Big Data solutions on a small scale: Evaluating accessible high-performance computing for social research

Dhiraj Murthy and Sawyer A Bowman

Big Data & Society 2014 1:
DOI: 10.1177/2053951714559105

Original Research Article

BIG DATA & SOCIETY

Big Data & Society
July–December 2014: 1–12

DOI: 10.1177/2053951714559105
bds.sagepub.com
SAGE

Big Data solutions on a small scale: Evaluating accessible high-performance computing for social research

Dhiraj Murthy[1] and Sawyer A Bowman[2]

Abstract

Though full of promise, Big Data research success is often contingent on access to the newest, most advanced, and often expensive hardware systems and the expertise needed to build and implement such systems. As a result, the accessibility of the growing number of Big Data-capable technology solutions has often been the preserve of business analytics. Pay as you store/process services like Amazon Web Services have opened up possibilities for smaller scale Big Data projects. There is high demand for this type of research in the digital humanities and digital sociology, for example. However, scholars are increasingly finding themselves at a disadvantage as available data sets of interest continue to grow in size and complexity. Without a large amount of funding or the ability to form interdisciplinary partnerships, only a select few find themselves in the position to successfully engage Big Data. This article identifies several notable and popular Big Data technologies typically implemented using large and extremely powerful cloud-based systems and investigates the feasibility and utility of development of Big Data analytics systems implemented using low-cost commodity hardware in basic and easily maintainable configurations for use within academic social research. Through our investigation and experimental case study (in the growing field of social Twitter analytics), we found that not only are solutions like Cloudera's Hadoop feasible, but that they can also enable robust, deep, and fruitful research outcomes in a variety of use-case scenarios across the disciplines.

Keywords

Big Data, social media research methods, Big Data research methods, digital humanities, digital sociology, Twitter

Introduction

Over the past decade, there has been an exponential increase in the amount of quantitative social trace data—statistical data pertaining to sociological phenomena—available to researchers across the globe. Facebook boasts 1.32 billion active monthly users (Associated Press, 2013) while Twitter, the increasingly pervasive microblogging service, has 271 million active monthly users generating over 400 million tweets a day (Holt, 2013). Other technology companies are part of a rush to bring a wide variety of broad-based and niche social media services, products, and ecosystems into the global online marketplace. For example, Instagram, a social media site for sharing photos that debuted in 2010, has 200 million active monthly users (Instagram, 2014). As a result of this rapid growth, there has been an increasing demand for systems and methods that allow the collection, storage, and analysis of these vast troves of social trace data. Big Data typically refers to data sets so large that they challenge the abilities of more traditional software tools and systems typically used in data collection, storage, and analysis (Manovich, 2011). As the desire and need to efficiently collect and store such large data sets have grown, many researchers have turned towards distributed cloud and cluster-based data storage and retrieval systems that efficiently process Big Data by spreading the data and

[1]Goldsmiths, University of London, London, UK
[2]Bowdoin College, Brunswick, ME, USA

Corresponding author:
Dhiraj Murthy, Goldsmiths, University of London, Lewisham Way, New Cross, London, UK.
Email: d.murthy@gold.ac.uk

processing tasks across many computing nodes (Ruflin et al., 2011).

Increasingly, many forms of computational work in a wide variety of fields in research pertain to tackling large data sets. Heralded as pioneering technology which will "transform how we live, work, and think" (Mayer-Schonberger and Cukier, 2013), Big Data remains a loosely defined, often nebulous term, for large data sets that require complex technologies for the capture, storage, and analysis procedures (Manovich, 2011). Despite the growing trend of marketing and media hyperbole on the value of such data to society as a whole, Big Data does have significant applications to research. According to a recent study, the creation and replication of digital information per year were found to have a growth factor of 44 (Gu et al., 2011). Additionally, rapidly growing emerging markets and the steep increase in web and mobile technologies suggest this growth trend will continue (Baru et al., 2012). As a result of this incredible growth, the need for Big Data technologies in many disciplines is more pressing than ever.

One of the specific areas this article seeks to contribute to is the increasing importance of large collections of social trace data in traditionally low-technology research fields, including the humanities and the social sciences. Moreover, this article emphasizes small-scale solutions that can leverage the power and potential of Big Data technologies that can be replicated, implemented, and maintained with low cost and minimal expertise. While large-scale Big Data research has been conducted (e.g. using the full Firehose Twitter stream), there is little information available about scaling these solutions down to fit the needs of individual researchers or smaller research labs. Cost and ease of setup are limiting factors with regard to large-scale solutions. Therefore, solutions that possess the same tools to process Big Data but on a smaller scale, smaller budget, and with the ability to scale up, empower individual researchers or researchers in non-computational fields to pursue research questions that were previously unfeasible due to limits imposed on the data set by price, experience with technology, and size of the data. Solutions that are capable of handling large volumes of data while addressing the limitations of cost and familiarity with technology are needed for social science and humanities fields to take advantage of Big Data methods.

For individual researchers, there may be a variety of considered pros and cons associated with their decision to attempt to work with and synthesize large data sets. There is little question that more complete social data provide more opportunity of discovery. Businesses looking to market new products, for example, see the utility of Big Data social analytics as a vital means towards understanding their audiences and to better target advertisements. The medical field has used Big Data to better understand novel drugs, the relationships between chemicals, and the potential impact any one chemical may have on the human body (Joshi and Yesha, 2012; Xia et al., 2008). In computer science, social media Big Data has been used for trend detection (Preotiuc-Pietro et al., 2012). Similarly, Big Data can be used as a means of studying social forces. Speaking from our own research, we used Twitter data to investigate urban American social media use (Murthy et al., forthcoming).

Social media have grown enormously. As a result of the increased use of these technologies, the ways in which people interact and connect on a daily basis have changed fundamentally. Social research stands to benefit from analyses of society's deep engagement in technologically mediated culture.

