Developing a Real-time Data Analytics Framework For Twitter Streaming Data

Babak Yadranjiaghdam, Seyedfaraz Yasrobi, Nasseh Tabrizi
Department of computer science
East Carolina University
Greenville, NC
{Yadranjiaghdamb15, yasrobis14}@students.ecu.edu, tabrizim@ecu.edu

Abstract— **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, 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.**

Keywords: Streaming processing, Big Data, Kafka, Spark, Twitter, Real-time

I. INTRODUCTION

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 [1]. 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, as a result, there has been a recent transition from hard disk drive storage to memory storage. In-memory processing significantly decreases the amount of access latency, which will have a crucial role when real-time analytics is performed.

Analyzing data in real-time requires data ingestion and processing of the stream of data before the data storage step [2]. Some of the applications of the real-time data analytics are surveillance, environment, health care, business intelligence, marketing, visualization, cybersecurity, and social media. This study presents a real-time data analytics framework for analyzing Twitter data. The basic difference between this study and other researches is that the proposed framework can 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.

II. 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. Thus, 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 most people, their interests, opinions, and 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; such as art, sporting events, health, and many other issues people deal with daily. The common practice among 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 [1].

Dealing with social media data, including many different data types such as text messages, photos, and videos which are arriving in a large volume in every second, needs a proper framework which does not rely upon storing data on hard disks and can process data in memory, as it arrives [3].

There are many studies including [4] 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

978-1-5386-1996-4/17 $31.00 © 2017 IEEE
DOI 10.1109/BigDataCongress.2017.49

routes based on various data sensors is another example of necessity of real-time processing based on information. Social media data on the other hand is usually the sort of data based upon opinions and feelings. Despite 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 [5]. Social media platforms can be quite informative through a crowdsourcing standpoint. Nguyen and Jung [6] 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 was developed [7] 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 [8] 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 [9], which focuses upon different sources of healthcare data varying from biomedical images and EHRs to social media data. Wachowicz et al. [10] 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 [11] search for trends in Twitter posts using complex event-processing 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 [12] make use of real-time data analytics query suggestion and spelling correction.

For data management, many of the methodologies use Apache Storm [13] to analyze data as it arrives (unlike batch processing), 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 can 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 [9, 10, 14].

Spark is a Big Data processing framework built to offer speed beside sophisticated analytics, it runs streaming computation [15] as a series of very small, deterministic batch jobs. These batches may be consisted of Tweets in a short time as low as half a second with a latency of about one second. Although the latency in Spark is just a little bit higher than event processing frameworks [16], in many cases this may be considered close to real-time. The framework presented in this paper 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 particular Twitter data in monitoring the impacts of earthquakes. The authors [17] analyzed 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 their results with the data gathered by U.S. Geological Survey revealed 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 [18] 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 [19] constructed an earthquake reporting system 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.

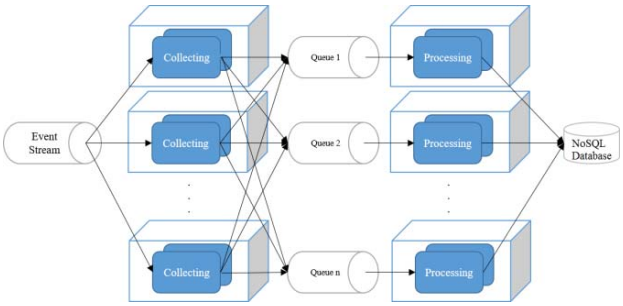
The U.S. Geological Survey (USGS) is investigating how Twitter may help in augmenting earthquake response systems and may improve the delivery of information. The authors [20] show that this fast way of monitoring the earthquakes in 75% of the samples may detect the event within two minutes of origin time, which is much faster than seismographic detections in poorly instrumented areas.

This study presents a framework for analyzing the streaming data in real-time. Using a framework with a powerful engine which has the capability of processing huge streaming information along with its complex analytic features (for instance, machine learning) makes it possible to offer sophisticated analysis in real-time. Sentiment analysis is one of the examples of implementing machine learning algorithms in real-time analysis. However, building the machine learning model is mostly based on the historical data, in this way the produced model will be used in production on live events. However, the engine used does provide some streaming machine learning algorithms, but still often there is a need to do an analysis of historical data.

III. Real-time Data Analytics Framework

The purpose of this study is to develop a framework for analysing Twitter data in real-time. 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 amount of 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.

