LEVERAGING BIG DATA FOR MANAGING TRANSPORT OPERATIONS

Deliverable 1.3 **Big Data Methodologies, Tools and Infrastructures**

Kim Hee[1], Todor Ivanov[1], Roberto V. Zicari[1], Rut Waldenfels[1], Hevin Özmen[1], Naveed Mushtaq[1], Minsung Hong[2], Tharsis Teoh[3], Rajendra Akerkar[2]

Goethe-University Frankfurt[1],
Western Norway Research Institute[2],
Panteia B.V.[3]

July 2018

Work Package 1

Project Coordinator

Prof. Dr. Rajendra Akerkar (Western Norway Research Institute)

Horizon 2020 Research and Innovation Programme

MG-8-2-2017 - Big data in Transport: Research opportunities, challenges and limitations

Distribution level	Public (P)
Due date	30/06/2018 (M8)
Sent to coordinator	10/07/2018
No. of document	D1.3
Title	*Big Data Methodologies, Tools and Infrastructures*
Status & Version	*Final*
Work Package	*1: Setting the stage on big data in transport*
Related Deliverables	*D1.1*
Leading Partner	*Goethe-University Frankfurt*
Leading Authors	*Kim Hee, GUF (Chapter 1, 2, 4, 6)* *Todor Ivanov, GUF (Chapter 1, 3, 5, 6)* *Roberto V. Zicari, GUF (Chapter 1, 7)* *Rut Waldenfels, GUF (Chapter 2)*
Contributors	*Naveed Mushtaq, GUF* *Hevin Özmen, GUF* *Minsung Hong, WNRI* *Tharsis Teoh, Panteia* *Rajendra Akerkar, WNRI*
Reviewers	*Gerben Zwart, Panteia*
Keywords	Big Data, Transportation, Technologies, Benchmark

Executive summary

Big Data opens up new opportunities to define "Intelligent" mobility and transportation solutions. The transportation industry is a leader in creating the so-called *Internet of Everything*. Each day vast volumes of data are generated through sensors in passenger counting and vehicle locator systems and ticketing and fare collection systems, just to name a few.

The goal is to create value out of this amount of data, by providing a comprehensive picture of what's happening, using business analytics, leveraging big data tools and predictive analytics, to help transportation agencies improve operations, reduce costs and hopefully better serve travelers.

The technical challenge is that much of this Big Data is non-standard data (e.g., social, geospatial or sensor-generated data that does not an easy fit into traditional, structured, relational data warehouses or databases).

An additional challenge is that with such an amount of real-time structured and unstructured data captured from a variety of sources, it is difficult to determine which data is most valuable. Terabytes of data are collected and result in an added complexity to the underlying IT infrastructures.

These terabytes of data require immense amounts of storage in silo after silo of transportation operator data centers. In order to analyze Big Data, an appropriate Data Infrastructure needs to be in place to:

1. store and maintain data
2. analyze data
3. present results in a clear visual way

Several Big Data platforms have been proposed recently, open source and proprietary. In order to tackle the demands and challenges in the transportation domain, an optimal stack of Big Data technologies needs to be selected and designed based on the application requirements.

This is not an easy task.

This report, which is a follow up of Deliverable 1.1, offers an in-depth introduction to relevant technologies for Big Data Analytics and Big Data Management. It also looks at how these technologies are applied to build a Big Data Platform suitable for the transport sector. We present in detail how application-specific benchmarking can be used in order to evaluate which Big Data technologies are most suited for the domain. We conclude the report with an applied example of using data analytics for urban mobility.

This document offers the reader a technical insight into existing Big Data technologies at various levels: software management, data platform, and application. In order to evaluate which specific software components in the Big Data stack are more suitable for transport applications, with high volume and high-velocity requirements, a benchmarking approach is presented.

The future of data analytics in transportation has many applications and opportunities.

The main challenge is using significantly improved technologies and methods to gather and understand the data in order for business decisions to be informed by better insights.

Table of contents

List of Figures

List of Tables

Glossary

Abbreviation	Expression
ACID	Atomicity, Consistency, Isolation, and Durability
API	Application Programming Interface
APS	Advanced Planning and Scheduling
BDGS	Big Data Generator Suite
BI	Business Intelligence
CPU	Central Processing Unit
CRISP-DM	Cross-industry Standard Process for Data Mining
CVST	Connected Vehicles and Smart Transportation
D1.1	LeMO Deliverable 1.1
D1.2	LeMO Deliverable 1.2
DBMS	Database Management System
DS	Database
DSPS	Distributed Stream Processing Systems
ETL	Extract Transform Load
FPGA	Field Programmable Gate Array
FPMR	MapReduce framework on FPGA
GFS	Google File System
GPU	Graphics Processing Unit
GWPG	Workstation Performance Group
HDFS	Hadoop Distributed File System
HPG	High-Performance Group
HVE	Hadoop Virtual Extension
I/O	Input/Output
ITS	Intelligent Transportation Systems
KDD	Knowledge Discovery in Databases
Lat	Latitude
Lon	Longitude
MPP	Massively Parallel Processing
MRBS	MapReduce Benchmark Suite
NYC	New York City
OLAP	Online Analytical Processing
OLTP	Online Transaction Processing
OS	Operating System
OSG	Open Systems Group
RDBMS	Relational Database Management System
RDD	Resilient Distributed Datasets
REEF	Retainable Evaluator Execution Framework
RFID	Radio-Frequency Identification
RIoTBench	Real-Time IoT Benchmark Suite
SDMS	Stream Data Management Systems

SEMMA	Sample, Explore, Modify, Model, and Assess
SF	Scale Factor
SPEC	Standard Performance Evaluation Corporation
SQL	Structured Query Language
SSD	Solid State Drive
STAC	Securities Technology Analysis Center
SZTS	ShenZhen Transportation System
TPC	Transaction Processing Performance Council TPC
TMS	Traffic Management Systems
UDF	User-Defined Function
YARN	Yet Another Resource Negotiator
YSB	Yahoo Streaming Benchmark
YCSB	Yahoo! Cloud Serving Benchmark

1 Introduction

1.1 Abstract

Big Data opens up new opportunities to define "Intelligent" mobility and transportation solutions. The transportation industry is a leader in creating the so-called *Internet of Everything*. Each day vast volumes of data are generated through sensors in passenger counting and vehicle locator systems and ticketing and fare collection systems, just to name a few.

The goal is to create value out of this amount of data, by providing a comprehensive picture of what's happening, using business analytics, leveraging big data tools and predictive analytics, to help transportation agencies improve operations, reduce costs and hopefully better serve travelers.

The technical challenge is that much of this Big Data is non-standard data (e.g., social, geospatial or sensor-generated data that does not an easy fit into traditional, structured, relational data warehouses or databases).

An additional challenge is that with such an amount of real-time structured and unstructured data captured from a variety of sources, it is difficult to determine which data is most valuable. Terabytes of data are collected and result in an added complexity to the underlying IT infrastructures.

These terabytes of data require immense amounts of storage in silo after silo of transportation operator data centers. In order to analyse Big Data, an appropriate Data Infrastructure needs to be in place to:

4. store and maintain data
5. analyse data
6. present results in a clear visual way

Several Big Data platforms have been proposed recently, open source and proprietary. In order to tackle the demands and challenges in the transportation domain, an optimal stack of Big Data technologies needs to be selected and designed based on the application requirements.

This is not an easy task.

This report, which is a follow up of Deliverable 1.1, offers an in-depth introduction to relevant technologies for Big Data Analytics and Big Data Management. It also looks at how these technologies are applied to build a Big Data Platform suitable for the transport sector. We present in detail how application-specific benchmarking can be used in order to evaluate which Big Data technologies are most suited for the domain. We conclude the report with an applied example of using data analytics for urban mobility.

1.2 Purpose of the document

The objective of this deliverable is to help to understand and choosing the right Big Data Technologies for transportation. The data flow in the transport sector comprises three components, namely: data generators, Big Data platform, and applications. Figure 1 depicts the abstract view of the data flow from data generators to applications. LeMO Deliverable 1.1

(Hee et al., 2018) defines data generators and applications, while the "Big Data platform" has remained a black box until now**.**

Emerging Big Data approaches such as Kafka and Spark are defined for building a system that supports real-time processing and analysis. The successful technology stack is only determined based on a strong understanding of the application requirements and the Big Data Vs (four challenges introduced in D1.1: volume, velocity, variety, and veracity).

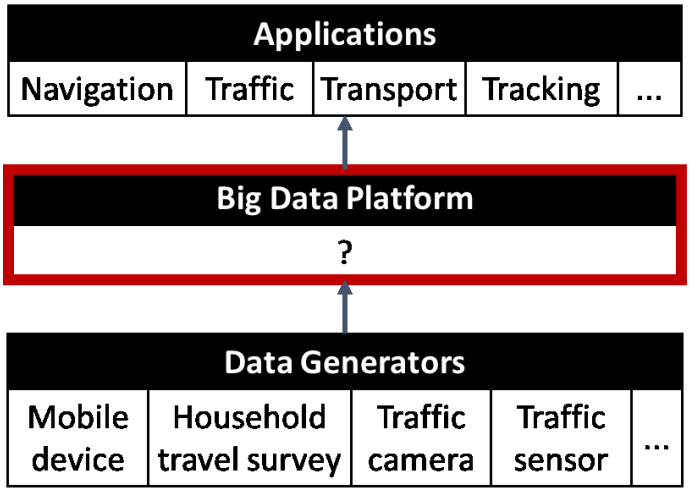

Figure 1: Data flow from the data generators to applications in the transport sector

In the following chapters, this report addresses each challenge step by step. Chapter 2 introduces Big Data analytics and Big Data methodology. Analytics in the transport sector is also investigated. Chapter 3 surveys state-of-art Big Data technologies and exposes the complexity of the ecosystem, i.e. how the hardware, management, platform and application must be integrated. For instance, each technology performs a particular task in contrast to conventional databases, which can execute several tasks such as storing, processing, and analyzing. Chapter 4 introduces the Big Data architecture utilizing Big Data technologies in the transport sector. Chapter 5 investigates Big Data benchmarks to learn, how we can select appropriate tools and architecture. Chapter 6 is our contribution that proposes a custom application benchmark. It is a proposal to convert an application into a benchmark tailored to a specific use case and specific data. Finally, Chapter 7 summarizes all findings, provides general conclusions and identifies needs for further research.

1.3 Target audience

The findings of this document will bring value to those who develop Big Data platforms in the transport industry. D1.1 revealed that many organizations and companies, in research and industry, who target big data applications in the transport domain are confronted with the technical challenges of utilizing big data technologies. This indicates the great interest that lies in the transport industry to tackle these challenges. This report aims to meet the needs by providing a platform-level-solution utilizing Big Data technologies. In addition, this document offers a practical guideline for anyone who wants to implement an application by harnessing state-of-art Big Data technologies.

2 Big Data Analytics

Big Data analytics is the process of extracting values from the complex and large amount of data. Type of data analytics is determined corresponding to the data mining objective. This means that the type of data analytics needs to be set at the initial phase of the data mining process. Some data mining objectives are relatively simple such as generating a report or gaining the awareness of the data. However, some objectives are complex as providing an actionable value with a specific decision.

There are four dominant types of analytics namely: *descriptive analytics*, *diagnostic analytics*, *predictive analytics* and *prescriptive analytics*. And each of these offers a different level of insights as shown in Figure 2.

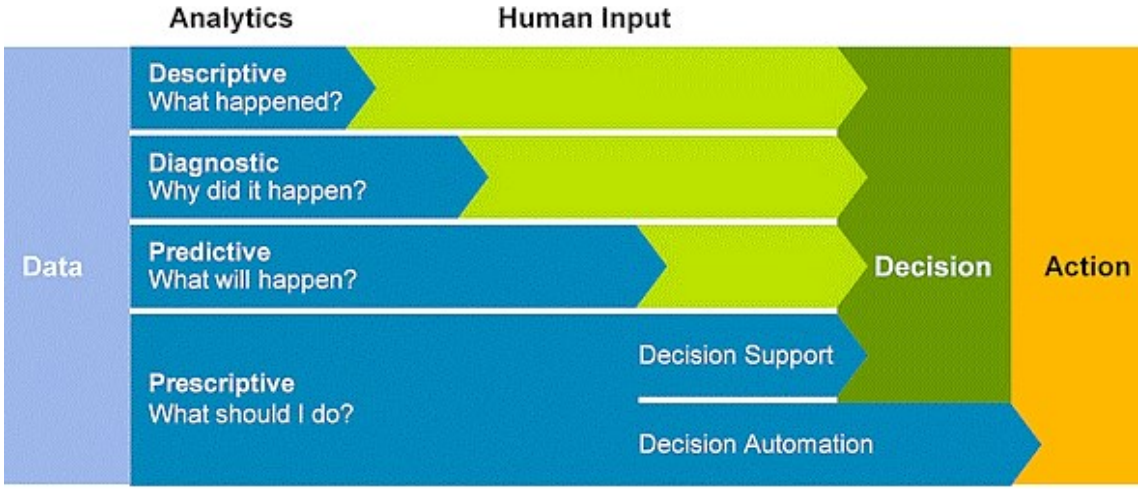

Figure 2: Visualization of data analytics type[1]

Descriptive analytics provides information on a given dataset. It aims to offer a better understanding of data with numerous perspective views. Data visualization of the statistical summary is the de facto standard method in this phase. Clustering algorithm in machine learning is a new approach to understand the intrinsic structure of data. For instance, popular cases in descriptive analytics are the automatic customers' segmentation or news article segmentation.

