Healthcare Uses Cases with Apache Spark

Illinois Institute of Technology
Pooja Dusane
pdusane@hawk.iit.edu

ABSTRACT

In this paper we will we will explore and see how we can use Spark for ETL and descriptive analysis. Big Data aims to collect data from pre-treatment and pre-diagnosis data to the end stage. This data is aggregated with clinical and diagnostic data which makes predicting cancer more feasible. This predictive analysis helps to categorize different cancers and improves healthcare treatment. Many use cases in healthcare institutions are well suited for a big data solution. Some of the academic or research oriented healthcare institutions are either experimenting with big data or using it in advanced research projects. In healthcare industry, there is large volume of data that is being generated. Electronic Health Record (EHR) alone collects a huge amount of data. But apart from EHRs, there are various other sources of data in healthcare industry. Over the last decade, pharmaceutical companies have been aggregating years of research and development data into medical databases and because of this, the patient records have been digitized. In parallel, recent technical advances have made it easier to collect and analyze information from multiple sources which is a major benefit for health care institutions, since data for a single patient may come from various hospitals, laboratories, and physician offices.

INTRODUCTION

With a lot of medical data coming from various sources, guided decisions can be made from the insights gained through big data by using various Machine Learning Algorithms. Traditionally, physicians use their judgment while making treatment decisions, but in the last few years there has been a shift towards evidence-based medicine. This involves systematical review of clinical data and making treatment decisions based on the best available information. Aggregating individual data sets into big-data algorithms often provides the most robust evidence, since nuances in sub populations (such as the presence of patients with gluten allergies) may be so rare that they are not clear in small samples. Healthcare industry generates huge data about every patient but accessing, managing, and interpreting the data are critical to creating actionable insights for better care and efficiency.

HEALTHCARE DATA

The American Healthcare expenses represent 17.6% of its GDP, which is far more than the expected benchmark. Clinical trends also play a role in the rise of Big Data in Healthcare. Earlier physicians used their judgments to make treatment decisions, but the last few years have seen a shift in the way these decisions are being taken. Physicians review the clinical data and make an informed decision about a patient's treatment. Financial concerns, better insights into treatment, research, efficient practices contribute to the need of Big Data in Healthcare industry. As per prediction by IDC, 30 percent of providers will use cognitive analytics with patient data by 2018.

IOT adds a great value to the healthcare industry. Devices that generate data about a person's health and send it to the cloud will lead to a plethora of insights about an individual's heart rate, weight, blood pressure, lifestyle and much more. Big Data allows real-time monitoring of patients, which leads to proactive care. Sensors and wearable devices will collect patient health data even from home. Healthcare institutions monitor this data to provide remote health alerts and lifesaving insights to their patients.

Smartphones have added a new dimension. The apps enable the smartphone to be used as a calorie counter to keep a track of calories; pedometers to keep a check on how much you walk in a day. All these have helped people live a healthier lifestyle. Moreover, this data could be shared with a doctor, which will help towards personalized care and treatment. Patients can make lifestyle choices to remain healthy. Big Data helps predict the spread of epidemics. The mobile phone location data track population movements, which predict the spread of the virus. This gives insights about the most affected areas, which in turn leads to better planning of treatment centers and enforce movement restriction in those areas.

By leveraging historical data of patients with similar conditions, predictive algorithms can be developed using R and big data machine learning libraries to project patient trajectory. Clinical studies can be performed in a much efficient manner. Researchers who conduct clinical studies can take a variety of factors combined with multiple statistics to attain higher precision in their studies. Genomic data is very important for the healthcare industry. The values of diagnostic tests are vital to the reduction in lab testing and genome analysis costs.

Socioeconomic data can play a significant role in the predictive analysis. This data might show that people with certain zip code do not have access to cars (rural places) or other vehicles. Health systems thus identify patients in these areas and predict missed appointments, non-compliance with medications and more. The possibilities with predictive analysis with Big Data are endless. There are many benefits of Big Data in Healthcare for managing the hospital inventory. It averages the supplies per treatment enabling Just in Time Inventory which reduces cost.

Health care costs are driving the demand for big-data driven Healthcare applications. U.S. health care spending has outpaced GDP growth for the past several decades and exceeds spending in any other developed country. Despite being more expensive, according to the Organization for Economic Co-operation and Development (OECD), the US Health System ranks last among eleven countries on measures of access, equity, quality, efficiency, and healthy lives. Standards and incentives for the digitizing and sharing of healthcare data along with improvements and decreasing costs in storage and parallel processing on commodity hardware, are causing a big data revolution in health care with the goal of better care at lower cost.

Unstructured data forms about 80% of information in the healthcare industry and is growing exponentially. Getting access to this unstructured data—such as output from medical devices, doctor's notes, lab results, imaging reports, medical correspondence, clinical data, and financial data—is an invaluable resource for improving patient care and increasing efficiency.
Examples of healthcare data sources that will benefit from big data and analytics:
Claims: are the documents providers submit to insurance companies to get paid. A key component of the Health Insurance

Portability and Accountability Act (HIPAA) is the establishment of national standards for electronic healthcare transactions in order to improve efficiency by encouraging the widespread use of Electronic Document Interchange (EDI) between healthcare providers and insurance companies. Claim transactions include International Classification of Diseases (ICD) diagnostic codes, medications, dates, provider IDs, the cost.

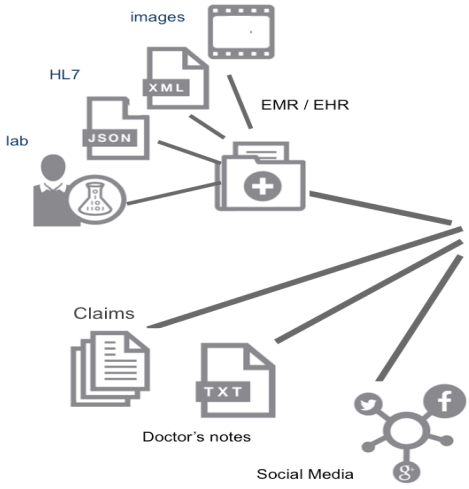
Electronic Health/Medical Record data (EHR or EMR): Medicare and Medicaid EHR incentive programs were established to encourage professionals and hospitals to adopt and demonstrate meaningful use of certified EHR technology. EHRs facilitate a comprehensive sharing of data with other providers and medical applications. EHRs contain the data from the delivery of healthcare which includes diagnosis, treatment, prescriptions, lab tests, and radiology. Health Level Seven International (HL7) provides standards for the exchange, integration, sharing, and retrieval of electronic health record data.

