6th International Conference on Computing, Communication and Networking Technologies (ICCCNT), pages 1–7. IEEE, 2015.

[31] D. Xu, D. Wu, X. Xu, L. Zhu, and L. Bass. Making real time data analytics available as a service. In Proceedings of the 11th International ACM SIGSOFT Conference on Quality of Software Architectures, QoSA ’15, pages 73– 82, New York, NY, USA, 2015. ACM.

[32] A. R. Baig and H. Jabeen. Big Data analytics for behavior monitoring of students. Procedia Computer Science, 82:43–48, 2016.

[33] F. A. Batarseh and E. A. Latif. Assessing the quality of service using Big Data analytics: With application to healthcare. Big Data Research, 2015.

[34] A. B. Patel, M. Birla, and U. Nair. Addressing Big Data problem using Hadoop and map reduce. In 2012 Nirma University International Conference on Engineering (NUiCONE), pages 1–5. IEEE, 2012.

[35] D. Borthakur, J. Gray, J. S. Sarma, K. Muthukkaruppan, N. Spiegelberg, H. Kuang, K. Ranganathan, D. Molkov, A. Menon, S. Rash. Apache Hadoop goes realtime at Facebook. In Proceedings of the 2011 ACM SIGMOD International Conference on Management of data, pages 1071–1080. ACM, 2011.

[36] A. O'Driscoll, J. Daugelaite, and R. D. Sleator. 'Big Data', Hadoop and cloud computing in genomics. Journal of biomedical informatics, 46(5):774–781, 2013.

[37] Z. Prekopcsa´k, G. Makrai, T. Henk, and C. Gaspar- Papanek. Radoop: Analyzing Big Data with rapidminer and Hadoop. In Proceedings of the 2nd RapidMiner community meeting and conference (RCOMM 2011), pages 865–874. Citeseer, 2011.

[38] H. Duan, Y. Peng, G. Min, X. Xiang, W. Zhan, and H. Zou. Distributed in-memory vocabulary tree for real- time retrieval of Big Data images. Ad Hoc Networks, 35:137–148, 2015.

[39] M. Bilal, L. O. Oyedele, O. O. Akinade, S. O. Ajayi, H. A. Alaka, H. A. Owolabi, J. Qadir, M. Pasha, and S. A. Bello. Big Data architecture for construction waste analytics (cwa): A conceptual framework. Journal of Building Engineering, 6:144–156, 2016.

[40] J. Wei, K. Chen, Y. Zhou, Q. Zhou, J. He. Benchmarking of Distributed Computing Engines Spark and GraphLab for Big Data Analytics. IEEE Second International Conference on Big Data Computing Service and Applications, 2016.

[41] Y. Yan, L. Huang, L. Yi. Is Apache Spark Scalable to Seismic Data Analytics and Computations? IEEE International Conference on Big Data (Big Data), 2015.

[42] F. Provost and T. Fawcett. Data science and its relationship to Big Data and data-driven decision making. Big Data, 1(1):51–59, 2013.

[43] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, “Spark: Cluster computing with working sets.” HotCloud, vol. 10, pp. 10–10, 2010.

[44] W. Yang, X. Liu, L. Zhang, and L. T. Yang. Big Data real-time processing based on Storm. In 2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications, pages 1784–1787. IEEE, 2013.

[45] C. Wang, I. A. Rayan, and K. Schwan. Faster, larger, easier: reining real-time Big Data processing in cloud. In Proceedings of the Posters and Demo Track, page 4. ACM, 2012.

[46] P.B. Makeshwar, A. Kalra, N.S. Rajput, K.P. Singh, Computational Scalability with Apache Flume and Mahout for Large Scale Round the Clock Analysis of Sensor Network Data, National Conference on Recent Advances in Electronics & Computer Engineering, 2015.

[47] V. Ta, C. Liu, G. Wandile. Big Data Stream Computing in Healthcare Real-Time Analytics. IEEE International Conference on Cloud Computing and Big Data Analysis, 2016.

[48] S. Shahrivari. Beyond batch processing: towards real-time and streaming Big Data. Computers, 3(4):117–129, 2014.

[49] B. He, H. P. Huynh, and R. G. S. Mong. Gpgpu for real-time data analytics. Parallel and Distributed Systems (ICPADS), 2012 IEEE 18th International Conference on, pages 945–946, Dec 2012. [51] Q. Shi and M. Abdel-Aty. Big Data applications in real-time

traffic operation and safety monitoring and improvement on urban expressways. Transportation Research Part C: Emerging Technologies, 58:380–394, 2015.

[50] Z. Liu, B. Jiang, and J. Heer. Immens: Real-time visual querying of Big Data. Computer Graphics Forum, 32(3.4):421–430, 2013.

[51] Q. Shi and M. Abdel-Aty. Big Data applications in real-time traffic operation and safety monitoring and improvement on urban expressways. Transportation Research Part C: Emerging Technologies, 58:380–394, 2015.

[52] P. Chopade, J. Zhan, K. Roy, and K. Flurchick. Real- time large-scale Big Data networks analytics and visualization architecture. Emerging Technologies for a Smarter World (CEWIT), 2015 12th International Conference Expo on, pages 1–6, Oct 2015.

[53] H. Demirkan and D. Delen. Leveraging the capabilities of service-oriented decision support systems: Putting analytics and Big Data in cloud. Decision Support Sys- tems, 55(1):412–421, 2013.

[54] L. Deng, J. Gao and C. Vupplapati, Building a Big Data Analytics Service Framework for Mobile Advertising and Marketing, IEEE first imternational conference on Big Data computing services and applications, 2015.

[55] L. Deng, J. Gao, An Advertising Analytics Framework Using Social Network Big Data, 5th International Conference on Information Science and Technology, 2015.

[56] B. He, H. P. Huynh, and R. G. S. Mong. Gpgpu for real-time data analytics. Parallel and Distributed Systems (ICPADS), 2012 IEEE 18th International Conference on, pages 945–946, Dec 2012.