Data processing workflow usually connects computing resources to automate a sequence of tasks by processing large volumes of data, where 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 framework consists of three sections: i) data ingestion; ii) data processing; and iii) 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 processing tasks on data, and finally sends the results to be monitored. Figure 2 shows the real-time data analytics framework with its different components.

A. 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. To balance the incoming load, Topics are defined and each of these Topics is split into multiple partitions, each storing one or more of those partitions with ability to accept multiple formats, varying from text, image, video and other formats. This is an essential requirement 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 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.

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 our 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, it chooses which record to assign to which partition in the Topic; this may help with the load balancing. Each record published to a Topic is accessed by one consumer, where it 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.

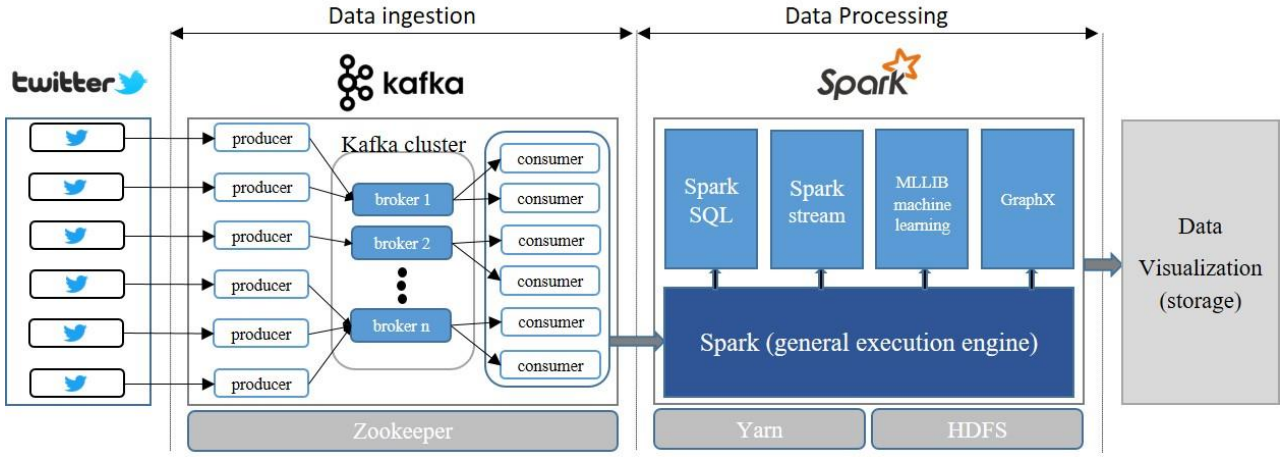


Figure 2: Real-time data analytics framework

B. Streaming Data Processing

In our framework, the data processing task uses Spark as shown in Figure 2. Spark has a core which is the distributed execution engine. Additional libraries, built on the core, allow various workloads for streaming, SQL, and machine learning. These libraries allow Spark to perform more sophisticated processings. For example, machine learning algorithms are often iterative, and Spark's ability to cache the dataset in memory helps enormously to speed 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 [21] for stream processing: i) process each record individually as soon as it is arrived; or ii) combine a set of records in mini-batches. Here, mini-batches may be created either by time or number of records in a batch. 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.

C. 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 Twitter data usually belongs to the current situation and may be much different in another time and circumstance.

D. 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.

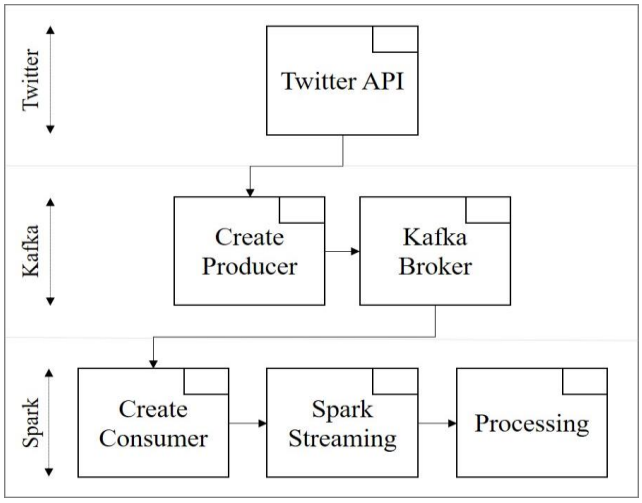


Figure 3: Software Architecture of Real-time Data Analytics Framework

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.

IV. 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 based on 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 with Linux Ubuntu server 14.04 64-bit and Juniper EX4200 switch.

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.