Pharmaceutical R&D, Clinical Trials Data, Genomic Data, Patient behavior and sentiment data, Medical Device Data, Patient sensor data from the home or hospital. There is a move toward evidence-based medicine, which involves making use of all clinical data available and factoring that into clinical and advanced analytics. Capturing and bringing all the information about a patient together gives a more complete view for insight into care coordination and outcomes-based reimbursement, population health management, and patient engagement and outreach.

The cost of fraud, waste and abuse in the healthcare industry is a key contributor to spiraling health care costs in the United States, but big data analytics can be a game changer for health care fraud. The Centers for Medicare and Medicaid Services prevented more than $210.7 million in healthcare fraud in one year using predictive analytics. UnitedHealthcare transitioned to a predictive modeling environment based on a Hadoop big data platform, in order to identify inaccurate claims in a systematic, repeatable way and generated a 2200% return on their big data/advanced technology. The key to identifying fraud is the ability to store and go back in history to analyze large unstructured datasets of historical claims and to use machine-learning algorithms to detect anomalies and patterns. Healthcare organizations can analyze patient records and billing to detect anomalies such as a hospital's overutilization of services in short time periods, patients receiving healthcare services from different hospitals in different locations simultaneously, or identical prescriptions for the same patient filled in multiple locations.

Initiatives such as meaningful use are accelerating the adoption of Electronic Health Records and the volume and detail of patient information is growing rapidly. Being able to combine and analyze a variety of structured and unstructured data across multiple data sources, aids in the accuracy of diagnosing patient conditions, matching treatments with outcomes, and predicting patients at risk for disease or readmission. Predictive modeling over data derived from EHRs is being used for early diagnosis and is reducing mortality rates from problems such as congestive heart failure and sepsis. Congestive Heart Failure (CHF) accounts for the most health care spending. The earlier it is diagnosed the better it can be treated avoiding expensive complications, but early manifestations can be easily missed by physicians. A machine learning example from Georgia Tech demonstrated that machine-learning algorithms could look at many more factors

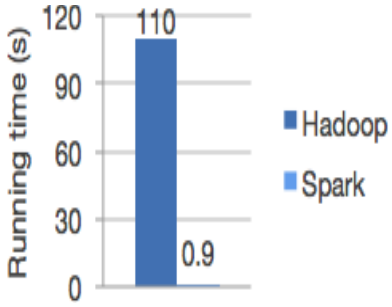
in patients' charts than doctors, and by adding additional features there was a substantial increase in the ability of the model to distinguish people who have CHF from people who don't.
Predictive modeling and machine learning on large sample sizes, with more patient data, can uncover nuances and patterns that couldn't be previously uncovered. Optum Labs has collected EHRs of over 30 million patients to create a database for predictive analytics tools that will help doctors make Big Data-informed decisions to improve patients' treatment.



FEATURES OF SPARK

Apache Spark has following features.

• Speed− Spark helps to run an application in Hadoop cluster, up to 100 times faster in memory, and 10 times faster when running on disk. This is possible by reducing number of read/write operations to disk. It stores the intermediate processing data in memory. It uses the concept of a Resilient Distributed Dataset (RDD), which allows it to transparently store data on memory and persist it to disc only it's needed. This helps to reduce most of the disc read and write – the main time consuming factors – of data processing.



• Ease of Use – Spark lets you quickly write applications in Java, Scala, or Python. This helps developers to create and run their applications on their familiar programming languages and easy to build parallel apps. It comes with a built-in set of over 80 high-level operators. We can use it interactively to query data within the shell too. Word count in Spark's Python API datafile = spark.textFile("hdfs://...")
datafile.flatMap(lambda line: line.split())
.map(lambda word: (word, 1))
.reduceByKey(lambda a, b: a+b)

• Combines SQL, streaming, and complex analytics – In addition to simple "map" and "reduce" operations, Spark supports SQL queries, streaming data, and complex analytics such as machine learning and graph algorithms out-of-the-box. Not only that, users can combine all these capabilities seamlessly in a single workflow.

• Supports multiple languages − Spark provides built-in APIs in Java, Scala, or Python. Therefore, you can write applications in different languages. Spark comes up with 80 high-level operators for interactive querying.

• Advanced Analytics − Spark not only supports 'Map' and 'reduce'. It also supports SQL queries, Streaming data, Machine learning (ML), and Graph algorithms.

• Runs Everywhere − Spark runs on Hadoop, Mesos, standalone, or in the cloud. It can access diverse data sources including HDFS, Cassandra, HBase etc.

APACHE SPARK USECASES

Apache Spark is the new shiny big data bauble making fame and gaining mainstream presence amongst its customers. Startups to Fortune 500s are adopting Apache Spark to build, scale and innovate their big data applications. Banks are using the Hadoop alternative - Spark to access and analyse the social media profiles, call recordings, complaint logs, emails, forum discussions, etc. to gain insights which can help them make right business decisions for credit risk assessment, targeted advertising, and customer segmentation. One of the financial institutions that has retail banking and brokerage operations is using Apache Spark to reduce its customer churn by 25%. The financial institution has divided the platforms between retail, banking, trading, and investment. Information about real time transaction can be passed to streaming clustering algorithms like alternating least squares (collaborative filtering algorithm) or K-means clustering algorithm. The results can be combined with data from other sources like social media profiles, product reviews on forums, customer comments, etc. to enhance the recommendations to customers based on new trends. As healthcare providers look for novel ways to enhance the quality of healthcare, Apache Spark is slowly becoming the heartbeat of many healthcare applications. Many healthcare providers are using Apache Spark to analyse patient records along with past clinical data to identify which patients are likely to face health issues after being discharged from the clinic. This helps hospitals prevent hospital re-admittance as they can deploy home healthcare services to the identified patient, saving on costs for both the hospitals and patients.