Quantitative sociology has been traditionally driven by manageable, structured data sets. Digital sociology—the sociology of online networks, communities, and social media—is now quickly emerging as a major field due to the rise of social networking sites like Facebook and Twitter. Big Data from these social media sites has been used to study social behavior online (Gold, 2012). With the large amount of data that is potentially available, one should not have to sacrifice size over the quality of the data set or vice versa (Manovich, 2011). As a result of the increased availability and user-friendliness of analytic techniques and jumps in processing power and storage capabilities, a variety of disciplines including, but not limited to, the digital humanities, social sciences, and information systems are becoming increasingly interested in capturing, storing, and analyzing large data sets that were previously inaccessible to most.

Though the literature abounds with Big Data's promise, the very nature of its size represents a significant challenge. The exponential increase in size of available data sets holds the potential for developing a richer understanding of online social formations. However, it can be difficult, expensive, and time consuming to store and process this amount of data. Most make a cost-benefit decision to limit or filter the scope of their research to data sets of a size they know they can handle. This introduces a bias on the direction of research studies within a field. Acquisition of the data presents yet another difficulty. With the ever-decreasing cost of data storage, it is retrieval and organization of data that represent the biggest obstacle. Dimensions such as the height (the number of records), width (the number of variables recorded per record), and diversity pose considerable challenges in making sense of the data (Heer and Kandel, 2012). In addition, the application and study of Big Data in the humanities and

social sciences represent several new and formidable challenges. These disciplines face a steep technological learning curve as they must develop new means for understanding larger sample sizes. Ultimately, however, the stakes are worth it, as these fields are important not only for theory generation around Big Data but also for social critiques leveraging Big Data methods.

The purpose of this article is to explore the feasibility and suitability of emergent and established Big Data on "small-scale" systems. This article presents specific insights gleaned from our experimental case study in building and piloting a small-scale, Big Data collection and analysis engine that collects publicly available streaming data from the Twitter application programming interface (API). We first acknowledge and address the unique challenges and advantages of common distributed Big Data storage and analysis engines in comparison to more traditional approaches. We then discuss our findings and summarize a number of leading Big Data solutions with a focus on which ones might have the most utility when implemented using a minimal configuration and low-cost hardware. Ultimately, we center our attention on Cloudera's Hadoop, a distribution of Hadoop that maximizes performance in storage, retrieval, and analysis for a limited budget. We detail the design, testing, and evaluation of our small-scale Big Data system while commenting on the strengths and limitations of the systems we piloted. Lastly, we present an overview of the types of data we collected. The integration of small data technologies with research in the humanities and social sciences has been met largely with success, but the rise of Big Data poses new challenges to these established methods. Consequently, we feel that our discovery process can serve as a model and proof of concept in many interdisciplinary fields of social research.

"Big Data" challenges in storage, retrieval, and analysis

In understanding the role and potential of Big Data in quantitative social research, it is important to understand and identify the challenges inherent in the collection and use of such data sets. Chief among these concerns is the choice and implementation of technologies that are capable of efficient information storage, retrieval, and analysis at the Big Data scale. While traditional relational databases have been successfully used with large-scale social research, it is important to consider how Big Data generates a new set of challenges that often render these older techniques obsolete or inefficient at best. For instance, traditional database systems may be fully capable of the storage and indexing of large data sets, given available storage space, but depending on the goals of the research, processing efficiency may degrade significantly with the increased size of the data set. Almost all data storage and retrieval engines not specifically designed with Big Data and distributed processing in mind have some fail points in a variety of use-case scenarios beyond which the processing of data becomes untenable. It is important to understand specific processing needs and the capabilities of any chosen storage technology prior to beginning a project. Otherwise, it is possible to end up with vast amounts of data that are unfeasible to successfully navigate. It can often become difficult to simply transfer the collected data into a more appropriate system in certain situations.

All databases have different underlying methods for the storage and retrieval of data, and these differences are best understood through the CAP theorem, which explains three factors that make up the ideal database: consistency, availability, and partition tolerance. Consistency allows for all clients to have the same view of the data. Availability grants each client read and write capabilities. Partition tolerance maintains good system performance in spite of physical network partitions (Hurst, 2013). A strict interpretation of the CAP theorem would argue that no database has been able to satisfy all three, but recent interpretations reveal that modern system designers make trade-offs to each database. Furthermore, some suggest that "there is actually space to outmaneuver the constraints imposed by the CAP theorem with clever design" (Andrikopoulos et al., 2012).

In this section, we explore the three key functions of any database system: storage, retrieval, and analysis, while giving consideration to the differences, with regard to CAP, between more traditional relational database management systems (RDBMS) and emerging distributed systems and how each responds to the unique challenges posed by Big Data.

Storage

The RDBMS is the most widely used database across the world. The strengths of RDBMS are consistency and availability of the data. This means all clients have the same view of the data and each client can perform read and/or write operations (Padhy et al., 2011). Data are traditionally stored as tables, and RDBMS stores these data by forging relations between pieces of data. In other words, the RDBMS links different pieces of information together by assigning tables to keys (Padhy et al., 2011). However, relational databases are generally not ideal for the storage of Big Data. As the size of a data set continues to grow, the database must also scale. Although relational databases exhibit great vertical scalability, they have restrictions on just how far they can scale. The large amount of

data makes it extremely difficult for the storage of the data on a single machine or cluster (Ruflin et al., 2011). It is possible to add more space on that machine or cluster, but this vertical scalability has limits. Big Data lends itself to newer techniques that utilize elastic scaling, scaling out, or horizontal scalability. This process entails distributing the data across several hosts or servers, which allows for an easier and more dynamic storage of Big Data (Nance et al., 2013).