Diagnostic analytics aims to identify a set of driving predictors (features) to a data mining goal. It is also known as data forensics because this phase tries to capture the reasons leading to the mining questions. It is not necessarily the causation, but more association with the target variable. The outcome of this phase can be a subset of features or weighted features which are corresponding to the mining goal.

Predictive analytics tries to predict what will happen to the newly arriving data (unknown data) based on the given dataset. Classification algorithm and regression algorithm in machine learning are the popular algorithm families. They create either a classifier[2] or a regressor[3]. For instance, a binary classifier can predict if my flight would be delayed or not, whereas a regressor can predict how many free parking lots will be available at a certain time and date.

[1] https://www.gartner.com/newsroom/id/2881218
[2] https://en.wikipedia.org/wiki/Statistical_classification
[3] https://en.wikipedia.org/wiki/Regression_analysis

Prescriptive analytics aims to provide actionable values after making a prediction. It is the next step of predictive analytics that encourages a decision maker to make an action based on the prediction from the previous phase. For instance, a system can predict traffic jams at a certain time and recommend the best routing connection based on the predicted result.

2.1 Analytics in the transport sector

In the transport sector, *descriptive analytics* is the most studied in the last 15 years according to (Ghofrani et al., 2018). This study conducted an intensive literature review and investigates 106 studies on recent applications of big data analytics in railway transportation systems.

It shows that the majority of studies are at descriptive analytics in railway transportation systems (37%), predictive analytics (34%) is following and prescriptive analytics (29%) is the least studied.

Figure 3 classifies the analytics type of 106 studies and Figure 3 shows the distribution of level of result-oriented analytics in each year. Almost half the studies were focused on the use of analytics for the maintenance of the railway transportations system using amongst others sensor, traffic and inspection data.

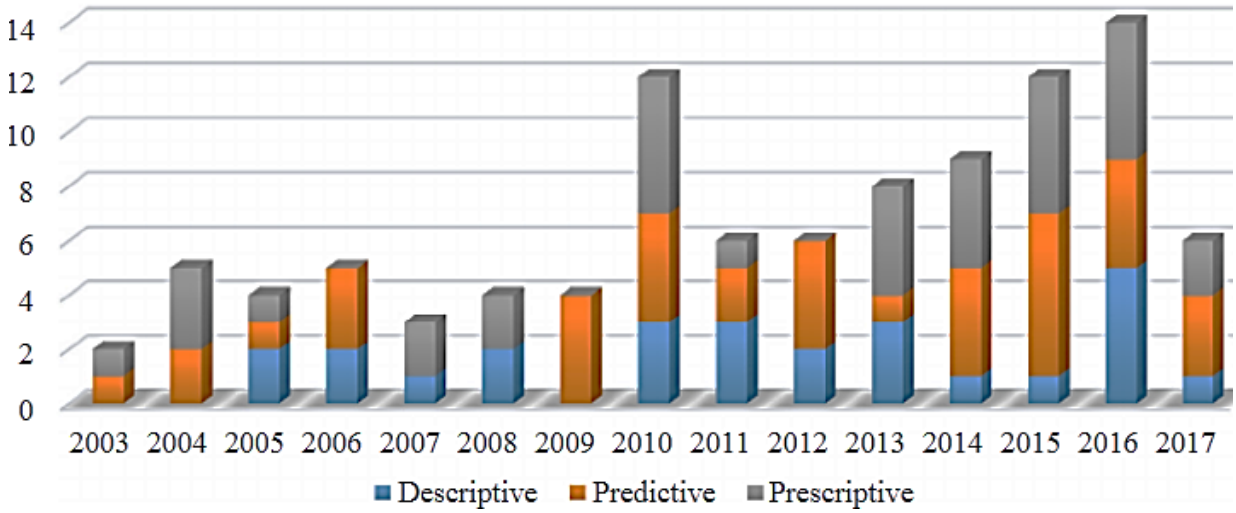

Figure 3: Type of analytics in the studied articles by year (Ghofrani, F., et al., 2018)

An online survey conducted 2015 by Schoenherr et al. showed that 45% of individuals, who operate in the field of transport and logistics, use data analytics or at least plan to use it in the future. The participants of the survey owned a position commonly associated with logistics and transportation. In total 240 individuals were asked. The selection of the participants suggests that very unspecific conclusions can be made, since there no indication to which degree participants hold decision making powers.

Since 2013 the number of research papers surveyed by Ghofrani et al. (2018) are suggesting an increasing interest in data analytics in transport. According to Schoenherr et al. (2015) almost half of the individuals related to the transport and logistics industry see at least a future use of data analytics in their field. There is no conclusive research available clarifying which and how many companies already implemented data analytics in the transport sector in Europe, or are planning to, which technologies are used and what the measurable benefits

are. The next work package of this project will focus on case studies of companies and institutions in the transport sector and their potential use of data analytics. The analysis of this research will provide some insight regarding this topic.

A Big Data analytics process consists mainly of two individual layers, a conceptual and an architecture or technology layer. These two layers are framed by the formulation of the domain-specific goals and tasks for which the analytics are utilized.

Figure 4 visualizes a schematic overview of the Big Data analytics pipeline adapted to the transport sector. This depiction is not aiming for completeness but offers an understanding of the interaction between challenges in the transport sector and the corresponding Big Data architecture. Here the conceptual layer is about defining and formulating the necessary steps of the analytics process. On top of the technology layer, sequential steps of data processing are defined. Each step in the conceptual layer corresponds to a specific Big Data architecture which suits the corresponding task and meets the necessary requirements.

Data analytics is a toolset which can help reach existing goals in transportation. These goals can be of political or social nature, like improving the quality of life in urban areas by finding efficient solutions for passenger mobility or freight transport. These solutions can, for example, reduce the occupation of space by individual car usage. Pollution in urban areas can be reduced by using analytics for optimizing energy consumptions. Facing the challenge of an aging society, data analytics can be employed to provide a sufficient mobility service in rural areas. Another important goal is to reach climate protection target emissions within the transport sector.

Companies associated in the transport sector employ Big Data analytics for example to reduce their costs for maintenance of vehicles and infrastructure, to increase the energy and cost efficiency in the transportation process or optimize their logistics to name a few. Another important objective is the improvement of customer's satisfaction (Khan et al., 2017).

In order to reach the predefined goals, tasks structuring data analytics procedures are formulated. In the transport sector, the amount of available data is huge, coming from various sources related to transportation, as freight, passengers, vehicles, infrastructure, etc. Data can be generated by a sensor, via human input or even be the output of a previous analytics process. This first stage is indicated in the left black box of Figure 4 labeled as *Available Data*. In Deliverable 1.1 (Hee et al., 2018) of this project various sources of data and corresponding available information are provided and discussed in detail.

The unprocessed data is often stored in so-called *Data Lakes*, which allow the storage of huge amounts of data with various formats and offer several interfaces for data retrieval, relational or non-relational. This software is built on a hardware architecture which fulfills the corresponding demands in capacity and speed. In sections 3.1 to 3.3 of this report, the different components of the Big Data software and hardware technology are discussed in detail.

Defining which information to use for a specific task, preparing and selecting data from relevant features and instances, is the first step of the analytics procedure. The data needs to be cleaned and the data quality needs to be assessed. It undergoes a process of discretization, compression, and spatial and temporal alignment. Suitable formats are chosen and the data

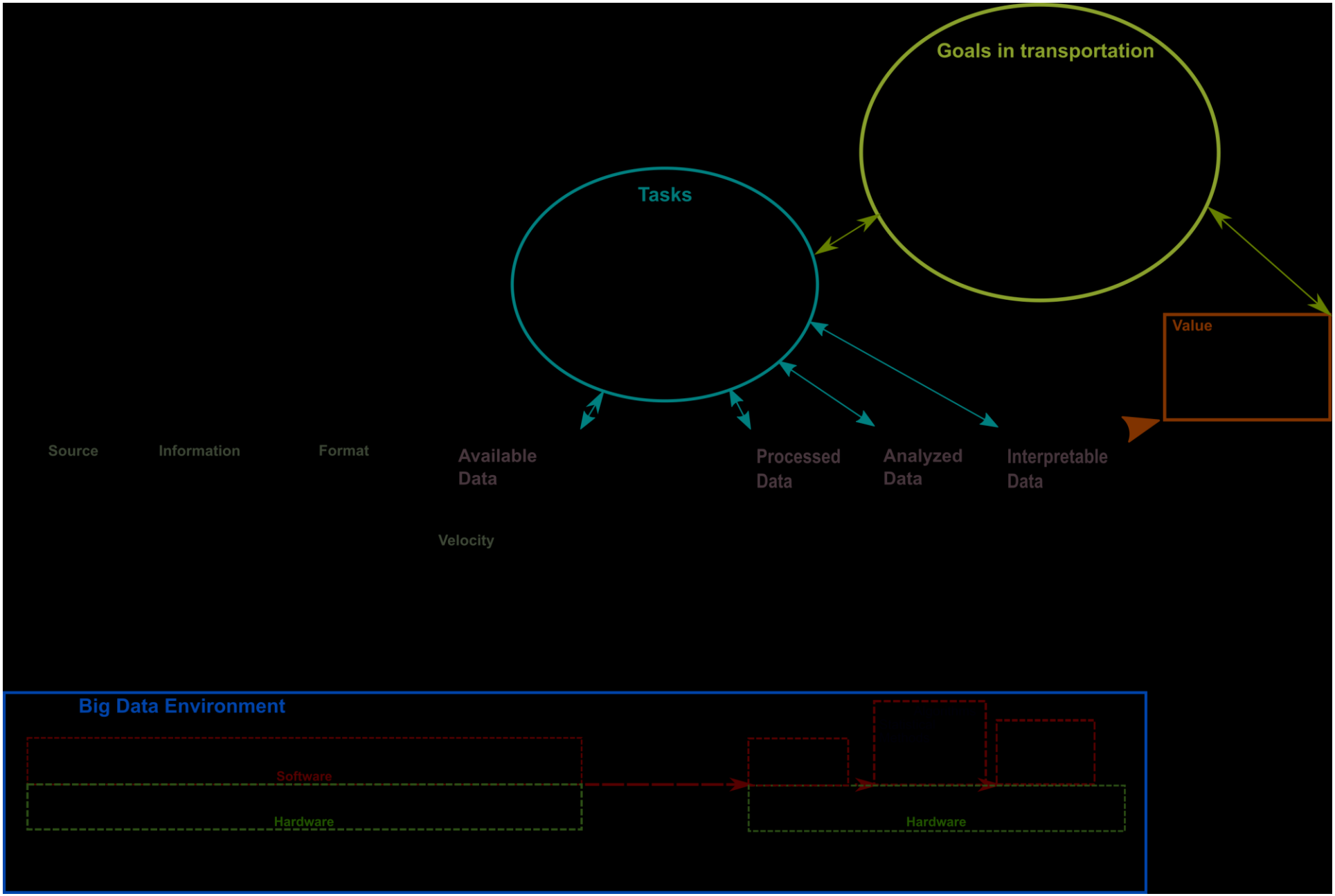

Figure 4: The Big Data analytics pipeline in the Big Data technology environment adapted for the example of transportation.

is managed in a way it can be dealt with in accordance to existing policies like privacy regulations. For machine learning tasks a training set is built.

This step requires certain processing and management technology which is explained in greater detail in section 3.4.

This first step is in many cases the most work-intensive one offering many challenges. For example, Traffic Management Systems (TMSs) are often build of components with limited integration capabilities and non-standardized interfaces and display rules. This makes the employment of data analytics hardly feasible. A standardized approach of data collection and dispatching can help to enable an integrated TMS. This would mean the creation of standardized interfaces and a standardized technology environment specifically for transport processes (Luckow et al., 2017).

In a next step the actual analytics, modeling, or statistical analysis is done using the preprocessed data. This intermediate step starts from the black box indicated in Figure 4 labeled with *Processed Data*. This step might, for example, consist of building interfaces with monitoring systems providing access to timestamp and position of vehicles and the real-time transmission of this information by writing a suitable processing script. Now algorithms are designed based on the formulated tasks or suitable statistical methods and modeling prescriptions are selected. Section 3.4 of this report offers an overview of the most relevant tools used for this step.

Before the analyzed data (indicated in Figure 4 as a black box labeled with *Analyzed Data*) can be used as a basis for decision making or predictions in order to reach the underlying goals, it must be made interpretable. It will be processed using visualization techniques to create infographics, texts or tables. Using suitable technology, one finally can create interpretable data. Only after having completed this step (indicated in Figure 4 as a black box labeled with *Interpretable Data*) value in terms of predictions, the basis for decision making, etc. can be created from the use data analytic tools.

The complete analytics procedure can be understood as an iterative process. Interpretable data supports reaching the desired goals but also triggers the formulation of new ones through newly gained insights. The increasing amount of available data will lead to a formulation of new tasks, which again influences the formulation of goals and so on.

The two main domains of transport, *passenger* and *freight* transport, offer distinct challenges and goals to reach. The modal split is quite different for both sectors.