Apache Spark is used in genomic sequencing to reduce the time needed to process genome data. Earlier, it took several weeks to organize all the chemical compounds with genes but now with Apache spark on Hadoop it just takes few hours. This use case of spark might not be so real-time like other but renders considerable benefits to researchers over earlier implementation for genomic sequencing.

Apache Spark is used in the gaming industry to identify patterns from the real-time in-game events and respond to them to harvest lucrative business opportunities like targeted advertising, auto adjustment of gaming levels based on complexity, player retention and many more.

Few of the video sharing websites use apache spark along with MongoDB to show relevant advertisements to its users based on the videos they view, share and browse.

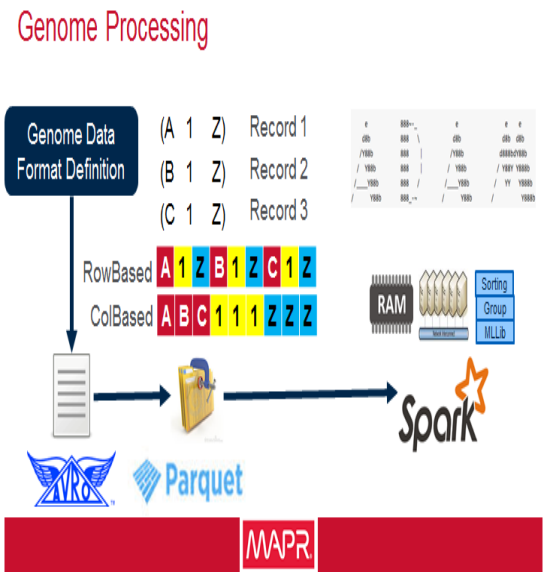
Yahoo uses Apache Spark for personalizing its news webpages and for targeted advertising. It uses machine learning algorithms that run on Apache Spark to find out what kind of news - users are interested to read and categorizing the news stories to find out what kind of users would be interested in reading each category of news.

Earlier the machine learning algorithm for news personalization required 15000 lines of C++ code but now with Spark Scala the machine learning algorithm for news personalization has just 120 lines of Scala programming code. The algorithm was ready for production use in just 30 minutes of training, on a hundred million datasets.

The largest health and fitness community MyFitnessPal helps people achieve a healthy lifestyle through better diet and exercise. MyFitnessPal uses apache spark to clean the data entered by users with the end goal of identifying high quality food items. Using Spark, MyFitnessPal has been able to scan through food calorie data of about 80 million users. Earlier, MyFitnessPal used Hadoop to process

2.5TB of data and that took several days to identify any errors or missing information in it

The Novartis team chose Hadoop and Apache Spark to build a workflow system that allows them to integrate, process and analyze diverse data for Next Generation Sequencing (NGS) research while being responsive to advances in the scientific literature.



Combined with Spark's distributed and massively parallel computing capabilities, this allows the processing of datasets that are beyond the capabilities of a standalone R program. An open source project called ADAM is using Apache Spark for scalable genomics data processing. Some of the incoming clinical data likely meet the Big Data criteria of volume, velocity, and variety. This is particularly true for physiological time series from wearable sensors. Apache Spark's cluster computing capabilities provide the scalability required to process these large volumes of data. Spark can be used to implement the Lambda Architecture enabling both batch and real-time data stream processing and mining. Functional Programming languages like Scala have several benefits that are particularly important for applying Machine Learning algorithms on large data sets. Like functions in mathematics, functions in Scala have no side effects and can be composed as well. This provides referential transparency which facilitates reasoning about the behavior of application logic. Machine Learning algorithms are in fact based on Linear Algebra and Calculus. Scala supports high-order functions as well. Variables are immutable witch greatly simplifies concurrency. For all those reasons, Machine Learning libraries like Apache Mahout have embraced Scala, moving away from the Java-based MapReduce paradigm. Apache Spark is itself written in Scala and provides a REPL-based interactive shell for data analysis. Spark also uses a Scala-based reactive framework called Akka. To make the most of the latest NGS research, they needed workflow tools that were robust enough to process vast amounts of raw data, yet flexible enough to keep up with quickly changing research techniques. Although NGS data requires high data volumes that are ideal for Hadoop, a common problem is that researchers rely on many tools that don’t work on native HDFS. Since these researchers previously couldn’t use systems like Hadoop, they have had to maintain complicated "bookkeeping" logic to parallelize for optimum efficiency on traditional High-Performance Computing (HPC). This workflow system uses Hadoop for its performance and robustness and to provide the POSIX file access that lets bioinformaticians use their familiar tools. Additionally, it uses the researchers’ own metadata to allow them to write complex workflows that blend the best aspects of Hadoop and traditional HPC. The team then uses Apache Spark to integrate the highly diverse datasets. Their unique approach to

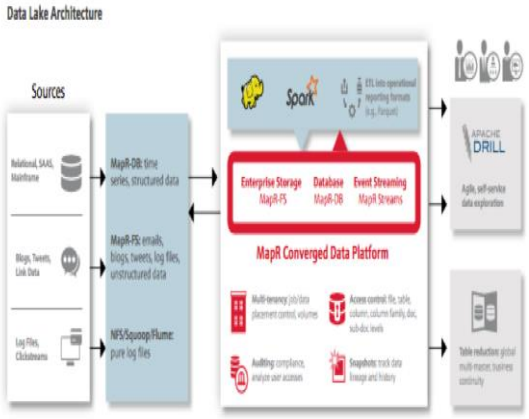
dealing with heterogeneity was to represent the data as a vast knowledge graph (currently trillions of edges) that is stored in HDFS and manipulated with custom Spark code. This innovative use of a knowledge graph lets Novartis bioinformaticians easily model the complex and changing ways that biological datasets connect to one another, while the use of Spark allows them to perform graph manipulations reliably and at scale.