Retrieval

The biggest challenges posed by Big Data are the ability to retrieve, sort, and filter large data sets. Typically, these tasks are aided through partitions and indexes. Partitions physically store data files in different locations based on the range of values of some defined variable. Queries that seek to filter results based on the partitioning variable only need access to the data bins defined in the query, thus speeding up retrieval. Indexes typically evaluate the diversity of values of a given variable and create a reference structure in memory (like a binary search tree). This allows for fast identification and retrieval of data when queries ask for records with a variable exactly matching one or more specific values. With large data sets in RDBMS systems, performance of indexes and partitions typically degrades as the number of records grows. Every table insertion can cause the index to be rewritten; over time, this can add up to many additional processing cycles. In distributed database systems, the responsibility for indexes and partitions is distributed across all the nodes in the system. Technically, they would still be subjected to the performance drag of RDBMS, but the effects may not be noticeable until one had collected orders of magnitude more data records than was possible with RDBMS.

Data processing

In the social sciences, social trace data are often composed of many different data types (Ruflin et al., 2011). This provides a considerable challenge to relational databases, which have a static schema. This quality enhances performance with structured data, but it proves to be a limitation in other scenarios (Padhy et al., 2011). For instance, with the introduction of social data that is semi-structured or unstructured, databases that can adapt, change, and accommodate new data types become more desirable than those with rigid schema (Ruflin et al., 2011). For some research questions, non-relational databases provide for less stringent data model restrictions (Padhy et al., 2011). Furthermore, they allow for easy incorporation of new data types, which is valuable to research situations where the data is in flux. These systems offer new forms of flexibility, especially in terms of storing new, diverse, and high-volume data.

Common Big Data technologies

The use of non-relational database systems has risen substantially over the past few years due to benefits such as scalability, high availability, fault tolerance, and a compatibility with heterogeneous data (Shi et al., 2010). While each database strives to be flexible yet robust, the application and implementation tend to vary significantly. As a result of the differences in performance between each database, we have selected three databases for comparison. In this section, we quickly compare MongoDB, Riak, and Hadoop and investigate their overall performance, applicability to the collection and analysis of social data, and their limitations. It should be noted that the purpose of this is not an in-depth comparison, but rather a brief overview.

MongoDB is one popular NoSQL database. The schema is flexible and largely uses document structure and storage. The documents are of JavaScript Object Notation (JSON)-style, a type of text-based data that offers both simplicity and power (Dede et al., 2013) (JSON is used by the Twitter API and many other social media platforms). Additionally, indexes allow for the quick organizing of documents, particularly ones corresponding to frequent queries. MongoDB creates a replica set of documents that ensures an automated failover. This also provides for redundancy and high availability. MongoDB also scales through sharding, a process that partitions a collection of documents and then stores each segment on a different machine (Dede et al., 2013). This creates a balanced load across the machines. MapReduce is also a critical component of MongoDB. This command is meant to handle complex aggregation jobs. The map function ensures each instance is created and the reduce function creates ''sorted groups of instances that share a common key'' (Borkar et al., 2012). GridFS, another key component, is used to store and retrieve files that exceed the BSON—the binary representation of JSON documents—document size limit. It achieves this by dividing a document into parts and creating multiple new documents. While MongoDB represents a powerful database technology, there are many limitations, particularly with the user interface. For example, MongoDB tends to be significantly slower—nearly five times slower—than the Hadoop Distributed File System (HDFS) for large data input (Dede et al., 2013).

Riak is another popular NoSQL database. It places emphasis on availability achieved through replication. Additionally, data are retrieved so that read and write

operations can be called even during failure conditions. In the case of a network or hardware failure, loss of access to nodes can occur without data loss. Adding machines to the cluster can be done easily, and the data in that cluster are automatically distributed through hashing. Each node is the same, and this sets a foundation for fault tolerance and scalability. Riak uses a key/value model for object storage. Any type of data can be stored as an object. Much like MongoDB, Riak uses MapReduce for aggregation tasks, though Riak has its own search and index system. While Riak is able to rebalance automatically due to dividing data space into equal partitions, this process of equal partitioning is overwhelmed by a high load of data (Konstantinou et al., 2011). Consequently, Riak does not compare with HDFS in high-request rate scenarios.

Hadoop, a popular open-source platform for data-intensive applications, has a software library that is used to process large data sets (White, 2012). Like MongoDB, it also uses the MapReduce model. This is done through the use of several nodes, which make Hadoop both reliable and highly available (Dede et al., 2011). In comparison to MongoDB, Hadoop outperforms in read and write operations as well as scalability (Dede et al., 2013). Another project-specific goal we considered included cost. As a result, we had to consider how Hadoop would perform as a single node of low-node cluster on low-cost hardware. Hadoop's performance over MongoDB and its ability to handle a higher load of data than Riak make Hadoop a great candidate for social research. The configuration options, availability of support, and the active development for scalability were also factors that were weighed in our decision. Specifically, a significant amount of Big Data social research is Hadoop-based and online support is readily available. Though there are several different implementations of Hadoop, the most relevant to social data projects is the Cloudera open-source distribution of Hadoop (Cloudera Inc., n.d.). Essentially, this version of Hadoop not only possesses the same methods, functions, and general properties that Hadoop has, but it also incorporates other Big Data tools to effectively sort social data objects, such as tweets. Specifically, Flume, HDFS, Oozie, and Hive are all used in this control flow as a means of storing, sorting, and analyzing the data.

Table 1 summarizes the three databases discussed in this section. This table highlights key differences between the three databases for NoSQL implementations. Of the three prominent databases explored in this section, Apache Hadoop is the most widely used and perhaps most applicable to the common needs of Big Data research in the social sciences (Khuc et al., 2012; Lee et al., 2012). For example, Hadoop has been successfully implemented in a social media cloud-computing application (Kim and Lee, 2011) and in the Lydia TextMap system which enables social scientists to study the intersections between blogs, newspapers, patents, scientific abstracts, and legal documents (Bautin et al., 2010). This is largely due to Hadoop offering a higher read rate, which is essential for processing the large amounts of data that must be read into the database's storage (Ruflin et al., 2011). Additionally, the combined scalability (elasticity) and relative speed for a high throughput of data make Hadoop a good fit (Dede et al., 2013; Konstantinou et al., 2011).