In freight transport in Europe ca. 75% of freight is transported on roads, 18% on the rail and almost 7% via inland shipping. The modal split is almost constant for the last decade. More than 92% of passenger transport is conducted on roads, which is the sum of 82% share with passenger cars and 10% busses. Only ca. 8% of passenger transport in Europe is done via rail (Eurostat, 2017).

In order to reach the formulated goals in the transport sector, data analytics can help to improve accessibility and availability of certain modes supporting a change in modal split. An example would be to decrease the number of routes in cities conducted with cars or trucks via roads via for example ride sharing or alternative modes.

Freight: the context of the logistics industry and supply chain

In freight, five main sources for data generation exist: the infrastructure, the vehicles, the drivers, the company and the freight itself.

Data from other sources, like weather data or social media data might also be useful for certain domain specific tasks. A huge amount of data in logistics is produced by sensors implemented in the vehicles, infrastructure, freight or even driver's mobile devices. Possible sources of data are, just to mention a few, RFID chips attached to freight, sensors collecting data about the fuel, air, pressure and other data on the physical status of vehicle parts. Data is collected by the signaling or satellite navigation systems like GPS – either from data receivers in vehicles or freight, but also from mobile devices of passengers or drivers. This data can be combined with e.g. weather or geographical data, or customer demand data.

Some of the major topics to be addressed by data analytics in the logistics industry transport sector are (Jeske et al., 2013):

- *(Real-time) route optimization*: routes are dynamically adjusted with the help of data analytics using traffic, weather, supply and demand information,
- *Network and capacity planning*: short- to long term planning of available network and capacity along the supply chain,
- *Predictive and prescriptive maintenance* of vehicles and infrastructure,
- *Risk evaluation and resilience planning*: using data analytics for predictive maintenance of the involved vehicle stock and minimize failure due to climate or geographical incidents,
- *Environmental intelligence*: collected data on pollution, traffic, noise and occupied space can be used to improve conductions,
- *A special emphasis lies on the "last mile",* wherein the context of distribution of goods a multimodal approach can be useful in order to optimize for example deliveries.

Analytics of these data types is employed for logistic planning, e.g. for optimizing the distribution of freight. Distribution of goods can be formulated as a network problem starting from the production point (or points), through possible intermediate points, until reaching the destination, with different values for supply, capacity, and demand on each network point or path. In order to react to fluctuating customer demands and network capacity, predictive analytics are essential tools to deal with the complex network problems.

The optimization of routing is another important topic in logistics. The optimal route depends on the necessary number of stops (nodes) and the available data on the paths between these nodes (e.g. traffic, weather, infrastructure status, etc.) and analytics. Long term fleet monitoring can help minimize the cost of maintenance, material, capital investment and is, therefore, an important sector for data analytics. Strategy decisions and risk management of logistics processes can be better accessed using descriptive and predictive analytics.

Predictive or prescriptive maintenance using data analytics can be distinguished into three main parts: system component degradation modeling, infrastructure and vehicle cost modelling and the real-time monitoring and visualization of the vehicles and infrastructure condition. Coping

with climate change predictive analytics provide improved usability of energy efficient transport modes and offer predictive analytics of greenhouse emissions or nitrogen oxide outputs.

Passengers: improved and connected mobility service

Other than in the logistics industry, data analytics for creating improved passenger mobility is offered by one or several parties like public transport companies, car retailers or an independent platform, but used by a private person or group. Improved passenger mobility can refer to intelligent transport in urban areas. Based on data analytics and software it enables the use of multimodal mobility options to optimize time and cost and to minimize for example pollution or traffic.

In order to create bigger value, data analysis methods and platforms owned by the individual players, e.g. public transport companies, car sharing providers, taxis, bike or car rental companies, etc., need to be connected. This connection needs advanced analytics itself in order to create meaningful action or products.

Predictive analytics from data like satellite navigation, traffic, social media, etc. can be used to optimize the connection of different parts of an entire passenger route, combining different modes. The matching of travel routes of different individuals for the purpose of ride or vehicle sharing can be realized in an efficient way based on data analytics. In rural areas, the challenges for mobility concepts are others than in the urban areas. In rural region, data analytics might be most useful in order to provide autonomous, demand-based public transport in order to operate cost effectively. New business models offering flexible, connected and multimodal mobility are needed and can be designed using descriptive data analytics.

Important analytics methods for improved passenger transport using anonymized passenger data are modeling and prediction of movement patterns, dwell and travel mode analysis and the optimization of the first and last mile (Poonawala et al., 2016; Heggenber et al., 2018).

2.2 Data Driven Methodologies

Knowledge discovery in databases (KDD) is the data mining process of turning data into knowledge (Goebel & Gruenwald, 1999). The KDD process can be split into multiple stages and there are many standardized methodologies available for the KDD process. However, there is no specific data driven methodology for transportation. Three methodologies are commonly used in research and in practice namely: CRISP-DM, SEMMA, and KDD.

- The CRISP-DM methodology (Anand et al., 2007) is the acronym of CRoss Industry Standard Process for Data Mining developed by DaimlerChrysler, SPSS, NCR, and OHRA.
- The SEMMA methodology (Matignon, 2007) is the acronym for Sample, Explore, Modify, Model, and Assess developed by SAS Institute.
- KDD process is short for the Knowledge Discovery in Databases. It has emerged from a research community and first introduced by (Fayyad & Smyth, 1996) and (Piatetsky-Shapiro et al., 1996).

This report explains the data mining process with a focus on CRISP-DM and then SEMMA and KDD will then be compared to the stage by stage based on CRISP-DM as shown in Table 1.

Table 1: Summary of methodologies

CRISP-DM	SEMMA	KDD
Business understanding	-	Learning the application domain
Data understanding	Sample	Creating a target dataset
	Explore	Data cleaning and preprocessing
Data preparation	Modify	Data reduction and projection
Modeling	Model	Choosing the function of data mining
		Choosing the data mining algorithm
		Data mining
Evaluation	Assessment	Interpretation
Deployment	-	Using discovered knowledge

Table 2 is a compilation of studies using data driven methodology in the transportation sector. This is the extended table based on the work by (Mohd Selamat et al., 2018). This report remarks the mode of each study according to the LeMO taxonomy namely: Air, Rail, Road, Urban, Water, and Multimodal. All modes are covered except the multimodal transportation mode. Ten out of fourteen studies adopted the CRISP-DM model and the remaining studies used the SEMMA and KDD model equally. The popularity of CRISP-DM is proven by the polls conducted at KDNuggets, a leading website on Data Science. In 2002[1], 2004[2], 2007[3] and 2014[4], CRISP-DM is nominated as the most used methodology for developing data mining projects.

Table 2: Examples of Studies using Data Driven Methodology for Transportation

Studies	Methodology	Mode	Description
(Chen et al., 2004)	CRISP-DM	Road	Understand the status of the driver such as being out on duty, driving against traffic regulations.
(Viglioni et al., 2007)	CRISP-DM	Rail	Prediction of railroad demands to facilitate operation and manpower planning
(Wong & Chung, 2007)	KDD	Air	Mining passengers ' demographic, travel behavior and core service quality information for customer retention initiatives
(Haluzová, 2008)	CRISP-DM	Urban	Identification of the accident influences between car and tram on the electric tramway net
(Agenda et al., 2008)	CRISP-DM	Urban	Understand the accident and delay behavior of tram in Prague.
(Shin et al., 2009)	SEMMA	Urban	Analyzing passenger pick-up location patterns to proposed potential pick-up locations for empty taxis
(Mirabadi & Sharifian, 2010)	CRISP-DM	Rail	Analyzing historical accident data to discover the unsafe condition contributing factors
(Zhang et al., 2010)	KDD	Rail	Deriving intelligent-based decision making in accident treatments

[1] https://www.kdnuggets.com/polls/2002/methodology.htm
[2] https://www.kdnuggets.com/polls/2004/data_mining_methodology.htm
[3] https://www.kdnuggets.com/polls/2007/data_mining_methodology.htm
[4] https://www.kdnuggets.com/polls/2014/analytics-data-mining-data-science-methodology.html

(Nayak et al., 2011)	CRISP-DM	Road	Modeling the crash proneness of road segments for road safety.
(Greis & Nogueira, 2011)	CRISP-DM	Water	Identification of high-risk shipments reaching the US ports
(de Almeida & Ferreira, 2013)	SEMMA	Road	Identification of the most fuel-efficient resources in route operation and areas of resources for improvements
(Lukácová et al., 2014)	CRISP-DM	Air	Analyzing the aviation historical incident data to predict potential incidents and implications
(Moreno-Díaz et al., 2015)	CRISP-DM	Urban	Predicting passenger demand for efficient resource planning and deployment
(Cristóbal et al., 2018)	CRISP-DM	Urban	Analyzing travel time to understand the behavior patterns of travel times on the different transport routes

The CRISP-DM is not a linear process, but an iterative process with six phases and numerous bidirectional arrows as shown in Figure 5. The outer circle and the arrows in the diagram represent the nature of the data mining process. The lessons learned from one phase can trigger another phase, but the process may not end to the stage of Deployment due to the bidirectional dependencies.

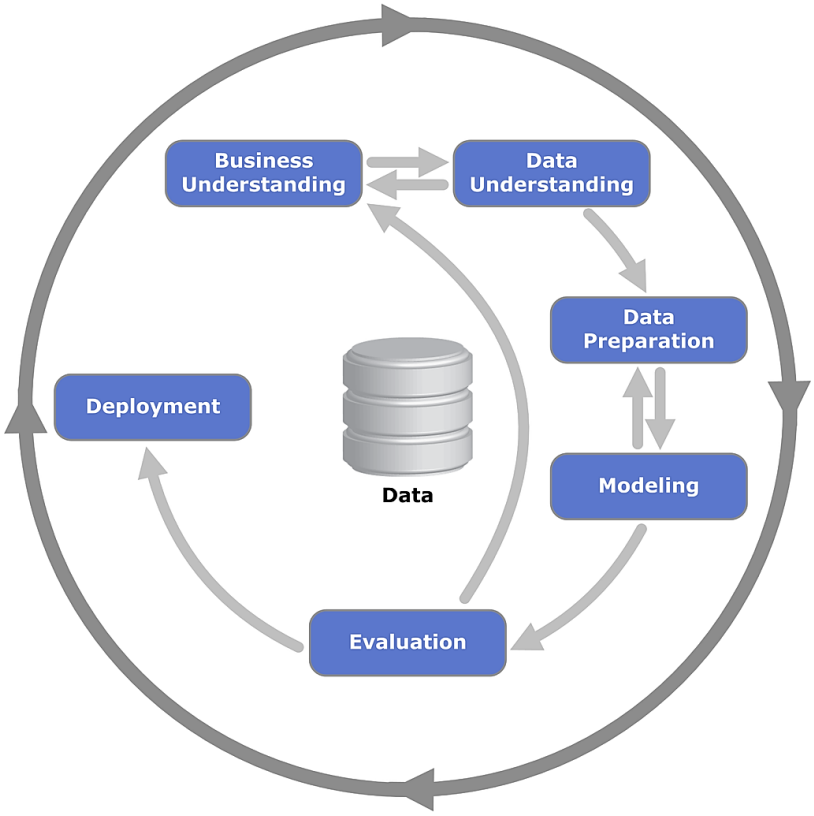

Figure 5: CRISP-DM showing the relationship between the different phases (IBM, 2011)

The six phases of CRISP-DM are namely: Business Understanding, Data Understanding, Data Preparation, Modeling, Evaluation, and Deployment. The initial phase is the ***Business***

understanding in CRISP-DM. This phase does not exist in SEMMA. The goal of this phase is setting a data mining objective in the context of the problem domain.

The next phase is the ***Data understanding*** consists of multiple activities such as data collection, data exploration, data sampling and more. The goal of this phase is to get familiar with the data and discover hidden insights into the data. The corresponding phases in SEMMA are ***Sample*** and ***Explore***, meanwhile two phases are found in KDD namely: ***Creating a target dataset*** and ***Data cleaning*** **and** ***preprocessing.***

The next phase is the ***Data preparation*** which is known as the most recursive and time consuming phase. The goal is to construct the final dataset which will be fed into the modeling phase. This phase includes cleaning, imputation, feature engineering (feature selection, extraction, and construction) and more. ***Modify*** is the corresponding phase in SEMMA, while in KDD it is ***Data reduction and projection***.

The following is the ***Modeling*** phase, which includes various activities. For instance, the problem type of data mining setting (classification, regression, clustering, abnormally detection or reinforcement learning), algorithm selection, and hyper-parameter setting.

Then, the ***Evaluation*** phase is the stage to check the quality of the model before the deployment. The goal of this phase is to validate if the model achieves the data mining objective(s) defined in the initial phase. It is equivalent to the Assessment in SEMMA, while it is equivalent to the Interpretation in KDD process.

The last phase is the ***Deployment*** which is either a data pipeline or a data-driven result. The end-to-end data pipeline is more appreciated because it is reusable and repeatable. It is noteworthy to mention of the cyclical nature of the method. The experiences of the previous process can trigger a new process, in other words, the new process will benefit from the lessons learned from the previous process. SEMMA does not cover this phase, but KDD has a similar step called ***Using discovered knowledge phase***.