On the analytics side, researchers can access data directly through a Spark API, or through a number of endpoint databases with schemas tailored to their specific analytic needs. Their tool chain allows entire schemas with 100 billions of rows to be created quickly from the knowledge graph and then imported into the analyst's favorite database technologies.

VALENCE HEALTHCARE

Valence Health is using the MapR Converged Data Platform to build a data lake that is the company's main data repository. Valence consumes 3,000 inbound data feeds, with 45 different types of data, daily. This critical data includes lab test results, patient health records, prescriptions, immunizations, pharmacy benefits, claims and payments, and claims from doctors and hospitals, which are used to inform decisions about improving both healthcare outcomes and reimbursement. The company's rapid client growth and the associated increasing volumes of data were straining its existing technology infrastructure.

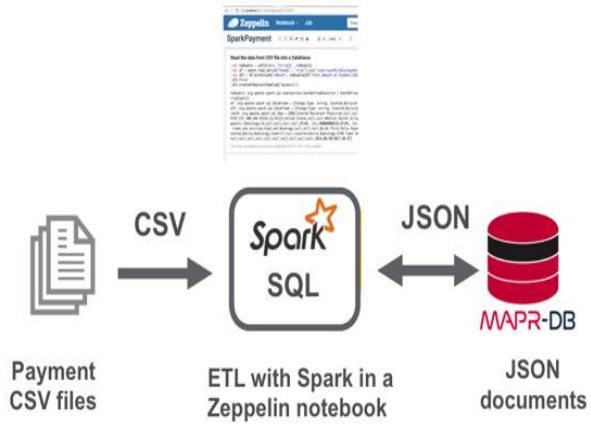
Prior to their MapR solution, if they received a feed with 20 million lab records, it would take 22 hours to process that data. MapR cut that cycle time down from 22 hours to 20 minutes, running on much less hardware. Valence Health is also now able to accommodate customer requests that were very difficult to address in the past. For example, a customer might call and say, "I sent you an incorrect file three months ago, and I need you to take that file out." Their traditional database solution might take 3-4 weeks to get that data deleted. MapR snapshots provide point-in-time recovery that enables Valence to roll back and remove that file in minutes.

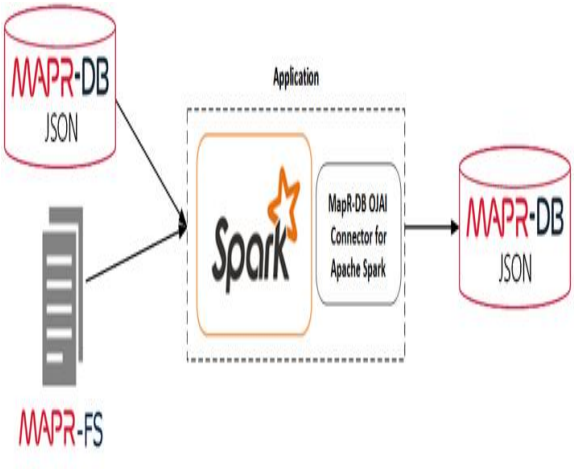


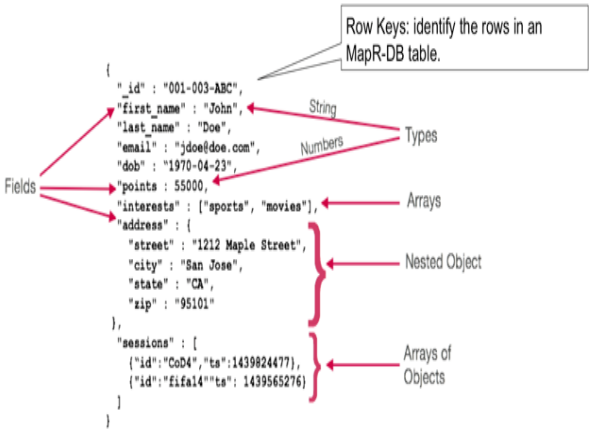
SPARK SQL WITH HEALTHCARE

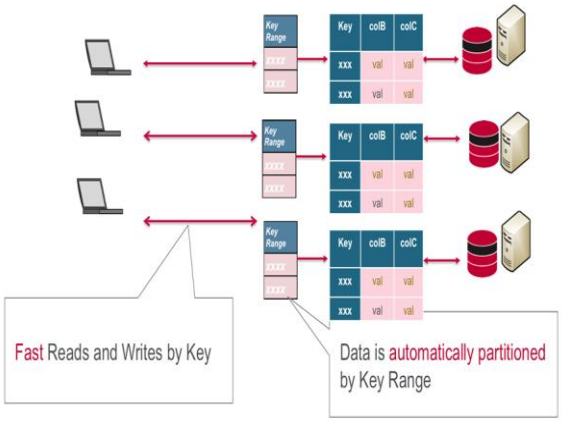
A large health payment dataset, JSON, Apache Spark, and MapR-DB are an interesting combination for a health analytics workshop because JSON is an open-standard and efficient format that uses human-readable text to represent, transmit, and interpret data objects consisting of attribute-value pairs. Because JSON is easy for computer languages to manipulate, JSON has supplanted XML for web and mobile applications.

Newer standards for exchanging healthcare information such as FHIR are easier to implement because they use a modern web-based suite of API technology, including REST and JSON. Apache Spark SQL, DataFrames, and datasets make it easy to load, process, transform, and analyze JSON data. MapR-DB, a high-performance NoSQL database, supports JSON documents as a native data store. MapR-DB makes it easy to store, query, and build applications with JSON documents.