Table 1. Comparison between NoSQL data collection technologies.

	Difficulty level[a]	Language	Querying	Storage
Mongo DB	Easy	C++	JavaScript	Document
Riak	Moderate	Erlang & C	Riak Search	Key-Value
Hadoop	Moderate	Java	Hive	Column

[a]Level of difficulty/complexity of each, taking into special consideration setup of required hardware, databases, and querying languages. In fields that are not traditionally associated with technology, it is important to consider the learning curves associated with these different tools.

Experimental case study: Twitter data collector using hive

Though there is a desire on the part of researchers across the disciplines to implement powerful distributed database architectures, little information is available to suggest how this should be accomplished at the level of individual researchers (rather than well-staffed labs). Enterprise-level Big Data storage solutions are designed and optimized to be deployed on large cloud-based server farms consisting of hundreds to thousands of nodes. This is entirely appropriate for a system of providing database services simultaneously to many connected clients. Little information is available about how such technologies can usefully be leveraged by individual researchers or small teams with one or two large data sets that they would like to be able to analyze. Even less information is available about performance of such systems when implemented at a small scale (1–5 nodes). The lack of information about such systems implemented on small scales serves as a critical barrier towards researchers attempting to use such technologies. Often, they will not be able to determine if their project is feasible, possible, or if it will perform better or worse than their current data management system. As a result, most stick with what they know: possible investing, constant upgrades, workarounds, hacks, and consultants to get their legacy data systems to meet their basic needs. A sad consequence of this is that

the divide between business and academic expertise and applications of Big Data methods continues to grow.

Specifically, most information available about the design and implementation of distributed storage and retrieval systems focuses on large, multi-node systems, which are likely overkill for most academic social research case scenarios. This has the effect of influencing those researchers determined to take the leap to cloud-based storage and analysis engines to perhaps over-invest in hardware, personnel, and support for tasks which could have been accomplished with a much lower investment. Both of these problems potentially contribute to an unnecessary drain of research funding pools, and funds are often misallocated or over-allocated. Our approach to this case study was to choose a popular and well-documented distributed storage engine and implement it at the smallest reasonable scale. We would then populate the system with Big Data and see how it performs in comparison to more traditional approaches. The value of this approach is the potential to determine what is possible at the lowest level (lowest entry barrier), while at the same time detailing a system which can easily be incrementally scaled up to grow with a live project's data and performance needs.

Towards this end, the following criteria were set as goals for our experimental case study. The first goal was to develop a system using commonly available hardware with a sub $5000 (£3000) price point. We decided to implement the freely available packaged distribution of a common distributed-data storage engine, in this case the Cloudera distribution package of the Apache Hadoop ecosystem. Our case study was a system capable of collecting up to one year's worth of Twitter data from a 1% sample of all tweets. (While our system has indeed collected a year's worth of data, we only test on three months of data in this article.) Additional considerations were given towards creating a system that would be fault tolerant and provide acceptable retrieval and analysis performance in at least some common use cases.

Resources

The backbone of the system we designed was a stock Dell PowerEdge T320 server. This machine has twelve 1.9 GHz processing threads on six cores. We added 32 GB of RAM and four 2 TB hard drives in a RAID 3 configuration, providing 5.4 TB of usable storage space on the single node. It should be noted that in multi-node systems, data replication can be employed across the node, making RAID unnecessary. This system was built and configured for $3000 (£1800). Though several configurations met our goals, this one is entirely middle of the road, readily available, and easily extensible. This represents considerable savings over traditional "I/O hungry" RDBMS node clusters, which can cost $50,000 for comparable implementations (Leetaru, 2012). In the next two sections, we discuss using Twitter as a data source, and we outline our approach in setting up a single-node Hadoop database.

Data source

With a user base of over 500 million, Twitter represents one of the most popular social media platforms and one which is seen as a valuable source for business intelligence (Culnan et al., 2010). The number of users continues to grow rapidly, and the amount of data generated by this user base is on the scale of Big Data. For example, the full Twitter Firehose, a paid API source that allows for the capture of every tweet, streams over 1.5 TB per day (Mishne et al., 2013). While our low-cost system is not appropriate for the capture and storage of the entirety of Twitter through Firehose, it can capture Twitter samples well into the Big Data level. There are a total of 112 metadata fields of Twitter data grouped into four different categories: Tweets, Users, Entities, and Places (Twitter, 2013). Social researchers have been particularly interested in the collection of Twitter data because of the ease of accessibility and richness of data. It is possible for the researcher to select particular metadata fields that best contribute towards the support of a specific hypothesis.

In our introduction, we highlighted the obstacles that traditionally limit Big Data research in the social sciences. One classic trade-off is size versus quality of the data set. Without Big Data technologies, researchers are forced to choose between breadth and depth. Often, the way researchers filter introduces biases in the collected data set. However, our system allows for a broad collection of items—tweets in our case study—as well as all of the associated metadata. One can then comb through these data with HiveQL queries to find data that are relevant to their particular case study, rather than starting with small data. We were able to answer a variety of sociological research questions with this data set (particularly regarding demographic attributes of users and the use of Twitter via mobile versus web clients). We were also able to detect tweet patterns. For example, using the time zone and time of tweet metadata, we were able to create graphs that visualized the tweeting patterns and behaviors in different time zones around the world. Not only did these graphs demonstrate differences in online behavior around the world, but they also revealed abnormal frequency spikes that could be linked to current events. Also, by pairing the matching of words with emotional content within tweets with the device a user tweeted from (i.e. mobile or web), we were able to evaluate

whether mobile and web-based tweets were likely to be more positive or negative. As Twitter introduces more metadata fields, the flexibility of our architecture would enable us to handle these changes gracefully.