3 Big Data Technologies

There are many technologies available, for advanced analytical workloads of large data. Commercial platforms consist of a predefined combination of hardware and software components. Examples of commercial platforms are Aster Data Systems[1], Oracle Big Data Appliance[2], IBM Netezza[3], SAP HANA[4] and HP Vertica[5]. Open-source systems or platforms, on the other hand, can run on any hardware. Unlike the commercial platforms, they are not like readymade appliances, thus they need to be configured by a developer. Examples of open-source platforms are Apache Hadoop[6], Apache Geode[7], VoltDB[8], Memcached[9], and Hazelcast[10].

This report focuses on the heterogeneity of Big Data Technologies in the Hadoop Ecosystem. Hadoop, the most commonly used framework, combines commodity hardware with open-source software (McAfee et al., 2012). Apache Hadoop is a framework that allows managing distributed processing of big data across clusters of computers using simple programming models. It was inspired by Google's MapReduce and Google File System (GFS) and in practice, it has been realized to be adopted in a wide range of cases. Apache Hadoop is designed to scale up from single servers to thousands of machines, each of them offering local computation and storage. Hadoop is composed of the Hadoop Distributed File System (HDFS) and of the programming paradigm MapReduce (Karloff et al., 2010). HDFS allows applications to be run across multiple servers, which have usually a set of inexpensive internal disk drives; the possibility of the use of common hardware is another advantage of Hadoop. The MapReduce is capable to detect and solve failures automatically in the cluster. The redundancy, in fact, provides fault tolerance and capability to self-healing of the Hadoop Cluster.

A similar and interesting solution is HadoopDB, proposed by a group of researchers at Yale. HadoopDB was conceived with the idea of creating a hybrid system that combines the main features of two technological solutions: parallel databases in performance and efficiency, and MapReduce based system for scalability, fault tolerance, and flexibility. The basic idea behind HadoopDB is to use MapReduce as the communication layer above multiple nodes running single-node DBMS (Database Management System) instances. Queries are expressed in Structured Query Language (SQL) and then translated into Map-Reduce. In particular, the solution implemented involves the use of PostgreSQL as database layer, Hadoop as communication layer, and Hive as the translation layer (Abouzeid et al., 2009).

[1] https://www.teradata.com/Products/Analytics-Platform
[2] https://www.oracle.com/engineered-systems/big-data-appliance/index.html
[3] https://www.ibm.com/analytics/netezza
[4] https://www.sap.com/products/hana.html
[5] https://www.vertica.com/
[6] http://hadoop.apache.org/
[7] http://geode.apache.org/
[8] https://github.com/VoltDB/voltdb
[9] http://memcached.org/
[10] https://hazelcast.org/

However, the majority of the studies that we introduce will be focussing on the heterogeneity of the storage components in Hadoop Ecosystem. A Big Data system consists of multiple components each with specific functionality, thus the heterogeneity is also scattered in other layers. For example, data, stream or graph processing technologies can be used in a Big Data system depending on the use case. Numerous classifications and taxonomies of the Big Data technologies have been developed by many researchers.

However, there is no consensus among studies for a universal taxonomy of the Big Data technologies. For instance, Hu et al. present the history and definitions of Big Data with a map of the Big Data technologies dividing them into four phases: Generation, Acquisition, Storage, and Analytics. They further suggest a Big Data layered architecture consisting of infrastructure, computing and application layers (Hu et al., 2014). Hashem et al. propose the classification of the Big Data technologies with the large-scale data in the cloud. They identify five aspects: i) data source, ii) content format, iii) data stores, iv) data staging and v) data processing (Hashem et al., 2015). Lipic et al. present an overview of big data tools. Additionally, they provide an abstract Big Data analysis stack consisting of four layers: i) Cloud resources, ii) Processing engines, iii) Applications and iv) Data analysis (Lipic et al., 2014). The authors (Wu et al., 2015) present a comprehensive taxonomy based on various aspects of the large-scale data management systems covering the data model, the system architecture and the consistency model. The authors (Qiu et al., 2014) give an extensive overview of the Apache Big Data Stack and propose an integration with High-Performance Big Data Stack.

This report follows the classification proposed by (Ivanov et al., 2016). This study made an intuitive representation of a Big Data platform with the concept the heterogeneity paradigm. The heterogeneity paradigm is a concept to understand core components of the Hadoop Ecosystem and investigates the interconnection between its components. It also maps each component into different layers to provide a comprehensive overview of the Hadoop Ecosystem. Based on the concept of heterogeneity, an abstract view of a Big Data technology groups was defined and presented in Table 3.

The architecture consists of four layers, which are also called levels: hardware, management, platform, and application. This division in levels is not strict but represents the major features and functionality of the components in a Big Data platform. The hardware layer represents the server components of the system and the fact that they can vary in storage, memory and processor type and size. The management layer is dealing with the system resource management and offers services to the applications running on the upper layers. The platform layer represents the main storage and processing services that a Big Data platform provides. Finally, the application layer is hosting the variety of Big Data applications running on top of the services provided by the lower layers.

The rest of this section (3.1, 3.2, 3.3, and 3.4) looks deeper into each level by enlisting the respective technology components and their categories. Note that major part of the text has been published in (Ivanov et al., 2017) at Chapter 15 “The Heterogeneity Paradigm in Big Data Architectures” and reprinted here with permission of the authors.

Table 3: Abstract Big Data Technology Group

Heterogeneity Level	Abstract Big Data Technology Group	
Application	Data, Stream & Graph Analytics	Content Analysis
	Machine Learning	Procedural Language
	Application Framework	Search Engine
	SQL-on-Hadoop	Data Modeling
	Data Acquisition	Library Collection
Platform	Data Collection	Data Governance
	Data Serialization	Machine Learning Framework
	Data Layout	Workflow Scheduling
	In-Memory Storage	Execution Framework
	Data & Graph Storage	Data, Stream & Graph Processing
Management	System Interfaces	
	Cloud Application Deployment	Application Management
	Distributed Coordination	Messaging Management
	Cluster Monitoring & Management	
	Virtualization-based & Container-based Resource Management	
Hardware	Memory Type & Size	CPU Type & Number of Cores
	Storage Type & Size	Accelerator Modules

3.1 Hardware Level

Undoubtedly recent advances in the processing and storing capabilities of the current commodity (off-the-shelf) servers have drastically improved while at the same time becoming cheaper (Intel, 2006). This reduces the overall cost of large-scale clusters consisting of thousands of machines and enables the vendors to cope with the exponentially growing data volumes, as well as the velocity with which the data should be processed. However, there have been other components like Field Programmable Gate Arrays (FPGAs), GPUs (Graphics Processing Unit), accelerator modules and co-processors which have become part of the enterprise-ready servers. They offer numerous new capabilities which can further boost the overall system performance such as:

1. optimal processing of calculation intensive application;
2. offloading part or entire CPU (Central Processing Unit) computations to them;
3. faster and energy efficient parallel processing capabilities; and
4. improved price to processing ratio compared to standard CPUs.

Recently, there have been multiple studies investigating how these emerging components can be successfully integrated into the Big Data platforms. In (Shan et al., 2010), the authors present a MapReduce framework (FPMR) implemented on FPGA that achieves 31.8x speedup compared to a CPU-based software system. (Kambatla & Chen, 2014) investigate the performance improvements of using Solid State Drives (SSDs) as an alternative to hard-disk drives and conclude that SSDs can achieve up to 70% higher performance for the 2.5x higher cost-per-performance. Similarly, (Kang et al., 2013) show that sorting in Hadoop with SSDs can be more than 3 times faster and reduce drastically the power consumption compared to hard disks.

Diversifying the core platform components motivates the investigation of the concept of heterogeneity on a hardware level and the new challenges that it introduces. Using the right hardware modules for a particular application can be crucial for obtaining the best price-performance ratio.

3.2 Management Level

As seen in Table 3 the management layer is positioned directly above the hardware level. It is responsible for the management and optimal allocation and usage of the underlying hardware components. There are multiple ways to achieve this:

- directly installing the operating system,
- using a container technology (container-based virtualization),
- using a virtualization technology (hypervisor-based virtualization) and
- utilizing a hybrid solution between the operating system (OS) and virtualization.

In the recent years, virtualization has become the standard technology for infrastructure management both for bigger cloud and data center providers as well as for smaller private companies (Staten et al., 2008). However, along with the multiple benefits that virtualization brings, there are also new challenges. The co-location of virtual machines hosting different application workloads on the same server makes the effective and fair resource allocation problematic. Also, the logical division of virtual machines with similar characteristics is not always possible. In the case of Big Data platforms with changing workloads, it is difficult to meet the network and storage I/O (input/output) guarantees. Therefore, the container-based virtualization, which comes at much smaller overhead as it is directly supported by the operating systems, has become a very popular alternative. Virtualization technologies provide better resource sharing and isolation in exchange for a higher overhead, whereas container-based systems achieve near-native performance but offer poor security and isolation (Xavier et al., 2014).

The Serengeti project (VMware, 2013, 2014) is one of the first initiatives to automate the management, starting, stopping and pre-configuring of Hadoop clusters on the fly. It is an open source project started by VMware and now integrated into vSphere as Big Data Extension, which has the goal to ease the management of virtualized Hadoop clusters. By the implementation of hooks to all major Hadoop modules, it is possible to know the exact cluster topology and make it aware of the hypervisor layer. This open source module is called Hadoop Virtual Extension (HVE) (VMware, 2013). Very interesting is the new ability to define the nodes (virtual machines) as either only compute or data nodes. The above implies that some nodes are storing the data in HDFS, while others are responsible for the computation of MapReduce jobs. Another very similar project, called Sahara (OpenStack, 2014), was developed as part of the OpenStack platform.

At the same time, there are a variety of other technologies, which help and improve the management of a Big Data environment, such as monitoring, deployment, coordination, messaging and resource scheduling tools. An extensive list of such tools together with a short description is provided in Table 4.

Table 4: Management Level Components

Type	Tools	Description
Virtualization-based Resource Management	Serengeti/Big Data Extensions	It is an open-source project, initiated by VMware, to enable the rapid deployment of Hadoop (HDFS, MapReduce, Pig, Hive, and HBase) on a virtual platform (vSphere). (VMware, 2013, 2014)
	Sahara/Savanna	It aims to provide users with simple means to provision a Hadoop cluster at OpenStack by specifying several parameters like Hadoop version, cluster topology, nodes hardware details and a few more. (OpenStack, 2014)
Cluster Resource Management	Mesos	A cluster manager that provides efficient resource isolation and sharing across distributed applications, or frameworks like Hadoop, MPI, Hypertable, Spark, and other applications. (Hindman et al., 2011)
	YARN	YARN (Yet Another Resource Negotiator/MapReduce 2.0) is a framework for job scheduling and cluster resource management. (Vavilapalli et al., 2013)
Container-based Resource Management	Docker	It is an open platform for developers and sysadmins to build, ship, and run distributed applications. Consisting of Docker Engine, a portable, lightweight runtime and packaging tool, and Docker Hub, a cloud service for sharing applications and automating workflows. (Docker, 2014)
	LXC/Linux Containers	LXC provides operating system-level virtualization through a virtual environment that has its own process and network space, instead of creating a full-fledged virtual machine. LXC relies on the Linux kernel groups functionality that was released in version 2.6.24.
	CoreOS	CoreOS is an open source lightweight operating system based on the Linux kernel and designed for providing infrastructure to clustered deployments while focusing on automation, ease of applications deployment, security, reliability, and scalability.
Cluster Monitoring & Management	Ambari	It provides an intuitive, easy-to-use Hadoop management web UI backed by its RESTful APIs for provisioning, managing, and monitoring Apache Hadoop clusters
	Helix	Apache Helix is a generic cluster management framework used for the automatic management of partitioned, replicated and distributed resources hosted on a cluster of nodes. Helix automates reassignment of resources in the face of node failure and recovery, cluster expansion, and reconfiguration.
Application Management	Cloudera Manager	Cloudera Manager is application management tool for the Cloudera Hadoop Distribution. It automates the administration, installation, configuration and deployment of cluster applications as well as offers monitoring and diagnostic capabilities.
Cloud Application Deployment	Whirr	Whirr is a set of libraries for running cloud services. It provides a cloud-neutral way to run services, a common service API and can be used as a command line tool for deploying clusters.
	JCloud	Jcloud is an open source multi-cloud toolkit for the Java platform. It provides functionality to create and control portable applications across clouds using their cloud-specific features.
Distributed Coordination	ZooKeeper	A centralized service that enables highly reliable distributed coordination by maintaining configuration information, naming, providing distributed synchronization, and group services. (Hunt, Konar, Junqueira, & Reed, 2010; Junqueira & Reed, 2009)
Messaging Management	Kafka	It is a distributed messaging system for collecting and delivering high volumes of log data with low latency. (Kreps, Narkhede, & Rao, 2011)

System Interfaces	Hue	Hue is a Web interface for analyzing data with Apache Hadoop. It supports a file and job browser, Hive, Pig, Impala, Spark, Oozie editors, Solr Search dashboards, Hbase, Sqoop2, and more.