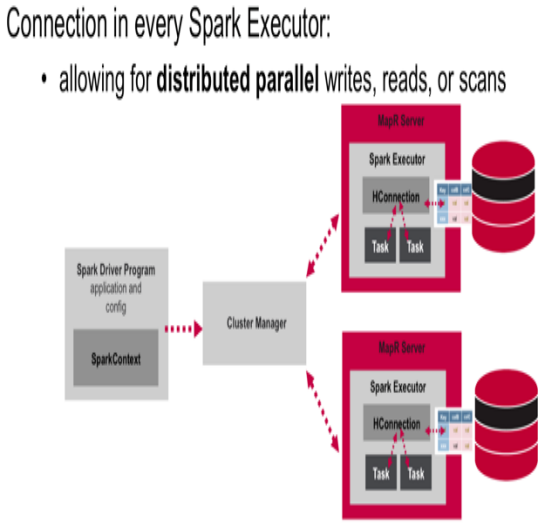


APACHE SPARK AND MapR DB

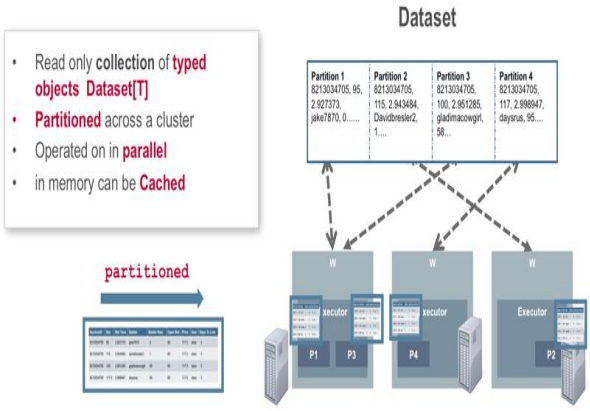
One of the challenges that comes up when you are processing lots of data is where you want to store it. With MapR-DB (HBase API or JSON API), a table is automatically partitioned into tablets across a cluster by key range, providing for scalable and fast reads and writes by row key.

The MapR-DB OJAI Connector for Apache Spark makes it easier to build real-time or batch pipelines between your JSON data and MapR-DB and leverage Spark within the pipeline. Included is a set of APIs that that enable MapR users to write applications that consume MapR-DB JSON tables and use them in Spark.

The Spark MapR-DB Connector leverages the Spark DataSource API. The connector architecture has a connection object in every Spark Executor, allowing for distributed parallel writes, reads, or scans with MapR-DB tablets.



A Spark dataset is a distributed collection of data. Dataset is a newer interface, which provides the benefits of the older RDD interface (strong typing, ability to use powerful lambda functions) combined with the benefits of Spark SQL's optimized execution engine. Datasets also provide faster performance than RDDs with more efficient object serialization and deserialization.



A DataFrame is a dataset organized into named columns Dataset[Row]. (In Spark 2.0, the DataFrame APIs merged with Datasets APIs.)

Read the Data From a CSV File Into a Dataframe

The SparkSession read method loads a CSV file and returns the result as a DataFrame.

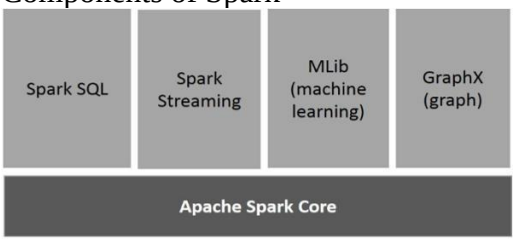
A user-defined method is used to convert the amount column from a string to a double. A local temporary view is created to easily use SQL. Next, we want to select only the fields that we are interested in and transform them into a Dataset of payment objects. First, we define the payment object schema with a Scala case class. Next, we use Spark SQL to select the fields we want from the DataFrame and convert this to a Dataset[Payment] by providing the Payment class. Then, we replace the Payment view.

PATIENT CARE ANALYSIS WITH SPARK

HealthCare Problem statement: Health Care Industry desires to classify the Patients using their pathology data for their CARE (Self-management, Doctor-Advise, Further Diagnostic and Chronic Medication) management improvement that facilitates to build a multi-classification model to build CARE Management Model (CMM) with right classification of patient.

Tools Used: Spark Cluster Spark MLLIB (Machine Learning Library) R & Python Language

Components of Spark



Apache Spark: Apache Spark is a lightning-fast cluster computing technology, designed for fast computation. It is based on Hadoop MapReduce and it extends the MapReduce model to efficiently use it for more types of computations, which includes interactive queries and stream processing. The main feature of Spark is its in-memory cluster computing that increases the processing speed of an application. Spark is designed to cover a wide range of workloads such as batch applications, iterative algorithms,

interactive queries and streaming. Apart from supporting all these workload in a respective system, it reduces the management burden of maintaining separate tools.
Spark SQL: Spark SQL is a component on top of Spark Core that introduces a new data abstraction called Schema RDD, which provides support for structured and semi-structured data. Spark Streaming Spark Streaming leverages Spark Core's fast scheduling capability to perform streaming analytics. It ingests data in mini-batches and performs RDD (Resilient Distributed Datasets) transformations on those mini-batches of data.
MLlib (Machine Learning Library): MLlib is a distributed machine learning framework above Spark because of the distributed memory based Spark architecture. It is, according to benchmarks, done by the MLlib developers against the Alternating Least Squares (ALS) implementations. Spark MLlib is nine times as fast as the Hadoop disk-based version of Apache Mahout(before Mahout gained a Spark interface). GraphX GraphX is a distributed graph-processing framework on top of Spark. It provides an API forexpressing graph computation that can model the userdefined graphs by using Pregel abstraction.
Python: Python is a dynamic, interpreted (bytecode-compiled) language. There are no type declarations of variables, parameters, functions, or methods in source code. This makes the code short and flexible, and you lose the compiletime type checking of the source code. Universally, Python has gained a reputation because of it’s easy to learn. The syntax of Python programming language is designed to be easily readable. Python has significant popularity in scientific computing. The people working in this field are scientists first, and programmers second. Nowadays working on bulk amount of data, popularly known as big data. The more data you have to process, the more important it becomes to manage the memory you use. Here Python will work very efficiently. Scala If you're the kind of data scientist who deals with large datasets, Scala will be invaluable. It's practically the de facto language for the current Big Data tools like Apache Spark, Finagle, Scalding, etc. Many of the high performance data science frameworks that are built on top of Hadoop usually are written and use Scala or Java. The reason Scala is used in these environments is because of its amazing concurrency support, which is key in parallelizing a lot of the processing needed for large data sets. It also runs on the JVM, which makes it almost a no-brainer when paired with Hadoop. Like Java, Scala is objectoriented, and uses a curly-brace syntax reminiscent of the C programming language. Unlike Java, Scala has many features of functional programming languages like Scheme, Standard ML and Haskell, including currying, type inference, immutability, lazy evaluation, and pattern matching.
Problem Understanding: This study is commissioned with the following objectives: The care management model system gives computer the ability to learn without being explicitly programmed. (Machine Learning) To classify patients based on different criteria. The Care management models will facilities for 1) Personalized Medicine 2) Predictive Analytics and Preventive Measures 3) Self-Motivated Care 4) Disease Modelling and Mapping. Concluding with Big data is the reality and is going to stay there for a long time. While have a care management model by any hospital they can predict the future condition of any patient without any wait. Time is a major problem for health care. Within the time anything may happens. So life is saved by care management model. So the model created