To facilitate collection, Twitter provides an API that allows researchers to connect to the stream. The API allows the researcher to request data via either the REST, an architectural style that relies on HTTP and XML, or the stream. Both provide numerous data collection options. The Spritzer stream collects approximately 1% of the total stream flow in real time (Natkins, 2012b). Other options include location streams, keyword, or REST calls for individual pieces and/or blocks of information. The data arrive in JSON format and contain data associated with the corresponding tweet in a text-based format (Bo, 2010). What makes Twitter especially interesting from a sociological perspective is not just the amount of data generated but also the relationships, trends, and social meaning that can be studied with these data. In particular, Twitter data allow for the aggregation of opinions, ideas, and trends by socio-demographic characteristics such as location, time of day, and pace of tweeting (Sankaranarayanan et al., 2009). In this sense, Twitter is increasingly seen by some social researchers as an important way of visualizing relationships and social communication between people (Boyd and Crawford, 2012; Murthy, 2013). In contrast to more traditional means of communication, Twitter provides an environment of almost synchronous feedback (Sankaranarayanan et al., 2009). As a result, Twitter acts as a source for the distribution of news and information (Park and Chung, 2012), which can provide important social insights.

Setup and configuration

There are several setup and configuration issues which should be considered at the start of any Big Data social research project or proposal. The first is data ingestion. This typically involves establishing a connection to a data source, a possible transformation step, and moving the data to a final storage location. The next consideration is the management and organization of data within the data store itself. Organization typically involves partitioning data, the indexing of variables within data records, and other concerns which help ease retrieval and analysis tasks. The main concerns in the retrieval and analysis step are to plan what ways one will connect to the data, store and request data, possibly process, transform, and/or analyze the results, and finally deliver those results to the requester via some method or format. Figure 1 illustrates the architecture of our single-node Hadoop database.

There are two main components to our data ingestion component. These are the Twitter API itself and a

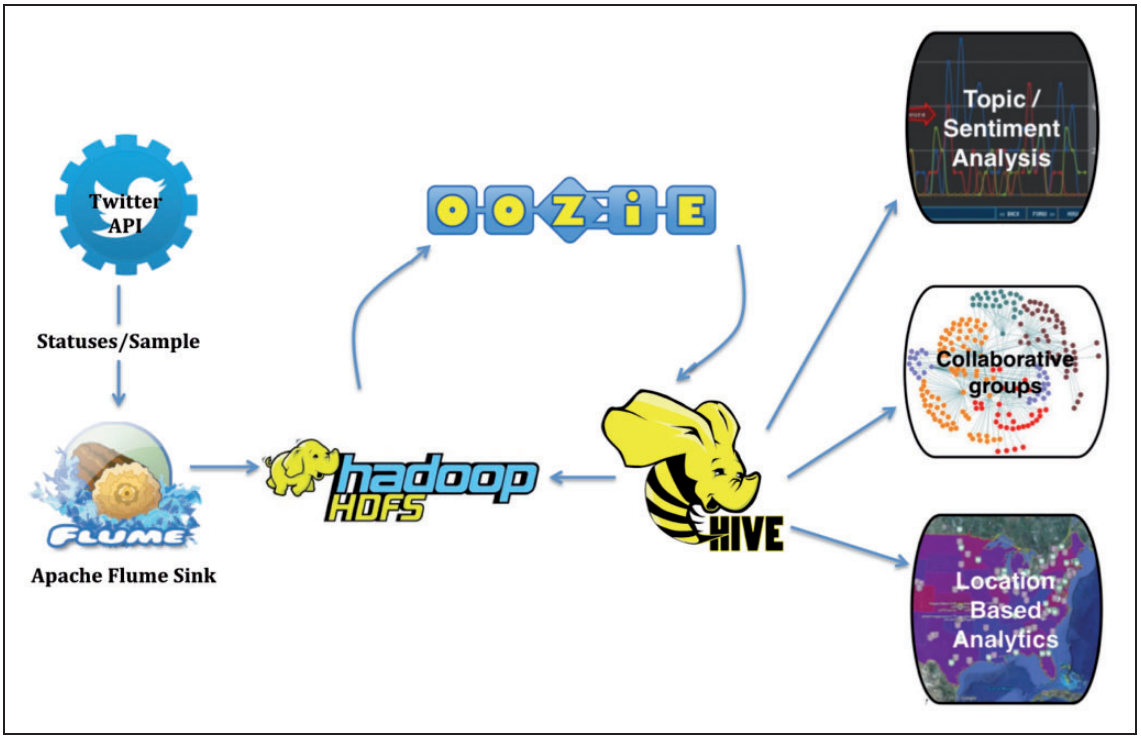

Figure 1. Our architecture implementing Hadoop.

Flume data source. Flume is a highly configurable open-source data ingestion system that will connect to one or more user-defined data sources and push the data to one or more data sinks, via data channels (Natkins, 2012b). Flume's sources, channels, and sinks allow for the definition of complex data flows. In this case, the data source used the open source twitter4j Java library to connect to the Twitter statuses/sample stream and push these data to an HDFS.

Once our data had been loaded into the HDFS, we used the Oozie workflow automation tool to automatically partition these data by hour and prepare these partitions to be accessible to storage and retrieval requests. Oozie is a component of the Apache Hadoop ecosystem that allows for the definition of potentially complex and repeating automated workflows (Natkins, 2012a). In this case, Oozie was configured with parameters that ran a script to automatically create data partitions and prepare these data for access via the Hive database system.

As Figure 1 illustrates, we used Hive as our tool for enacting queries on data stored in the HDFS. Hive is a data warehouse system that works in coordination with Hadoop to achieve easy data summarizing, ad-hoc queries, and analysis of large data sets (Apache Hadoop, 2013). It accesses the data in Hadoop through an SQL-like language. The performance of data loading and range queries in Hive is strong and surpasses the performance of alternative tools such as Cassandra and HBase in these categories (Shi et al., 2010). Hive is effective because it can handle unstructured, semi-structured, and poly-structured data (Natkins, 2012c). Hive has the ability to define a fully dynamic data serialization and deserialization interface (SerDe). The previous two features allow the storage of data in its native format, which allows for the processing of only the queried data as opposed to each piece of incoming data. This, in turn, makes insert speeds faster. Data can be searched and records processed while only parsing the data from its original format as needed (Natkins, 2012c). This is a particularly powerful technique for semi-structured data like XML or JSON, and one which is immediately applicable to the JSON Twitter records which are returned via the Twitter API. This workflow illustrates how the individual tools discussed throughout this section compose an efficient and accessible architecture. This combination of tools also enabled us to efficiently analyze our collected Twitter data.