3.3 Platform Level

The platform layer represents *the actual Big Data platform* which is responsible for the provision of general data and processing capabilities. In the last years Apache Hadoop has become the de facto platform for Big Data. It has two core components: HDFS and YARN (MapReduce 2.0). HDFS is responsible for the data storage, whereas YARN is for the processing and resource allocation between the jobs. More recently, Yahoo released the Storm-YARN (Yahoo, 2013) application which combines the advantages of both applications: real-time (low-latency) and batch processing. It enables Storm applications to utilize the Hadoop resources managed by YARN, which will offer new abilities for faster and more optimal data processing. The Spark platform developed by Zaharia *et al.* (Zaharia et al., 2012; Zaharia et al., 2010) is built on top of HDFS and introduces the concept of Resilient Distributed Datasets (RDDs). RDDs are fault-tolerant, parallel data structures that let users explicitly persist intermediate results in memory, control their partitioning to optimize data placement, and manipulate them using a rich set of MapReduce-like parallel operations (iterative machine learning algorithms and interactive data analytics).

The above are just a few examples of the existing platforms for data storage and processing. The question "*How to choose the right framework for a specific use case?"* is very important, but one needs sufficient background knowledge in order to answer it, as pointed out by Grover (Grover, 2015). In his post, he discusses and categorizes the different frameworks which can be run on top of HDFS. This complies with the chapter's goal, on providing an overview of the variety of frameworks in the platform layer. Table 5 provides a list of components, grouped by their functionality types. In the upper part are the storage components (Data, Graph and In-memory storage), followed by multiple processing frameworks (Data, Stream and Graph processing) and data tools. In addition, there are execution and machine learning frameworks as well as tools for workflow management.

The list of new frameworks and tools is constantly growing as are the new application requirements of the upper layer. Therefore, the importance of understanding the heterogeneity on this platform level is very essential for the successful management and processing of large datasets.

Table 5: Platform Level Components

Type	Tools	Description
Data Storage	HDFS	Apache HDFS (Hadoop Distributed File System) is a distributed file system that provides high-throughput access to application data. (Borthakur, 2008)
	Hbase	Apache Hbase is the Hadoop database, a distributed, scalable, big data store. It is used for random, realtime read/write access to your Big Data and is modeled after Google's Bigtable (Chang et al., 2008) (George, 2011)
	Accumulo	Apache Accumulo sorted, distributed key/value store is a robust, scalable, high performance data storage and retrieval system. It is based on Google's Bigtable design and is built on top of Apache Hadoop, Zookeeper, and Thrift.

	Hypertable	Hypertable is open source,scalable, distributed key/value store based on Google's Bigtable design, running on top of Hadoop.
	Cassandra	Apache Cassandra's data model offers the convenience of column indexes with the performance of log-structured updates, strong support for denormalization and materialized views, and powerful built-in caching.
	Phoenix	Apache Phoenix is high performance relational database layer over Hbase for low latency applications.
Graph Storage	Titan	Titan is a scalable graph database optimized for storing and querying graphs containing hundreds of billions of vertices and edges distributed across a multi-machine cluster. Titan is a transactional database that can support thousands of concurrent users executing complex graph traversals in real time.
In-Memory Storage	Tachyon	Tachyon is an open source, memory-centric distributed file system enabling reliable file sharing at memory-speed across cluster frameworks, such as Spark and MapReduce. (Li et al., 2013; Li, Ghodsi, Zaharia, Shenker, & Stoica, 2014)
Data Governance	Cloudera Navigator	Cloudera Navigator offers comprehensive auditing across Hadoop cluster by defining and automatically collecting data lifecycle activities such as retention and encryption policies.
	Falcon	Apache Falcon is a data processing and management solution for Hadoop designed for data motion, coordination of data pipelines, lifecycle management, and data discovery. Falcon enables end consumers to quickly onboard their data and its associated processing and management tasks on Hadoop clusters.
Data Collection	Chukwa	Apache Chukwa is a data collection system for managing large distributed systems. It also includes a flexible and powerful toolkit for displaying, monitoring and analyzing results to make the best use of the collected data. (Boulon et al., 2008; Rabkin & Katz, 2010)
Data Serialization	Avro	Apache Avro is a data serialization system. It provides: 1) rich data structures; 2) a compact, fast, binary data format; 3) a container file, to store persistent data; 4) remote procedure call (RPC) and 5) simple integration with dynamic languages.
Data Layout	Parquet	Apache Parquet is a columnar storage format available to any project in the Hadoop ecosystem, regardless of the choice of data processing framework, data model or programming language.
Data Processing	MapReduce	A YARN-based system for parallel processing of large datasets. (Dean & Ghemawat, 2008)
	Spark	Apache Spark is an open source cluster computing system that aims to run programs faster by providing primitives for in-memory cluster computing. Jobs can load data into memory and query it repeatedly much more quickly than with disk-based systems like Hadoop MapReduce. (Zaharia et al., 2012; Zaharia et al., 2010)
Stream Processing	Storm	An open source distributed real-time computation system. Storm makes it easy to reliably process unbounded streams of data, doing for real-time processing what Hadoop did for batch processing. (Leibiusky, Eisbruch, & Simonassi, 2012)
	Storm-YARN	It enables Storm applications to utilize the computational resources in a Hadoop-YARN cluster along with accessing Hadoop storage resources such as Hbase and HDFS. (Yahoo, 2013)
	Samza	Apache Samza is a distributed stream processing framework. It uses Kafka for messaging, and Hadoop YARN to provide fault tolerance, processor isolation, security, and resource management.
	S4	Apache S4 is a general-purpose, distributed, scalable, fault-tolerant, pluggable platform that allows programmers to easily develop applications for processing continuous unbounded streams of data.

	Spark Streaming	Spark Streaming makes it easy to build scalable fault-tolerant streaming applications by using the Spark's language-integrated API, which supports Java, Scala, and Python. (Zaharia et al., 2013)
Workflow Scheduling	Oozie	Apache Oozie is a workflow scheduler system to manage Apache Hadoop jobs. (Islam et al., 2012)
Execution Framework	REEF	REEF (Retainable Evaluator Execution Framework) framework builds on top of YARN to provide crucial features (Retainability, Composability, Cost Modeling, Fault handling and Elasticity) to a range of different applications. (Chun et al., 2013)
Graph Processing	Giraph	Apache Giraph is an iterative graph processing system built for high scalability. It originated as the open-source counterpart to Pregel (Malewicz et al., 2010), the graph processing architecture developed at Google.
	GraphX	GraphX is Apache Spark's API for graphs and graph-parallel computation. It unifies ETL, exploratory analysis, and iterative graph computation within a single system. (Gonzalez et al., 2014)
	Dato/ GraphLab	GraphLab is an open source, graph-based, high performance, distributed computation framework written in C++. (Low et al., 2012)
	Pegasus	PEGASUS is a Peta-scale graph mining system, fully written in Java. It runs in parallel, distributed manner on top of Hadoop. (U. Kang, Tsourakakis, & Faloutsos, 2009)
Machine Learning Framework	Oryx	The Oryx open source project provides simple, real-time large-scale machine learning / predictive analytics infrastructure. It implements a few classes of algorithm commonly used in business applications: collaborative filtering/recommendation, classification/regression, and clustering. (Cloudera, 2015)
	MLbase	MLbase is a platform for Implementing and consuming Machine Learning techniques at scale, and consists of three components: MLlib, MLI, ML Optimizer. MLlib is Spark's scalable machine learning library consisting of common learning algorithms and utilities. (Kraska et al., 2013; Talwalkar et al., 2012)
	H2O	H2O is an open source platform, offering machine learning algorithms for classification and regression over BigData. It is extensible and users can build blocks using simple math legos in the core. H2O keeps familiar interfaces like R, Excel & JSON. (0xdata, 2015)

3.4 Application Level

Satisfying all the Big Data application characteristics requires the platform to support all types of components starting from the data retrieval, aggregation, and processing including data mining and analytics. Moreover, applications with very different characteristics should be able to run effectively co-located on the same platform, which should further guarantee optimal resource and functionality management, fair scheduling and workload isolation. These requirements outline the importance of understanding the heterogeneity of the application level. To achieve these, the variety of existing technologies and their features should be thoroughly investigated and understood. Table 6 summarizes major part of the tools in the Hadoop Ecosystem, grouping them according to their functionality type.

In the first category defined as data acquisition are tools used to move and store data into Hadoop. Sqoop (Ting & Cecho, 2013) and Flume are the most widely used tools for data acquisition.

The second category, called SQL-on-Hadoop, represents the variety of Data Warehousing, Business Intelligence (BI), ETL (Extract-Transform-Load) (Baer, 2013) and reporting capabilities offered by the applications on top of Hadoop. Hive (A. Thusoo et al., 2010; Ashish Thusoo et al., 2009) is the most popular application in this category. It is a data warehouse infrastructure on top of Hadoop that provides data summarization and ad-hoc querying in SQL-like language, called HiveQL. Another important category is the application frameworks, which offer ready to use packages, libraries, and tools for building custom Big Data applications. The search engine category enlists components for enabling full-text search capabilities on top of Hadoop.

The last categories include different analytics types (Data, Graph and Stream analytics), machine learning and content analysis components, implementing specific use case functionalities.

Table 6: Application Level Components

Type	Tool	Description
Data Acquisition	Sqoop	A tool designed for efficiently transferring bulk data between Apache Hadoop and structured data stores such as relational databases. (Ting & Cecho, 2013)
	Flume	A distributed, reliable, and available service for efficiently collecting, aggregating, and moving large amounts of log data.
SQL-on-Hadoop	Hive	A data warehouse infrastructure that provides data summarization and ad hoc querying. (A. Thusoo et al., 2010; Ashish Thusoo et al., 2009)
	HCatalog	A set of interfaces that open up access to Hive's metastore for tools inside and outside of the Hadoop grid. It is now part of Hive. (Capriolo, Wampler, & Rutherglen, 2012)
	Impala	It is an open source Massively Parallel Processing (MPP) query engine that runs natively on Hadoop, enabling users to issue low-latency SQL queries to data stored in HDFS and HBase without requiring data movement or transformation. (Kornacker et al., 2015)
	Big SQL (IBM)	Big SQL is a massively parallel processing (MPP) SQL engine that deploys directly on the physical Hadoop Distributed File System (HDFS) cluster. This SQL engine pushes processing down to the same nodes that hold the data.
	SparkSQL (Shark)	A fully Hive-compatible data warehousing on top of Spark system that can run 100x faster than Hive. (Engle et al., 2012; Xin et al., 2013)
	Drill	Apache Drill is an open-source software framework (inspired by Google's Dremel) that supports data-intensive distributed applications for interactive analysis of large-scale datasets. (Hausenblas & Nadeau, 2013)
	Tajo	A relational and distributed data warehouse system for Hadoop, that is designed for low-latency and scalable ad-hoc queries, online aggregation and ETL on large-data sets by leveraging advanced database techniques.
	Presto (Facebook)	Presto is an open source distributed SQL query engine for running interactive analytic queries against data sources of all sizes ranging from gigabytes to petabytes. (Choi et al., 2013)
	HAWK (Pivotal, 2015b)	HAWQ is a parallel SQL query engine that combines the Pivotal Analytic Database with the scalability and convenience of Hadoop. HAWQ reads data from and writes data to HDFS natively. It delivers performance, linear scalability and provides tools interaction with petabyte range data sets. HAWQ provides users with a complete, standards compliant SQL interface.
	MRQL	Apache MRQL (pronounced miracle) is a query processing and optimization system for large-scale, distributed data analysis, built on top of Apache Hadoop, Hama, and Spark.

	BlinkDB	BlinkDB is a massively parallel, approximate query engine for running interactive SQL queries on large volumes of data. It allows users to trade-off query accuracy for response time, enabling interactive queries over massive data by running queries on data samples and presenting results annotated with meaningful error bars. (Agarwal et al., 2012, 2013)
Library Collection	DataFu	Apache DataFu is a collection of libraries for working with large-scale data in Hadoop. The project was inspired by the need for stable, well-tested libraries for data mining and statistics. (Hayes & Shah, 2013)
Data Modeling	Gora	Apache Gora is an open source framework that provides an in-memory data model and persistence for big data. It supports persisting to column stores, key/value stores, document stores and RDBMSs, and analyzing the data with extensive MapReduce support.
	Kite	Kite is a high-level data layer for Hadoop. It is an API and a set of tools that speed up development by enabling you to configure how Kite stores your data in Hadoop. (Kite, 2015)
Application Framework	Tez	Apache Tez is a general-purpose resource management framework which allows for a complex processing of directed-acyclic-graph of tasks and is built atop Hadoop YARN. (Apache, 2015)
	Cascading	Cascading is an open source application development platform for building data applications on Hadoop. It is used to create and execute complex data processing workflows on a Hadoop cluster using any JVM-based language (Java, JRuby, Clojure, etc.), hiding the underlying complexity of MapReduce jobs.
	Flink (Stratosphere)	Apache Flink features powerful programming abstractions in Java and Scala, a high-performance runtime, and automatic program optimization. It has native support for iterations, incremental iterations, and programs consisting of large DAGs of operations. (Alexandrov et al., 2014)
	Crunch	Apache Crunch Java library provides a framework for writing, testing, and running MapReduce pipelines. Its goal is to make pipelines that are composed of many user-defined functions simple to write, easy to test, and efficient to run.
Search Engine	Lucene	Apache Lucene is an open source, high-performance, full-featured text search engine library written entirely in Java. It is a technology suitable for nearly any application that requires full-text search, especially cross-platform. (McCandless, Hatcher, & Gospodnetic, 2010)
	Solr	Apache Solr is highly reliable, scalable and fault tolerant, providing distributed indexing, replication and load-balanced querying, automated failover and recovery, centralized configuration and more. It is built on Apache Lucene.
	Nutch	Apache Nutch is an open source web search engine based on Lucene and Java for the search and index component. It has a highly modular architecture, allowing developers to create plug-ins for media-type parsing, data retrieval, querying and clustering. (Khare, Cutting, Sitaker, & Rifkin, 2005)
	Elasticsearch	Elasticsearch is an open source, search server based on Lucene. It provides a distributed, multitenant-capable full-text search engine with a RESTful web interface and schema-free JSON documents.
Machine Learning	Mahout	A scalable machine learning and data mining library. (Owen, Anil, Dunning, & Friedman, 2011)
Data Analytics	Hama	Apache Hama is an open source project, allowing you to do advanced analytics beyond MapReduce.
Stream Analytics	SAMOA	Apache SAMOA is a distributed streaming machine learning (ML) framework that contains a programming abstraction for distributed streaming ML algorithms. (De Francisci Morales, 2013)

Procedural Language	Pig	A high-level data-flow language and execution framework for parallel computation. (Gates et al., 2009; Olston, Reed, Srivastava, Kumar, & Tomkins, 2008)
Content Analysis	Tika	Apache Tika is a toolkit that detects and extracts metadata and structured text content from various documents using existing parser libraries. (Mattmann & Zitting, 2011)
Graph Analytics	Faunus	Faunus is a Hadoop-based graph analytics engine for analyzing graphs represented across a multi-machine compute cluster.