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 close to the one occurred 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

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 world. The earthquake happened almost in the same place and the tsunami waves created an atmosphere of fear among the people living around the affected area. 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. These events have left a very bad memory and have made the people around the world sensitive to tsunami alerts.

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

As a case study, we decided to monitor the reactions on Twitter to this earthquake. This could help to evaluate the real-time nature of our framework, however, there was no time to use more in-depth analysis because of the unpredicted nature of the earthquake. So, we used some regular processing to distinguish the tweets related to this occurrence.

As the news was being broadcasted we started watching the tweets. The flow of tweets was ingested into 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 brokered the incoming stream on the cluster for analysis. The filtering step happened here and the flow of tweets were categorized based upon their content.

Then tweets analyzed based on their time, location, 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.

The processed data was then superimposed (se Figure 6) on the world map for the consecutive periods of eight hours. These maps show that people all around the world reacted to the news, most tweets about this earthquake were initiated from Southeastern Asia, the place that was most affected by the past tsunami. Figure shows the capability of our framework to import data, filter it, and analyze it in real time.

Figure 6 compares how in various hours, people around the world reacted to the so-called earthquake. The importance of such comparative charts could be more sensible if the phenomenon being analyzed has eal-time nature, so the charts can compare every minute changes. As an example, a presidential debate and reactions to each sentence in it can be analyzed by sentiment analysis in this framework.