Collection, retrieval, and analysis

For this case study, our Twitter collection and analysis system collected tweets from June 2013 to August 2013. It ran without any system-caused failures. Analytic tools bundled with the Cloudera distribution of Hadoop indicated that the system was not overly taxed and was in general good health over the research period, even while performing processor intensive queries alongside new data ingestion. In the first month, over 150 million metadata-enriched tweets were collected (approximately 5 million tweets per day), which consumed approximately 300 gigabytes worth of storage space. The collector read and stored, on average, 191,315 tweets or 0.4 GB per hour. This confirms the Spritzer stream's advertised 1% sample rate as the total volume of worldwide tweets was around 500 million the time of data collection (Holt, 2013).

HiveQL is the language utilized by Hive and is modeled after SQL for the sake of familiarity and includes SQL commands like "from clauses, joins, group bys, aggregations, and create table as select" functions (Stewart et al., 2011). However, not all SQL functions are implemented in HiveQL and vice versa. For example, HiveQL lacks some of the functions and indexing capabilities that are available in SQL, but HiveQL also offers extensions such as multi-table inserts, transform, map, and reduce functions (White, 2012).

We performed a variety of tests involving select variables from the Twitter data to investigate potential methods of data analytics. For example, we used the Hive ngrams function to create four different tables consisting of the top 10,000 unigrams, bigrams, trigrams, and quadgrams per language across the entire month. We were also able to generate tables of the top 1000 ngrams per day and evaluate the change in frequency for a particular ngram over the period of the month. Regular expression extractions allowed us to filter for a particular word that occurred after a common phrase. For example, we searched for the frequency of words that followed "I love" and "I want to buy." These queries and extractions provided a closer lens for examining the patterns and use of phrases as well as the exchange of information on Twitter.

Evaluation

In working with Hadoop and Twitter data, we experienced success as well as unanticipated challenges. Ultimately, we found that this architecture presented a suitable solution to overcoming previous difficulties with storing and filtering Big Data. In order to contextualize our single-node implementation of a Hadoop database within the sphere of Big Data research, we compared the performance of our Hive-Hadoop setup with a more traditional RDBMS database that we previously created. Our Hive-Hadoop database performed a full table scan of 150 million records in approximately 2–2.5 h on average. On the other hand, our RDBMS

database performed a full table scan of 250 million records, from a different database but using the same Twitter source, in over 14 hours. Taking the difference in size of each data table and the amount of time taken to perform a full table scan into consideration, our Hive-Hadoop database still demonstrates a 300% increase in performance over a similarly structured and queried MySQL database. This finding is notable for several reasons. First, it demonstrates that our Hive-Hadoop solution functions well for the collection and processing of Twitter Spritzer stream data. Second, it exhibits Hadoop's ability to offer an increase in performance over RDBMS solutions on certain tasks. Lastly, our Hadoop solution represents a low-cost (under $5000) database that can be set up with minimal expertise required. Our low-cost commodity hardware, single-node Hive-HDFS solution was not only designed and implemented with minimal Big Data technical expertise, but it also met and usually exceeded the performance of RDBMS in many situations. Our case study also affirms the accessibility of Hadoop in Big Data research, even in small-scale single-node implementations.

The most significant performance gains were seen on the largest data queries. Hive was strongest in retrieving and processing large chunks of data, as it can distribute the load across many nodes and processors. Despite this, Hive is relatively weak for seek operations to find specific pieces of information. This is mostly because of the large amount of overhead associated with performing any one query. A query (of any size) requires the start-up and initialization of at least one JVM, Java Virtual Machine, per node to process the thread and manage the job's execution. This can take several seconds per initialization and it does not significantly benefit from increased storage or processing capacity. Thus, even for a query on an indexed variable, there is a fixed start-up cost even if the query will eventually execute in sub-second time. This situates Hive-based Big Data solutions as one of the better options for very large table scan type analyses where time to result is less important and critically inadequate for queries where a near real-time response is desirable or required. A happy solution may be to use tools like Oozie to stage recent or historical data incrementally over time in other distributed data engines, which can provide real-time query response at the cost of memory, like Cassandra.

Another potential benefit that we identified in our architecture is the addition of nodes. This implementation of multiple nodes—as opposed to our single-node system—would increase the performance of the system as a whole. For example, both Cassandra and HBase exhibit significant improvements in the speed of most every operation from five nodes to 19 nodes (Shi et al., 2010). Due to the marked improvement of both Cassandra and HBase, we hypothesize that our Hive-Hadoop system would also result in similar increases in performance. The real benefit of the system structure we have tested is that storage or processing capacity can be incrementally added to obtain desired performance.

Limitations

One potential limitation we identified for Hive users relying on previous experience with SQL-like languages is that HiveSQL only implements a subset of the advanced functions and features of the SQL language specification. Many of these features may be recreated as Hive may be extended to include user-defined functions or one may be forced to reformulate their query using only the implemented features. In almost every case, there would be an acceptable work around. But those considering using Hive should be aware of this limitation to avoid encountering potential stumbling blocks.

Another limitation is that any system of any size eventually will be constrained by memory or storage limitations. For instance, a single-node Hadoop system is likely not appropriate for collecting the full Twitter Firehose, unless you are only interested in doing so for a short period of time. It would only be able to ingest data for about four days before available storage resources were consumed. In any event, the budgetary restrictions of most academic social media research limit researchers to freely available data rather than paid data like Firehose. Using our system architecture, we could collect Twitter's free streamed data for more than a year. It should be noted that once storage capacity is exhausted, one would still be able to fully utilize the retrieval functions of their chosen database system, and additional storage and processing capacity can be added at any time via the addition of new nodes.