4 Big Data Architectures

A big data architecture is an end-to-end system that covers data source ingestion, processing, and analysis to deal with a large amount of data. In the previous section, numerous Big Data technologies are outlined (Chapter 3). However, designing an optimal system architecture is much complex than technologies. One of the reasons is that Big Data architecture incorporates multiple Big Data technologies and tools, moreover, the challenges also need to cope with Big Data challenges at section 2.2.3 in LeMO deliverable 1.1 (Hee et al., 2018). In general, Big Data architectures are leveraging the parallel processing and the distributed storages over a scalable system. For instance, Spark is a popular Big Data technology to process a large amount of data with low latency in the Big Data architecture, whereas HDFS is one of NoSQL databases and widely used as a distributed storage in the Big Data architecture. Kafka is additionally utilized to ingest and process the data on the fly. (Mashrur et al., 2017) depicts a high-level Big Data architecture for connected transportation systems as shown in Figure 6.

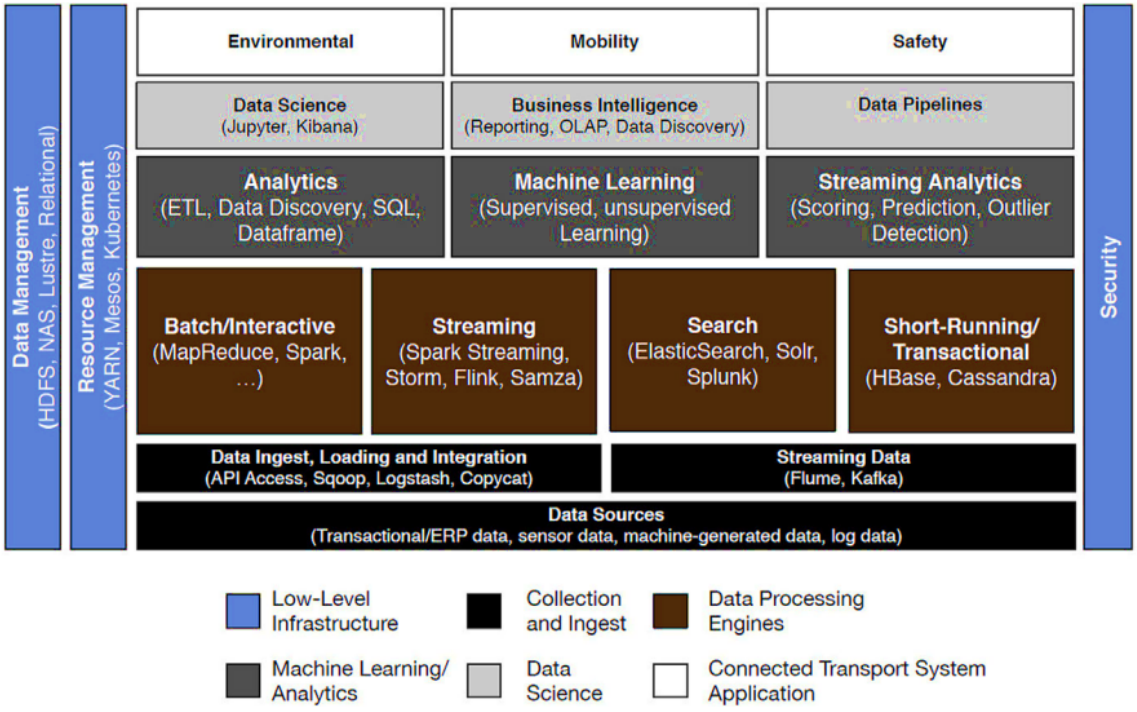

Figure 6: Big Data architecture for connected transportation systems (Mashrur et al., 2017)

The proposed architecture comprises multiple layers: The lowest layer is Collection and Ingest layer. It deals with various data sources. For example, vehicle and infrastructure sensor data, satellite data, signal data, social media data, weather and geographical data, and more. Various data sources and types are generated and ready to be stored or/and processed. Then, Data Processing Engines layer manages a basic processing such as batch processing or streaming processing. Analytics is done on the Machine Learning Analytics layer which is on the top of the processing layer. Data Science layer provides more advanced analytics above. Finally, numerous

transportation applications are running on the top. In addition, a few components on Hadoop Ecosystem provide some capabilities required by all layers.

In academia, numerous studies propose a Big Data architecture customized to a certain use case. In fact, most studies are investigate network level architectures. The studies leveraging Big Data technologies are mainly for the batch processing as shown in Table 7. Only a few architectures are designed for the streaming processing. This chapter will investigate further on two specific case studies: one is batch processing architecture, another is stream processing architecture.

Table 7: Examples of Studies on Big Data Architecture in the Transport Sector

Architecture type	Study	Big Data technologies	Reference
Architecture for the batch processing	A data platform for the highway traffic data	HBase, HDFS, Hive	(Mian et al., 2014)
	Sipresk: A big data analytics platform for smart transportation	HBase, HDFS	(Khazaei et al., 2015)
	An architecture for big data processing on intelligent transportation systems: An application scenario on highway traffic flows	Spark, mongoDB	(Guerreiro et al., 2016)
	Application of big data in intelligent traffic system	HDFS, Hive, Sqoop, Flume	(Zeng et al., 2015)
Architecture for the stream processing	Big data analytics architecture for real-time traffic control	Kafka, HDFS, NoSQL DB	(Amini et al., 2017)
	A cloud-based car parking middleware for IoT-based smart cities: Design and implementation	Kafka, Storm, HBase, HDFS, Flume	(Ganchev et al., 2014)

4.1 Sipresk: A Big Data Analytic Platform for Smart Transportation

Khazaei et al. proposed a Big Data architecture for an urban transportation application to gain insights into traffic patterns (Khazaei et al., 2015). The platform is called Sipresk architecture that is designed to be reliable, scalable, and adaptable to the changing operating conditions. This architecture consists of data layer, analytics layer, and management layer. It supports both static analysis (retrospective analysis) and streaming analysis (online analysis). This study validated several use cases such as finding average speed and congested segments in the major highways in Greater Toronto Area.

The architecture will be discussed in terms of requirements, data management system, analytics system, monitoring system and usage.

Requirements

Shtern et al. classified the four types of urban transportation stakeholders namely: transportation manager, traffic engineer, planner, researcher, and policymaker (Shtern et al., 2014). The requirements are different for each stakeholder. For instance, some are the functional requirements, the others are the non-functional requirements. The Sipresk

architecture considers both functional and non-functional requirements as follows: Scalability and Elasticity, Efficient range scans, Autonomic management, and High Availability.

Data management system

The Sipresk architecture consists of two core systems to meet the functional and the non-functional requirements from the stakeholders. They are Data management system (Figure 7) and Analytics system (Figure 8). Data management system pools traffic data and then stores the data. The data is retrieved from Connected Vehicles and Smart Transportation(CVST) (CVST, 2015) which is a platform to collect data from multiple sources. The raw data is stored in HBase or HDFS depending on the data type and size. On top of the raw data warehouse, Sipresk provides on-demand analytic storages (e.g., key-value, document, wide-column and/or graph stores).

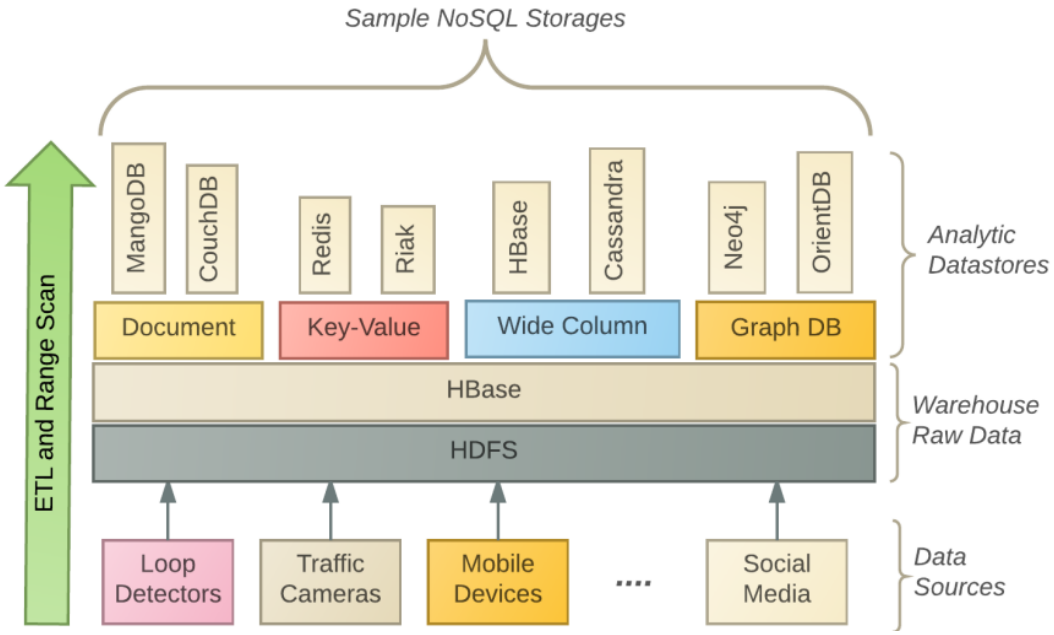

Figure 7: Data management platform in Sipresk (Khazaei et al., 2015)

Analytics system

The analytic system in Sipresk is based on Sahara project contributed by OpenStack foundation[1]. It is able to execute different data processing based on Apache Spark or Hadoop ecosystems. It consists of multiple Big Data tools as follows: Spark offers the distributed processing, R allows developers to implement the codes for the analytics, GraphX provides the iterative graph processing at large scale and more.

[1] http://www.openstack.org

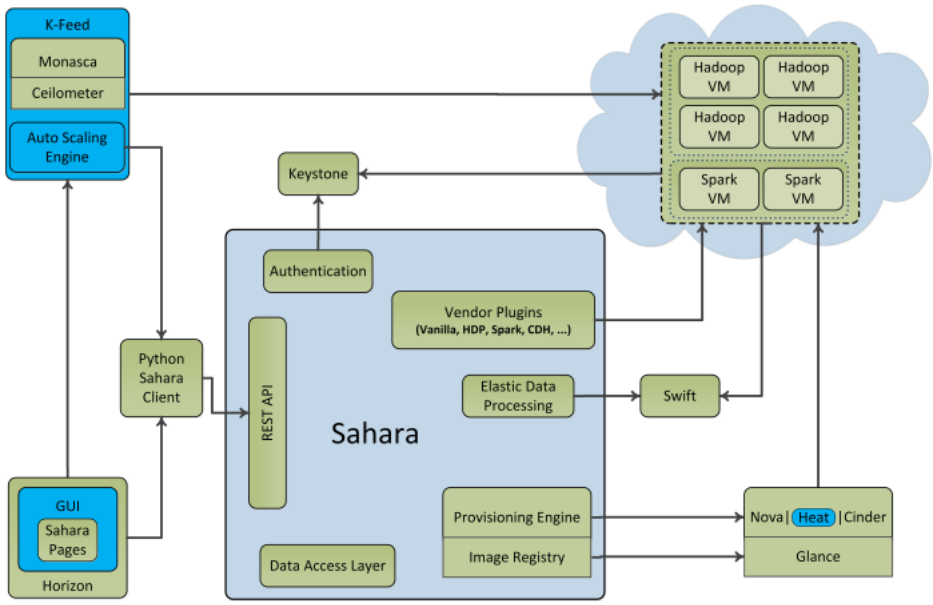

Figure 8: Analytic system in Sipresk (Khazaei et al., 2015)

Monitoring system

Finally, another system is designed to monitor the condition of the two core systems. The high level of the implemented architecture is shown in Figure 9. It is a monitoring that measures the CPU utilization in real-time, and it adds more resources (more worker nodes) if the system is not in a healthy condition (e.g., CPU utilization is higher than 60% for 2 minutes). On top of the health monitoring, it also supports the periodical model deployment.