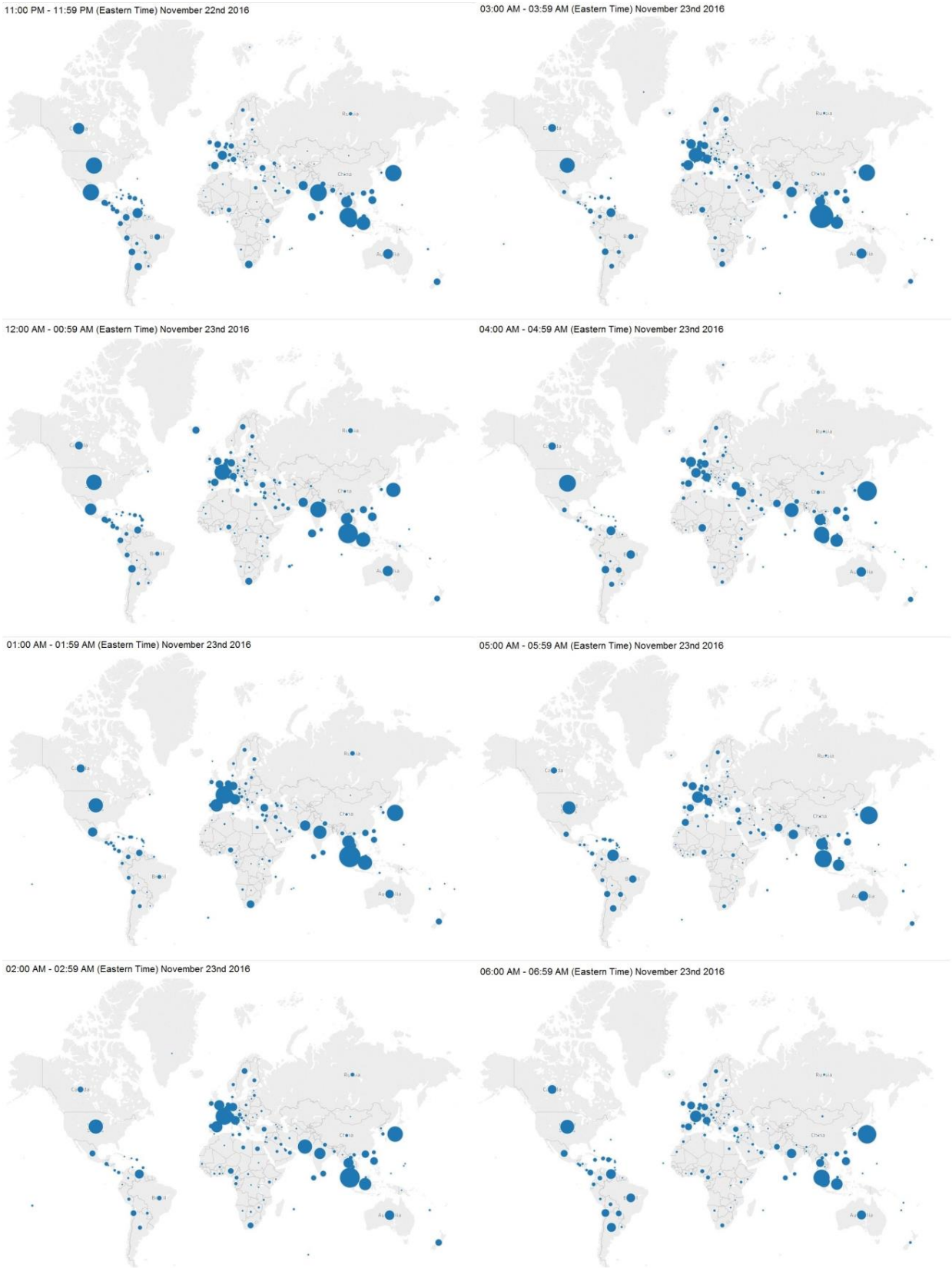


Figure 6: Number of tweets in different countries in an hourly basis for 8 consequent hours.

We started receiving tweets only hours after the occurrence of the earthquake. During that eight-hour period, we received over 50K tweets from all around the world about this incident accounting to 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.

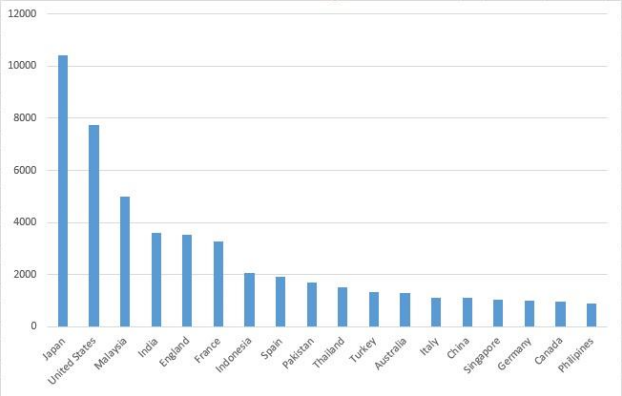


Figure 7: Countries with most tweets about Japan earthquake.

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

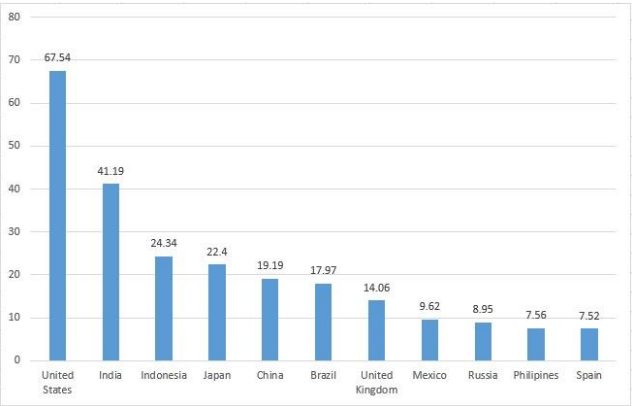


Figure 8: Countries with the highest Twitter users. © Statista 2016

The comparison between Figure 7 and Figure 8 shows that except the countries that were directly affected or located in the same region (e.g. Malaysia), countries with the highest Twitter users 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 relationship 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 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: Number of tweets from each country, the darker the color, the higher tweets are.

The social media data is not 100% accurate, 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 attempted to validate our framework for two important aspects. The first one is whether 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 can detect any tweet with any content from various locations. For the second part of testing we examined how long it took to retrieve the processed results after posting tweets. We could detect the details of tweets and their exact time of posting to a matter of seconds. The total time for this process consists of the network connections' latency, ingestion, processing, 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 connections could affect the total time. For this test, we read data from Twitter every five seconds and started tweeting and retrieving processed data. The results show that our framework was successful in to analyze Twitter data in real-time (or near real-time). We conducted further tests to tweet from distinct locations and various times with the sample hashtags and running spatial and temporal analyses. The result confirmed that our framework can accurately offer location and time based processing.

V. 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 an insight to 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 performed 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 processed 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 in this research aims to show the strength and the importance of real-time data analytics on social media streaming information. Earthquake has been chosen because of its unpredictable nature and vast impact, so the case study mostly tries to uncover the power of the framework to act in real-time, however more in depth analytics can take place in this framework. For instance, sentiment analysis of the tweets from across the country can be done in presidential debates. Doing so, each debate and the reactions of the social media users to it, can be used to train algorithms and in upcoming debates the built model can perform an accurate streaming processing online.

REFERENCES

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

[2] 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.

[3] 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.

[4] 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.

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

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

[7] 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.

[8] 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.

[9] 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.

[10] 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.

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

[12] 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.

[13] 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.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

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

[19] 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.

[20] 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.

[21] 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.