Conclusion

Big Data, with its promise of "complete" data sets, comes with an enormous level of complexity both in terms of storage and data analysis. Of course, this presents barriers of cost and technical expertise both within traditionally technical disciplines as well as within the humanities and social sciences. That being said, the high levels of social data being created as part of our online social media footprint have attracted the attention of social scientists in Big Data. Furthermore, the digital humanities cut their teeth on the digitization of large corpuses of books, and the field saw immediate payoff in Big Data analytics. Both traditionally technical fields as well as disciplines which have been

historically less technical see two major challenges in moving into Big Data research: cost and technical expertise. This article has sought to ameliorate these two challenges by presenting our evaluation of Big Data solutions for a Twitter-based social research data project. Using a small-scale yet extensible hardware setup, we used Apache Hadoop and Hive to provide an efficient and cost-effective storage and analytics solution for the collection of approximately 150 million tweets/month.

Ultimately, the increase of database solutions to handle Big Data is often confusing, as it is challenging for one to understand what platforms are most suitable for newer forms of data such as social media data. In an attempt to find a suitable database for the collection, storage, and analysis of large amounts of Twitter data for our study, we have compared and contrasted prominent database solutions. Like others, we sought a database that could provide horizontal scalability, a flexible data schema for unstructured social data, a familiar language like SQL, an intuitive user-interface, fast read and write capabilities, a reliable architecture, partitioning capabilities, and a sound method for analysis. RDBMS and NoSQL solutions were introduced and explored, though we ultimately chose to implement Hadoop. While Hadoop has major limitations, its performance in key categories outstrips the other databases we tested it against. As a result, we adapted Cloudera's version of Hadoop for our Big Data Twitter project. With the use of Big Data tools including Flume, Oozie, and Hive, we were able to effectively plug into the Twitter API, stream the unstructured JSON data into a distributed file storage system, automatically process work flows, and organize the previously unstructured data into partitions loaded into a query-able data table.

The experimental results discussed demonstrate not only the power of this solution but also its ability to be used for innovative forms of hypothesis generation for social research. The overhead associated with large-scale Big Data technologies often hinders individual researchers from pursuing hypotheses that require the size and quality of Big Data sets. However, the minimal technical expertise it takes to set up and utilize a small-scale Big Data technology compensates for lack of previous Big Data experience. The results detailed in this article merely scratch the surface when it comes to possible queries one could make through Hive. Our work not only evaluates and demonstrates the effectiveness of Hadoop in handling the challenges of Big Data, but it also critically addresses its limitations. There is great potential for Hadoop in terms of social media data collection. Ultimately, we found that our Hadoop-based architecture enabled us to implement a cost-effective data collection and analysis framework for a large Twitter-derived data set. Our aim is to provide a model for practitioners across the disciplines who are currently evaluating Big Data solutions but are on a budget and may have limited sets of expertise. We also feel that the gap in Big Data research methods between business and social research applications has been growing, and this article seeks to bridge some of this divide by opening the hood to accessible Big Data methods.

Acknowledgements

The authors thank Alexander Gross for extensive, generous, and invaluable technical assistance and support in all phases of this project and for assistance in producing Figure 1.

Declaration of conflicting interests

The authors declare that there is no conflict of interest.

Funding

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

References

Andrikopoulos V, Fehling C and Leymann F (2012) Designing for CAP – The effect of design decisions on the CAP properties of cloud-native applications. In: *CLOSER*, Porto, Portugal, 18 April.

Apache Hadoop (2013) Apache Hive TM. Available at: https://hive.apache.org/ (accessed 8 April 2013).

Associated Press (2013) Number of active users at Facebook over the years.– *Yahoo! News*, 1 May. Available at: http://bigstory.ap.org/article/number-active-users-facebook-over-years-5.

Baru C, Bhandarkar M, Nambiar R, et al. (2012) Big data benchmarking. In: *Proceedings of the 2012 workshop on management of big data systems*. San Jose, CA: ACM, pp. 39–40.

Bautin M, Ward CB, Patil A, et al. (2010) Access: News and blog analysis for the social sciences. In: *Proceedings of the 19th international conference on world wide web*. Raleigh, NC: ACM.

Bo Y (2010) *Querying JSON streams*. Uppsala, Sweden: Uppsala University.

Borkar V, Carey MJ and Li C (2012) Inside "Big Data management": Ogres, onions, or parfaits? In: *Proceedings of the 15th international conference on extending database technology*. Berlin, Germany: ACM, pp. 3–14.

Boyd D and Crawford K (2012) Critical questions for big data: Provocations for a cultural, technological, and scholarly phenomenon. *Information, Communication & Society* 15(5): 662–679.

Cloudera Inc. (n.d.) Cloudera standard. Available at: http://www.cloudera.com/content/cloudera/en/products/cloudera-standard.html (accessed 10 June 2013).

Culnan MJ, McHugh PJ and Zubillaga JI (2010) How large US companies can use Twitter and other social media to

gain business value. *MIS Quarterly Executive* 9(4): 243–259.

Dede E, Govindaraju M, Gunter D, et al. (2013) Performance evaluation of a MongoDB and hadoop platform for scientific data analysis. In: *Proceedings of the 4th ACM workshop on scientific cloud computing*. New York, NY: ACM, pp. 13–20.

Dede E, Govindaraju M, Gunter D, et al. (2011) Riding the elephant: Managing ensembles with Hadoop. In: *Proceedings of the 2011 ACM international workshop on many task computing on grids and supercomputers*. Seattle, WA: ACM, pp. 49–58.

Gold MK (2012) *Debates in the digital humanities*. Minneapolis, MN: University of Minnesota Press.