by this project is so important in the everyday life. But the project model wants to more accurate. The only way to improve the accuracy of this model is collect more data and applies it for modelling. So in future hospital have to concentrate on big data preparation and management in order to create a good model for the system. The project suggest to gain success hospital need to change service manner and increase service on patients who are professionally unbalanced and try to efficiently utilise care management model for patients, serious and quick action needed when pathology values get changing abnormally.

SPARK WITH FUNCTIONAL MRI

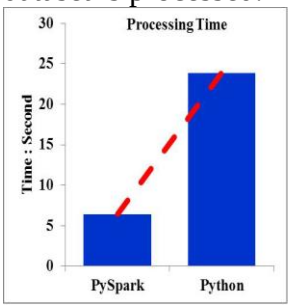
Spark Big data technologies and tools such as Hadoop or Apache. Hadoop are open-source software programming platform and projects for reliable, scalable, distributed computing [hadoop.apache.org]. The Apache Hadoop software library is a framework installed on specific hardware infrastructure enables developers and users for the distributed and parallel processing of large data sets across clusters and nodes by using programming models. In theory, Hadoop is designed to function in single servers or thousands of nodes performing local computation and storage [hadoop.apache.org]. This platform included several subprojects such as HDFS, MapReduce, YARN and etc. However, some of the difficulties in using Hadoop framework such as complicated installation process and highly dependency on hardware structure were motivations to develop other Big Data platforms. Also, Hadoop supported few programming languages and it is not user friendly for data analysts not having programming background. Therefore, Big Data platform developers aimed to develop a software library supporting more programming languages, having less dependency on hardware and being more memory efficient. Spark and Apache Spark is one of the modern big data platforms which was developed originally at the University of California, Berkeley's AMPLab. Apache Spark is practically a fast and general engine for big data processing. In memory data processing of Spark improved the performance up to 10 times faster than on-disk data processing. In addition, Spark offers over 80 high-level applications that can interactive with different programming languages such as Java, Scala, R and Python (which is more important in this paper). This feature shows the Spark ease of use compared to Hadoop platform. Another feature of Spark is Generality. Spark can easily handle data streaming and real-time data processing. Also, it can easily interact with SQL databases and data frames. Machine learning library of Spark (MLlib) and GraphX which is the graph analysis tool of this plat form allow users to perform data processing and analysis in a fast and parallelized environment that can be installed either on single node as a standalone version or installed on thousands of nodes.

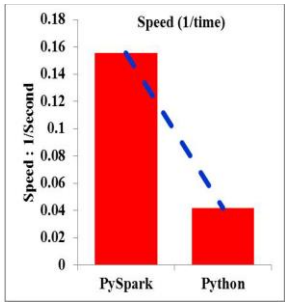
Functional MRI - Brain Networks

Functional Magnetic Resonance Imaging (fMRI) is a technique that measures brain activity by detecting the associated changes in blood flow. This MRI technique uses the change in magnetization between oxygen-rich and oxygenpoor blood in the brain as its primary outcome measure, with greater consumption of oxygen corresponding to greater neural recruitment within the brain. Our brain is an efficient network to be precise. It is a network made up of a large number of brain regions that have their own task and function but remain highly interactive by continuously sharing information with each other. As such, they form a complex integrative system in which information is continuously processed and transferred between structurally and