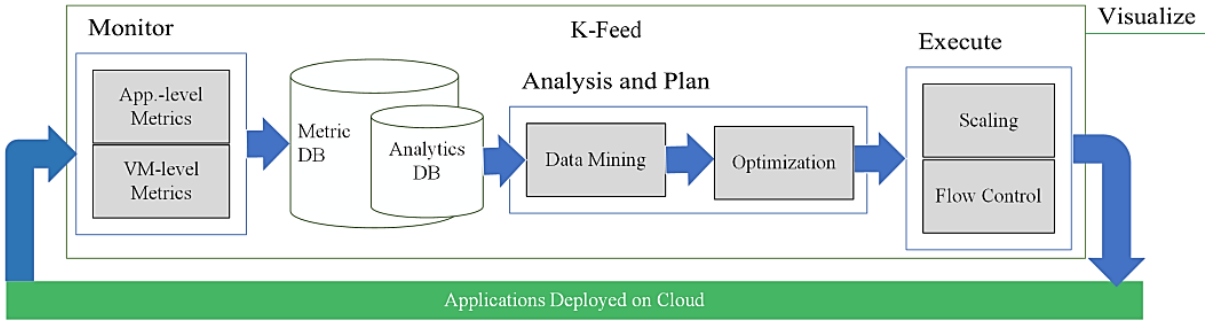

Figure 9: High level architecture of Sipresk (Khazaei et al., 2015)

Usage

An application of traffic congestion in Greater Toronto Area is deployed in the Sipersk architecture for 16 months (the year of 2014 and first 4 months of 2015). It investigates the major highways of Toronto and characterizes the average speed and occupancy. Figure 10 is

one example of the application usage. It shows the congested places on the map. The result is aggregated for Wednesdays in October 2014 during evening rush hours.

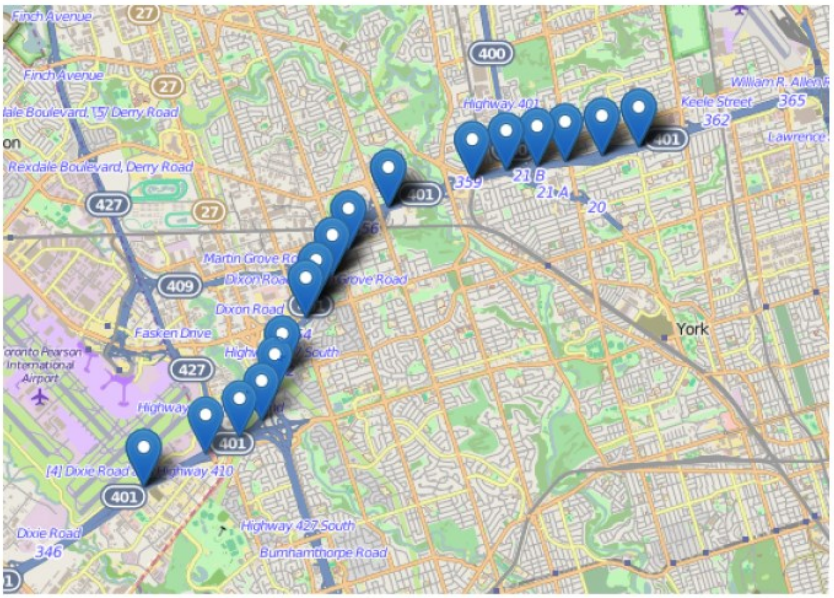

Figure 10: Congested points in 401 West (Khazaei et al., 2015)

4.2 Big Data Analytics Architecture for Real-Time Traffic Control

Amini et al. (2017) proposed and evaluated a high-level Intelligent Transportation System (ITS) architecture with a use case of the real-time traffic control. The proposed architecture is a scalable distributed computing platform and end-to-end pipeline to process data on-the-fly. It consists of numerous state-of-the-art Big Data technologies such as Kafka, HDFS, and NoSQL.

Requirements

The proposed architecture accommodates a number of requirements. Seven identified requirements are divided into two groups: system stability and analytics. For instance, the requirements of the former group are the architecture should be scalable and faults tolerant, on the other hand, the requirements of the latter group are the architecture should support two-speed analytics (batch and streaming analysis), the deterministic setting for analytics, reusable analytics, and integration of multiple data sources.

Architecture

In order to satisfy the requirements above, Amini and coauthors proposed a platform consisting of Big Data technologies. The high-level architecture is depicted in Figure 11. The core components of this architecture are the messaging technology called Kafka and its publisher and subscriber. ITS actors (i.e. drivers, detectors, actuators, operators, etc.) are tightly involved with Kafka by producing and consuming the streaming data.

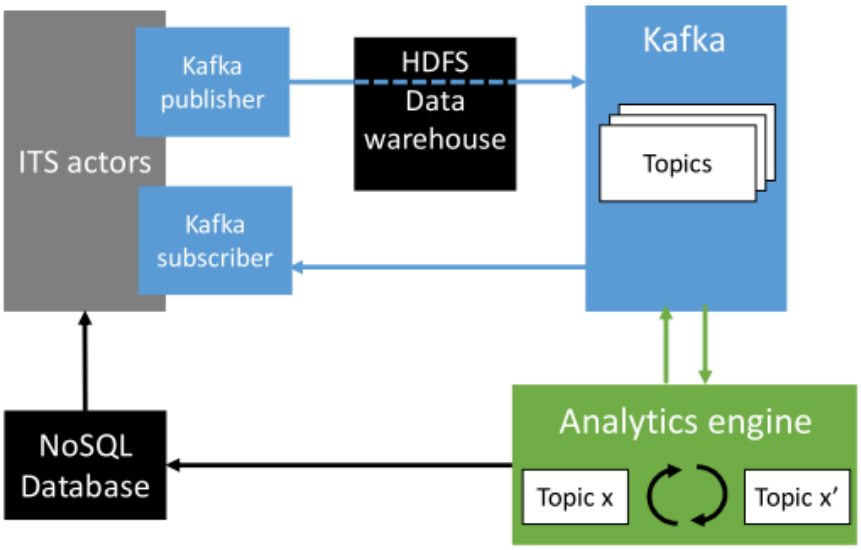

Figure 11: Proposed architecture for real-time traffic control

Usage

The application running in the proposed platform is a hard shoulder opening system. The application determines whether the hard shoulder lane should open or close. Data are collected from the 3 km segment of A9 freeway in the north of Munich. Eight data sources are collected namely: vehicle speed, vehicle position, time, nearby vehicles number, obstacle number, lane occupancy, and mean speed of the lane. The decision is carried out based on these real-time data. Figure 12 is an example of visualization that shows the traffic condition.

Figure 12: Speed information with color indicators (green means fast, red slow) (Krajzewicz et al., 2012)

5 Benchmarking

Widespread Big Data benchmarks will be described in three main categories – dealing with analytics, streaming data, and transportation use cases. This chapter has been published in (Ivanov et al., 2018), (Zicari et al., 2016), and (Ivanov et al., 2015) and selectively reprinted here with permission of the authors.

A benchmark is (Andersen and Pettersen, 1995): "A predefined position, used as a reference point for taking measures against". Jim Gray (Gray, 1992) describes the benchmarking process as follows:"this quantitative comparison starts with the definition of a benchmark or workload. The benchmark is run on several different systems, and the performance and price of each system are measured and recorded. Performance is typically a throughput metric (work/second) and the price is typically a five-year cost-of-ownership metric. Together, they give a price/performance ratio." In short, we define that a software benchmark is a program used for comparison of software products/tools executing on a pre-configured hardware environment according to a set of predefined user defined benchmark specification.

Choosing the right big data platform and configuring it properly to provide the best performance for a hosted application is not a trivial task. Especially with the new big data applications, there are requirements that make the platforms more complex, more heterogeneous, and harder to monitor and maintain. The role of benchmarking becomes even more relevant as a method for evaluating and understanding better the internals of a particular platform. Furthermore, benchmarks are used to compare different systems using both technical and economic metrics that can guide the user in the process of finding the right platform that fits their needs.

Nevertheless, the user has to first identify his needs and then choose the ideal Big Data Benchmarks. Big Data Benchmarks are a good way to optimize and fine-tune the performance in terms of processing speed, execution time or throughput of the big data system. A benchmark can also be used to evaluate the availability and fault-tolerance of a big data system. Especially for distributed big data systems, a high availability is an important requirement.

5.1 Benchmarking Organizations

5.1.1 TPC

The TPC (Transaction Processing Performance Council) (TPC, 2018) is a non-profit corporation operating as an industry consortium of vendors that define transaction processing, database, and big data system benchmarks. TPC was formed on August 10, 1988, by eight companies convinced by Omri Serlin (TPC, 2018). In November 1989 was published the first standard benchmark TPC-A with 42-pages specification (Gray, 1992). By late 1990, there were 35 member companies. As of 2017, TPC has 21 company members and three associate members. There are six obsolete benchmarks (TPC-A, TPC-App, TPC-B, TPC-D, TPC-R and TPC-W), 14 active benchmarks TPC-C (Raab, 1993), TPC-E (Hogan, 2009), TPC-H (Poess and Floyd, 2000), TPC-DS (Poess et al., 2017; Poess et al, 2007; Nambiar and Poess 2006), TPC-DI (Poess et al., 2014), TPC-V (Sethuraman and Taheri, 2010), TPCx-HS (Nambiar, 2014), TPCx-BB (Ghazal et al., 2013) and two common specifications (Pricing and Energy) used across all benchmarks.

5.1.2 SPEC

The SPEC (Standard Performance Evaluation Corporation) (SPEC, 2018) is a non-profit corporation formed to establish, maintain and endorse standardized benchmarks and tools to evaluate performance and energy efficiency for the newest generation of computing systems. It was founded in 1988 by a small number of workstation vendors. The SPEC organization is an umbrella organization that covers four groups (each with their own benchmark suites, rules, and dues structure): the Open Systems Group (OSG), the High-Performance Group (HPG), the Graphics and Workstation Performance Group (GWPG) and the SPEC Research Group (RG).

5.1.3 STAC

The STAC (Securities Technology Analysis Center) Benchmark Council (STAC, 2018) consists of over 300 financial institutions and more than 50 vendor organizations whose purpose is to explore technical challenges and solutions in financial services and to develop technology benchmark standards that are useful to financial organizations. Since 2007, the council is working on benchmarks targeting Fast Data, Big Data and Big Compute workloads in the finance industry.

5.2 Big Data Analytics Benchmarks

This section presents Analytics Big Data benchmarks that are most frequently referenced in current literature. They were developed to stress test and evaluate Big Data systems such as the Hadoop framework and its extensions into the open source ecosystem.

AMP Lab Big Data Benchmark

AMP Lab Benchmark (AMP Lab, 2013) measures the analytical capabilities of data warehousing solutions. This benchmark currently provides quantitative and qualitative comparisons of five data warehouse systems: RedShift, Hive, Stinger/Tez, Shark, and Impala. Based on Pavlo's Benchmark (Pavlo et al., 2009; Stonebraker et al. 2010) and HiBench (Huang et al., 2010; Intel, 2015), it consists of four queries involving scans, aggregations, joins, and user-defined functions (UDFs). It supports different data sizes and scaling to thousands of nodes.

BigBench

BigBench (Baru et al., 2013; Ghazal et al., 2013; BigBench, 2015) is an end-to-end Big Data benchmark that represents a data model simulating the volume, velocity, and variety characteristics of a Big Data system, together with a synthetic data generator for structured, semi-structured, and unstructured data. The structured part of the retail data model is adopted from the TPC-DS benchmark and further extended with semi-structured (registered and the guest user clicks) and unstructured data (product reviews). The BigBench raw data volumes can be dynamically changed based on a scale factor. The simulated workload is based on a set of 30 queries covering the different aspects of Big Data analytics proposed by McKinsey (Manyika et al., 2011). The benchmark consists of four key steps: (i) System setup; (ii) Data generation; (iii) Data load; and (iv) Execute application workload. A reference implementation (BigBench, 2015) for the Hadoop ecosystem is available. Currently, the TPC committee is working towards standardizing it as a TPC Big Data benchmark (Baru et al., 2014).

BigDataBench

BigDataBench (Wang et al., 2014) is an open source Big Data benchmark suite (ICT BigDataBench, 2015) consisting of 14 data sets and 33 workloads. Six of the 14 data sets are real-world based, generated using the Big Data Generator Suite (BDGS) (Ming et al. 2013) data generator. The generated data types include text, graph, and table data, and are fully scalable. According to the literature, it is unclear of what the upper bound of the data set sizes is. The remaining eight datasets are generated from a small seed of real data and are not scalable yet. The 33 workloads are divided into five common application domains: search engine, social networks, electronic commerce, multimedia analytics, and bioinformatics. BigDataBench has many similarities with the DCBench (ICT DCBench, 2013), a benchmark suite developed to test data center workloads. This is a rapidly evolving benchmark. Please check the official website for current updates.