Gu X, Hou R, Zhang K, et al. (2011) Application-driven energy-efficient architecture explorations for big data. In: *Proceedings of the 1st workshop on architectures and systems for Big Data*. Galveston Island, TX: ACM, pp. 34–40.

Heer J and Kandel S (2012) Interactive analysis of big data. *XRDS* 19(1): 50–54.

Holt R (2013) Twitter in numbers. *Telegraph*, 21 March. Available at: http://www.telegraph.co.uk/technology/twitter/9945505/Twitter-in-numbers.html.

Hurst N (2013) Visual guide to NoSQL systems. *Nathan Hurst's Blog*. Available at: http://blog.nahurst.com/visual-guide-to-nosql-systems (28 July 2013).

Instagram (2014) Instagram. Available at: http://instagram.com/press/ (accessed 24 October 14).

Joshi K and Yesha Y (2012) Workshop on analytics for Big Data generated by healthcare and personalized medicine domain. In: *Proceedings of the 2012 conference of the center for advanced studies on collaborative research*. Toronto, Ontario: IBM Corp, Riverton, NJ, USA, pp. 267–269.

Khuc VN, Shivade C, Ramnath R, et al. (2012) Towards building large-scale distributed systems for Twitter sentiment analysis. In: *Proceedings of the 27th annual ACM symposium on applied computing*. New York, NY, USA: ACM, pp. 459–464.

Kim M and Lee H (2011) SMCC: Social media cloud computing model for developing SNS based on social media. In: *Convergence and hybrid information technology*. Daejeon, Korea: Springer, pp. 259–266.

Konstantinou I, Angelou E, Boumpouka C, et al. (2011) On the elasticity of NoSQL databases over cloud management platforms. In: *Proceedings of the 20th ACM international conference on information and knowledge management*. Glasgow, New York, NY, USA: ACM, pp. 2385–2388.

Lee G, Lin J, Liu C, et al. (2012) The unified logging infrastructure for data analytics at Twitter. *Proceeding of the VLDB Endowment* 5(12): 1771–1780.

Leetaru KH (2012) Towards HPC for the digital humanities, arts, and social sciences: Needs and challenges of adapting academic HPC for big data. In: *2012 IEEE 8th international conference on E-Science (e-Science)*, Chicago, IL, 8–12 October.

Manovich L (2011) Trending: The promises and the challenges of big social data. In: Gold MK (ed.) *Debates in the digital humanities*. Minneapolis: University of Minnesota Press, pp. 460–475.

Mayer-Schonberger V and Cukier K (2013) *Big data: A revolution that will transform how we live, work, and think*. New York, NY: Houghton Mifflin Harcourt.

Mishne G, Dalton J, Li Z, et al. (2013) Fast data in the era of big data: Twitter's real-time related query suggestion architecture. In: *Proceedings of the 2013 international conference on management of data*. New York, NY: ACM, pp. 1147–1158.

Murthy D (2013) *Twitter: Social Communication in the Twitter Age*. Cambridge, UK: Polity Press.

Murthy D, Gross A and Pensavalle A (forthcoming) *Urban Social Media Demographics: An Exploration of Twitter use in Major American Cities*, Forthcoming manuscript (contact author for manuscript copy).

Nance C, Losser T, Iype R, et al. (2013) NOSQL VS RDBMS – Why there is room for both. In: *Proceedings of the Southern Association for Information Systems Conference*, Savannah, GA, USA, 8–9 March.

Natkins J (2012a) *Analyzing Twitter Data with Apache Hadoop*. Apache Hadoop for the Enterprise, Vol 2013. Palo Alto, CA: Cloudera.

Natkins J (2012b) *Analyzing Twitter Data with Apache Hadoop, Part 2: Gathering Data with Flume*. Apache Hadoop for the Enterprise, Vol 2013. Palo Alto, CA: Cloudera.

Natkins J (2012c) *Analyzing Twitter Data With Apache Hadoop, Part 3: Querying Semi-structured Data with Apache Hive*. Apache Hadoop for the Enterprise, Vol 2013. Palo Alto, CA: Cloudera.

Padhy RP, Patra MR and Satapathy SC (2011) RDBMS to NoSQL: Reviewing some next-generation non-relational databases. *International Journal of Advanced Engineering Science and Technologies* 11(1): 15–30.

Park JY and Chung C-W (2012) When daily deal services meet Twitter: Understanding Twitter as a daily deal marketing platform. In: *Proceedings of the 3rd annual ACM web science conference*. New York, NY, USA: ACM, pp. 233–242.

Preotiuc-Pietro D, Samangooei S, Cohn T, et al. (2012) Trendminer: An architecture for real time analysis of social media text. In: *Proceedings of the workshop on real-time analysis and mining of social streams*, Dublin, Ireland, 4 June.

Ruflin N, Burkhart H and Rizzotti S (2011) Social-data storage-systems. In: *Proceedings of Databases and Social Networks (DBSocial '11)*. New York, NY, USA: ACM, pp. 7–12.

Sankaranarayanan J, Samet H, Teitler BE, et al. (2009) TwitterStand: news in tweets. In: *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems*. New York, NY, USA: ACM, pp. 42–51.

Shi Y, Meng X, Zhao J, et al. (2010) Benchmarking cloud-based data management systems. In: *Proceedings of the second international workshop on cloud data management*. New York, NY, USA: ACM, pp. 47–51.

Stewart RJ, Trinder PW and Loidl H-W (2011) Comparing high level mapreduce query languages. In: *Advanced*

Parallel Processing Technologies. Berlin, Heidelberg: Springer.

Twitter Inc (2013) *Documentation: Twitter Developers*. Available at: https://dev.twitter.com/overview/documentation (accessed 4 November 2014).

White T (2012) *Hadoop: The Definitive Guide*. Sebastopol, CA: O'Reilly.

Xia M, Huang R, Witt KL, et al. (2008) Compound cytotoxicity profiling using quantitative high-throughput screening. *Environmental Health Perspectives* 116(3): 284.