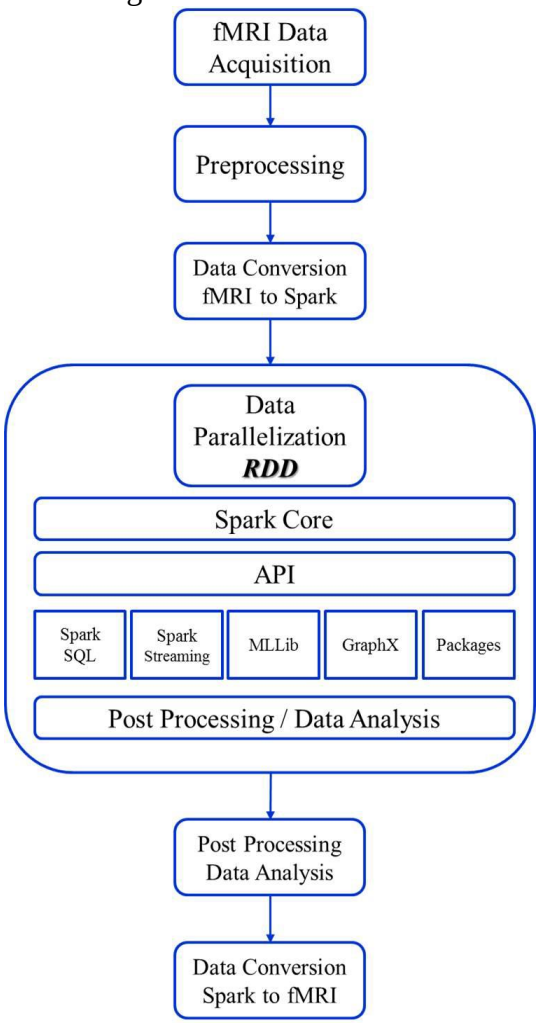
functionally linked brain regions: the brain network. The data which are collected in fMRI modality are four dimensional (4-D) as volumes images are acquired across time. The preprocessing and analysis of this huge volume of data is always time consuming and costly required parallelized infrastructure. Therefore, the fMRI data analysis deals with at least 2 Vs of big data analytics: Volume and Velocity. In small to large fMRI datasets, over Giga to Tera bytes of data are preprocessed and analyzed. Also, every day, imaging research and healthcare centers collectfMRI data acquisition 3D brain volume across time more and more data and archiving this volume of data has become challenging. In addition, data format is another issue causing some restrictions to use big data platform in imaging. None of imaging data formats can be directly stored in database and be preprocessed or analyzed by standard big data analytics tools. To merge big data analytics and medical imaging, we proposed and developed a pipeline that can be used on single node PC or huge clusters and is able to process data much faster the current methods and enables users to store the pre/post processed data in different data format compatible with big data plat forms especially Spark. Briefly, in the past studies, 7 males and 9 females with a mean age of 21.1 2.2 years were recruited and structural and functional MRI data were collected. The standard fMRI preprocessing steps were applied to raw data using FMRIB Software Library v5.0. The goal of that study was to extract the brain networks (especially Default Mode Network) from independent components of brain imaging data. Probabilistic Independent Components of the preprocessed data were calculated by FSL-MELODIC resulted in 84 components. Next, using our template matching algorithm the DMN was reconstructed from probabilistic independent components. In our brain network extractor and decision-making algorithm, different methods such as normalized cross correlation, sum of squared differenced and dice coefficient. In our current study, we only used sum of squared error (SSD) in order to test our Spark solution. In this work, we used PySpark Standalone version (http://spark.apache.org/) on single node to test and explore the potential application of our proposed pipeline. Those 84 brain components which were in Neuroimaging Informatics Technology Initiative (Nifti) format (standard format for NeuroImgaing data) were loaded into memory using Nibable package (http://nipy.org/nibabel) providing interfaces for neuroimaging data manipulation in Python. Next, the data in memory were converted to Resilient Distributed Datasets (RDD) format. RDD is a fundamental data structure of Spark. It is an immutable distributed collection of objects and ach dataset in RDD is divided into logical partitions, which may be computed on different nodes of the cluster. Formally, an RDD is a readonly, partitioned collection of records. RDDs can be created through deterministic operations on either data on stable storage or other RDDs. RDD is a fault-tolerant collection of elements that can be operated on in parallel. As mentioned above, we were to extract the default mode network from the components. Therefore, we used the DMN template developed by our team. The template was also loaded and converted to RDD. SSD between 84 components and DMN templated were calculated using following equation (1) in PySpark after flatmapping and zipping RDDs. $SSD = X\ x,y\ [f(x, y) - t(x - u, y - v)]2 \ldots (1)$ The performance was measured in PySpark which was equal to 6.43799 seconds. The same experiment was repeated in Python and the performance was measured and it was equal to 23.86625. This testing revealed that using PySpark big data platform even on single node runs the data

faster than pure Python around 4 times (exactly 3.7) in our case. In addition, if the number of images increases the difference between PySpark and Python performance will increase as well. The Fig. 4 compares the measured performance between Python and PySpark which ran against our template matching script. Around 4 times speed up the running time is promising and we argue that our program was executed on a single node and was not completely designed for parallel computing and big data platform. In other words, if we develop our serial template matching algorithm for PySpark environment and parallel processing and also use high performance cluster instead of standalone - single node version of Spark, the performance will potentially improve up to 10 to 20 times especially when a huge dataset is processed.





The processing time and speed comparison shows PySpark-based pipeline performs faster table comparison between pyspark and python in brain extraction application Platform Time (second) 1/Time PySpark 6.437999964 0.15533 Python 23.86625409 0.0419 IV. We developed and successfully tested our new PySparkbased pipeline on a single node to analyze functional MRI. Using our algorithm described in, we reconstructed the brain networks from independent components PySpark-based pipeline for fMRI data processing and analysis data for extracting brain networks. The new pipeline improved the processing time around 4 times faster than previous works while the accuracy remained at the same value. Furthermore, ease of use, in-memory data processing and storing results in different data structure are some important features of this pipeline. Also, this pipeline can easily expand to several nodes and high-performance computing clusters for massive data analysis on large datasets which will improve the processing time and the performance of the pipeline much more than a single node.



MEDICAL IMAGE PROCESSING

Imaging experiments involving complex specimens like full-animals, vascular structures in brain, or cellular material rheology are difficult or impossible to accurately characterize by eye and thus require computationally intensive algorithms to extract meaningful quantitative information from them. With improvements in flux, detector efficiency, and reconstruction algorithms, the rate at which image data is produced using Synchrotron-based X-ray Tomographic Microscopy is staggering. At the TOMCAT beamline of the Swiss Light Source, this rate reaches 8GB per second of image data[1], higher than even multinational companies with dedicated IT staff like Facebook or Instagram usually handle. We have developed a scalable framework based on Apache Spark and the Resilient Distributed Datasets proposed in [2] for parallel, distributed, real-time image processing and quantitative analysis. The cluster-/cloud-based evaluation tool performs filtering, segmentation and shape analysis enabling data exploration and hypothesis testing over millions of structures with the time frame of an experiment. The tools have been tested with clusters containing thousands of machines and images containing more than 100 billion voxels. The flexible infrastructure offers a full spectrum of shape, distribution, and connectivity metrics for cellular materials and networks and can be adapted to a wide variety of new studies requiring high sample counts ranging from long-term dynamics and evolution experiments to investigations of drug-gene interactions. Finally we examine the potential for real-time approximate analysis as compared with region of interest selection and down-sampling as a superior approach for speeding up post-processing

HIDDEN MARKOV MODELS AND SPARK TO MINE ECG DATA

The in-progress research explores the applicability of Hidden Markov Models on ECG readings, with the goal of detecting the emerging factors. Here we note the strategy for constructing our system. We obtained ECG signals from the QT Database (QTDB), using the WaveForm Database application suite. We also obtained two sets of annotations: one, marked atr, contains annotations that marks beats as normal, or as having some abnormality (preventricular contraction, for instance); the second set of annotations, marked pu0, contains waveform markers, such as p, t, and N (for normal qrs complex). Any records from the QTDB that did not contain annotations from atr were excluded, as we would not be able to verify our results against them. We transformed the pu0 annotations to provide clearer information. The standard for annotating waves is to open a wave with a paren, note the wave, and then close it with a paren. For instance, the p wave would be marked by the annotations (, p,). We wrote a script to process these annotations, and change them to the form pBegin, p, pEnd, so that all parenthesis were removed. This meant that the annotations themselves could now become a set of states for use in a Hidden Markov Model. The states derived from the annotations were: pBegin, p, pEnd, q, r, q, tBegin, t, tEnd, unknownBegin, and unknownEnd. However, we found that it was not practical to simply map the states annotated in pu0 to the beat classifications annotated in atr. When attempting to map the state sequence to PVC, for instance, no significant correlation could be found in a sample of PVC beats. We hypothesized that the duration of the states was also significant. It may be necessary to mark states as being faster or slower than normal. The duration between, for instance, pBegin and pEnd could tell us if the p wave were of

normal duration. With this in mind, we are determining a way to map the ECG signal itself to states. In [10], we find an algorithm for decomposing ECG signals into line segments. This algorithm moves a dynamically-sized window along the ECG signal. The window checks the distance between the endpoints and every point in-between, using normalized distances where needed. We can adjust the allowed error to accomodate noisy signals. We modify this algorithm to output a list of 4-tuples of the form (starting point, length, mean of segment, standard deviation of segment). This converts the continuous ECG signal into a set of data points. We must then convert this set of data points into states that correspond with the waveforms of the heart beat: the p wave, qrs complex, t wave, and the intervals between them.

TARGET PREDICTION IN DRUG DISCOVERY

The trajectory from a biological concept to a drug available to patients is expensive and typically spans over a decade. Drug discovery starts by mapping a disease to a scalable experiment in a test tube. This enables screening of libraries of chemicals for active compounds (hits), from which chemical starting points (leads) are selected. These leads are then taken through multiple follow-up assays in cellular systems and later animal models to optimize their potency on the intended protein targets implicated in disease, while controlling their activity on undesired targets associated with side effects. After the final step of fine-tuning delivery and dosing in test animals, the compound is transferred to drug development, where it is tested on human subjects in phases. The first phase assesses drug safety in healthy volunteers over the dose range considered. The second aims to establish proof-of-concept, ie. a tangible clinical benefit for patients. Finally, in the third phase, a drug label is defined, which specifies the exact disease indication, ie. patients eligible for treatment, and an optimized dosage regimen, and then quantifies the benefit in comparison to the pre-existing standard-of-care treatment. If approved, the drug label is disseminated to physicians. However, the vast majority of compounds that enter drug development do not make it through to approval. To increase the odds of seeing the development of a drug through to the end, the drug discovery phase must identify compounds that score well on efficacy ("do they affect the desired protein?") and selectivity ("do they only affect the desired protein?"). A study performed on drug failures [1] revealed that in 2011–2012, 56% of failures in testing phase two or three were due to lack of efficacy and a further 28% due to safety issues. One way of determining these properties is by applying the compound to several kinds of cells in a lab environment and seeing whether they have the desired effects. This is a slow and costly process, however, especially as one protein target might be affected by thousands of compounds, and one compound might affect multiple targets. An alternative is virtual screening: the identification of candidate compounds through computation [2]–[5]. This method takes advantage of the fact that compounds with a similar structure often interact similarly with the same proteins. Inside Janssen Pharmaceutica, the Chemogenomics project attempts to identify candidate compounds by deriving information from existing compound–protein databases by means of machine learning methods [3], [6], [7]. To this end, a number of "predictor" programs have been developed. These programs construct a model by training on known entries in the compounds–targets activation matrix. This model is then used to predict activation probabilities for unknown compounds–

targets combinations. For every predictor a number of parameters can be tuned, such as the distance and similarity metrics, certain thresholds, etc. Additionally, there exist different methods for generating fingerprints of compounds, which has an impact on the quality of the model learned [8]. In the search for the best model, this parameter space needs to be explored. Additionally, to prevent overfitting [9], cross-validation is used. This is a well-known technique where a part of the training data is withheld and used to gauge the accuracy of the trained model. The predictors are implemented as a set of separate multithreaded programs in C++, written using Boost and Intel Threading Building Blocks (TBB; [10]). Every predictor consists of a training stage, in which the program trains itself on the data in the training set, and a prediction stage where activations are predicted for the compounds. The prediction stage is implemented as a parallel pipeline in TBB, which enables the program to make use of all cores in a node. Unfortunately, the programs cannot easily be adapted to run on multiple nodes and business efforts are focused on improving the quality of prediction, not the speed

CONCLUSION

Over time, Apache Spark will continue to develop its own ecosystem, becoming even more versatile than before. In a world where big data has become the norm, organizations will need to find the best way to utilize it. As seen from these Apache Spark use cases, there will be many opportunities in the coming years to see how powerful Spark truly is. As increasingly organizations recognize the benefits of moving from batch processing to real time data analysis, Apache Spark is positioned to experience wide and rapid adoption across a vast array of industries. Big-data initiatives have the potential to transform health care. Stakeholders that are committed to innovation, willing to build their capabilities, and open to a new view of value will likely be the first to reap the rewards of big data and help patients achieve better outcomes.

REFERENCES

- https://mapr.com/blog/reduce-costs-and-improve-health-care-with-big-data/
- https://www.slideshare.net/dominodatalab/data-science-popup-seattle-using-spark-in-healthcare-predictive-analytics-in-the-or
- https://acadgild.com/blog/healthcare-use-case-apache-spark/
- https://cris.vub.be/files/5147244/spark.pdf
- https://www.mckinsey.com/industries/healthcare-systems-and-services/our-insights/the-big-data-revolution-in-us-health-care
- https://arxiv.org/abs/1603.07064
- https://data-flair.training/blogs/big-data-healthcare-real-world-use-cases/
- https://www.linkedin.com/pulse/advanced-real-time-healthcare-analytics-apache-spark-joel-amoussou/