BigFrame

BigFrame (Kunjir et al. 2014) is a benchmark generator offering a benchmarking-as-a-service solution for Big Data analytics. While the latest version together with documentation is available on GitHub (BigFrame, 2013), changes are still being made to the benchmark generator. The benchmark distinguishes between two different analytics workload, 1) one-analytics and 2) real-time analytics. It consists of structured data (Sales, Item, Customer and Promotion tables) adapted from the TPC-DS benchmark and semi-structured JSON data types containing unstructured text.

The current version of the benchmark provides data models for two types of workloads: historical and continuous query. The data in the historical's processed at typical data warehouse rates, e.g., week, whereas the continuous work is processed in real-time. It enables real-time decision making based on instant sales and user feedback updates. The development of mixed workloads combining relational, text and graph data is also in progress.

CloudRank-D

CloudRank-D (Cuo et al., 2012; ICT CloudRank 2013) is a benchmark suite for evaluating the performance of cloud computing systems running Big Data applications. The suite consists of 13 representative data analysis tools, which are designed to address a diverse set of workload data and computation characteristics (i.e., data semantics, data models, and data sizes, the ratio of the size of data input to that of data output). The benchmark suite reports two complementary metrics: data processed per second (DPS) and data processed per Joule (DPJ).

CloudSuite

CloudSuite (Ferdman et al., 2012) is a benchmark suite consisting of both emerging scale-out workloads and traditional benchmarks. The goal of the benchmark suite is to analyze and identify key inefficiencies in the processor's core micro-architecture and memory system organization when running today's cloud workloads. Table 2 summarizes the workload categories as well as the applications that were actually benchmarked.

GridMix

GridMix (Apache Software Foundation GridMix, 2013) is a benchmark suite for Hadoop clusters, which consists of a mix of synthetic jobs. The benchmark suite emulates different users sharing the same cluster resources and submitting different types and number of jobs. This includes also the emulation of distributed cache loads, compression, decompression, and job configuration in terms of resource usage. In order to run the GridMix benchmark a trace describing the mix of all running MapReduce jobs in the given cluster has to be recorded.

Hadoop Workload Examples

Since its first version, the Hadoop framework has included several ready to use MapReduce sample applications. They are located in the Hadoop-examples-version.jar. These applications are commonly used to both learn and benchmark Hadoop. The most popular ones include WordCount, Grep, Pi, and Terasort. The Hibench suite, which is briefly described in the next sub-section, also includes these example workloads. Grep Task Grep (Apache Software Foundation Grep, 2013) is a standard MapReduce program that is included in the major Hadoop distributions. The program extracts strings from text input, matches regular expressions against those strings and counts their number of occurrences. More precisely it consists of two MapReduce jobs running in sequence. The first job counts how many times a matching string occurred, and the second job sorts the matching strings by their frequency and stores the output in a single output. Pi (Apache Hadoop, 2015) is a MapReduce program computing the exact binary digits of the mathematical constant Pi. It uses multiple map tasks to do the computation and a single reducer to gather the results of the mappers. Therefore, the application is more CPU bound and produces very little network and storage I/O.

HiBench

HiBench (Huang et al., 2010; Intel, 2015) is a comprehensive benchmark suite for Hadoop consisting often of workloads including both synthetic micro-benchmarks and real-world applications. HiBench features several ready-to-use benchmarks from 4 categories: microbenchmarks, web search, machine learning, and HDFS benchmarks. The HiBench suite evaluates and characterizes the MapReduce framework in terms of speed (job running time) and throughput (the number of tasks completed per minute) and the HDFS in terms of bandwidth, system resource utilization, and data access patterns. The following list briefly describes the benchmarks currently implemented. For a complete description please refer to (Huang et al., 2010; Intel, 2015).

LinkBench

LinkBench (Armstrong et al., 2013) is a benchmark, developed by Facebook, using the synthetic social graph to emulate social graph workload on top of databases such as MySQL.

MRBench

MRBench (Kim et al. 2008) is a benchmark evaluating the processing of business-oriented queries and concurrent data modifications on MapReduce systems. It implements the 22 queries of the TPC-H decision support system benchmark directly in map and reduce operations. The MRBench supports three configuration options: database size and a number of map and reduce tasks.

MapReduce Benchmark Suite (MRBS)

MRBS (Sangroya et al., 2012a; Sangroya et al., 2012b; MRBS, 2013) is a comprehensive benchmark suite for evaluating the performance of MapReduce systems. The high-level metrics reported by the benchmark are client request latency, throughput, and cost. Additionally, low-level metrics like the size of read-/written data, the throughput of MR jobs, and tasks are also reported. The MRBS implements a service that provides different types of operations, which can be requested by clients. Two execution modes are supported: interactive mode and batch mode. The benchmark runs consists of three phases dynamically configurable by the end-user: warm-up phase, run-time phase, and slow-down phase. The user can specify the number of runs and the different aspects of load: dataload and workload. The dataload is characterized by the size and the nature of the data sets used as inputs for a benchmark, and the workload is characterized by the number of concurrent clients and the distribution of the request type. Domain Application Recommendation Benchmark based on real movie database Business Intelligence TPC-H Bioinformatics DNA sequencing Text Processing Search patterns, word occurrence and sorting on randomly generated text files Data Mining Classifying newsgroup documents into categories, canopy clustering operations

Pavlo's Benchmark (CALDA)

Pavlo's Benchmark (Pavlo et al., 2009; Sherif S., 2015; Apache Hadoop, 2015) consists of five tasks defined as SQL queries among which is the original MapReduce Grep task, which is a representative of most real user MapReduce programs. The benchmark was developed to specifically compare the capabilities of Hadoop with those of commercial parallel Relational Database Management Systems (RDBMS). Although the reported results do not favor the Hadoop platform, the authors remain optimistic that MapReduce systems will coexist with traditional database systems. Table 5 summarizes all types of tasks in Pavlo's Benchmark and their complimentary SQL statements.

PigMix

PigMix/PigMix2 (Baru et al., 2013) is a set of 17 queries specifically created to test the performance of Pig systems. Specifically, it tests the latency and scalability of Pig systems. The queries, written in Pig Latin (Olston et al., 2008), test different operations like data loading, different types of joins, group by clauses, sort clauses, as well as aggregation operations. The benchmark includes eight data sets, with varying schema attributes and sizes, generated using the DataGeneratorHadoop (Apache Hadoop DataGeneratorHadoop, 2015) tool. PigMix/PigMix2 are not considered true benchmarks as they lack some of the main benchmark elements, such as metrics.

PRIMEBALL

PRIMEBALL (Ferrarons et al. 2013) is a novel and unified benchmark specification for comparing the parallel processing frameworks in the context of Big Data applications hosted in the cloud. It is implementation- and technology-agnostic, using a relational news hub called New Pork Times, based on a popular real-life news site. Included are various use-case scenarios made of both queries and data-intensive batch processing. The raw data set is fetched by a crawler and consists of both structured XML and binary audio and video, which can be scaled by a pre-defined scale factor (SF) to 1 PB. The benchmark specifies two main metrics: throughput and

price performance. The throughput metric reports the total time required to execute a particular scenario. The price performance metric is equal to the throughput divided by the price, where the price is defined by the specific cloud provider and depends on multiple factors. Additionally, the benchmark specifies several relevant properties characterizing cloud platforms, such as 1) scale-up; 2) elastic speedup; 3) horizontal scalability; 4) latency; 5) durability; 6) consistency and version handling; 7) availability; 8) concurrency and other data and information retrieval properties.

Statistical Workload Injector for MapReduce (SWIM)

SWIM (Chen et al., 2011; Chen et al., 2012; Yanpei, 2013) is a benchmark, which takes a different approach in the testing process. It consists of a framework, which is able to synthesize representative workload from real MapReduce traces taking into account the job submit time, input data size, and shuffle/input and output/shuffle data ratio. The result is a synthetic workload, which has the exact characteristics of the original workload. Similarly, the benchmark generates artificial data. Then the workload executor runs a script which takes the input data and executes the synthetically generated workload (jobs with specified data size, data ratios, and simulating gaps between the job executions). Additionally, the reproduced workload includes a mix of job submission rates and sequences and a mix of common job types. Currently, the benchmark includes multiple real Facebook traces and the goal is to further extend the repository by including new real workload traces.

TPC-H

TPC-H (TPC Benchmark H, 2014) is the de facto benchmark standard for testing data warehouse capability of a system. Instead of representing the activity of any particular business segment, TPC-H models any industry that manages, sells, or distributes products worldwide (e.g., car rental, food distribution, parts, suppliers, etc.). The benchmark is technology-agnostic. The purpose of TPC-H is to reduce the diversity of operations found in a typical data warehouse application, while retaining the application's essential performance characteristics, namely: the level of system utilization and the complexity of operations. The core of the benchmark is comprised of a set of 22 business queries designed to exercise system functionalities in a manner representative of complex decision support applications. These queries have been given a realistic context, portraying the activity of a wholesale supplier to help the audience relate intuitively to the components of the benchmarks. It also contains two refresh functions (RF1, RF2) modeling the loading of new sales information (RF1) and the purging of stale or obsolete sales information (RF2) from the database. The exact definition of the workload can be found in the latest specification (TPC Benchmark H, 2014). It was adapted very early in the development of Hive and Pig, and implementations of the benchmark are available for both. In order to publish a TPC-H compliant performance result, the system needs to support full ACID (Atomicity, Consistency, Isolation, and Durability).

TPC-DS

TPC-DS (TPC Benchmark DS, 2015) is a decision support benchmark that models several generally applicable aspects of a decision support system, including queries and data maintenance. It takes the marvels of TPC-H and, now obsolete TPC-R, and fuses them into a modern DSS benchmark.

While TPC-DS may be applied to any industry that must transform operational and external data into business intelligence, the workload has been granted a realistic context. It models the decision support tasks of a typical retail product supplier. The goal of selecting a retail business model is to assist the reader in relating intuitively to the components of the benchmark, without tracking that industry segment so tightly as to minimize the relevance of the benchmark. The schema, an aggregate of multiple star schemas, contains essential business information, such as detailed customer, order, and product data for the classic sales channels: store, catalog, and the Internet. Wherever possible, real-world data are used to populate each table with common data skews, such as seasonal sales and frequent names. In order to realistically scale the benchmark from small to large datasets, fact tables scale linearly while dimensions scale sub-linearly. The benchmark abstracts the diversity of operations found in an information analysis application while retaining essential performance characteristics. As it is necessary to execute a great number of queries and data transformations to completely manage any business analysis environment, TPC-DS defines 99 distinct SQL-99 (with Online Analytical Processing (OLAP) amendment) queries and twelve data maintenance operations covering typical DSS like query types such as ad-hoc, reporting, iterative (drill down/up), and extraction queries and periodic refresh of the database. The metric is constructed in a way that favors systems that can overlap query execution with updates (trickle updates). As with TPC-H full ACID characteristics are required. Implementation with more than 50 sample queries is available for Hive (Apache Software Foundation: TPC-H and TPC-DS for Hive, 2015).

TPCx-HS

This section presents the TPCx-HS benchmark, its methodology and some of its major features as described in the current specification (version 1.3.0 from February 19, 2015) [56]. The TPCx-HS was released in July 2014 as the first industry's standard benchmark for Big Data systems (Nambiar et al., 2014). It stresses both the hardware and software components including the Hadoop run-time stack, Hadoop File System, and MapReduce layers. The benchmark is based on the TeraSort workload (Apache Hadoop: TPC Express Benchmark HS, 2015), which is part of the Apache Hadoop distribution. Similarly, it consists of four modules: HSGen, HSDataCkeck, HSSort, and HSValidate. The HSGen is a program that generates the data for a particular Scale Factor (see Clause 4.1 from the TPCx-HS specification) and is based on the TeraGen, which uses a random data generator. The HSDataCheck is a program that checks the compliance of the dataset and replication. The HSSort is a program, based on TeraSort, which sorts the data into a total order. Finally, HSValidate is a program, based on TeraValidate, that validates the output is sorted. A valid benchmark execution consists of separate phases which have to be run sequentially to avoid any phase overlapping. Additionally, Table 7 provides the exact description of each of the execution phases. The benchmark is started by the <TPCx-HS-master> script and consists of two consecutive runs, Run1 and Run2. No activities except the system cleanup are allowed between Run1 and Run2. The completion times of each phase/module (HSGen, HSSort and HSValidate) except HSDataCheck are currently reported. An important requirement of the benchmark is to maintain 3-way data replication throughout the entire experiment. The benchmark reports the total elapsed time (T) in seconds for both runs. This time is used for the calculation of the TPCx-HS performance metric also abbreviated with HSph@SF. The run that takes more time and results in lower TPCx-HS performance metric is defined as the performance run. On the contrary, the run that takes less time and results in TPCx-HS performance metric is

defined as the repeatability run. The benchmark reported performance metric is the TPCx-HS performance metric for the performance run. The scale factor defines the size of the dataset, which is generated by HSGen and used for the benchmark experiments. In TPCx-HS, it follows a stepped size model. Table 8 summarizes the supported scale factors, together with the

Yahoo! Cloud Serving Benchmark (YCSB)

YCSB (Cooper et al., 2015; Patil et al, 2011) a benchmark designed to compare emerging cloud serving systems like Cassandra, HBase, MongoDB, Riak, and many more, which do not support ACID. The benchmark consists of a workload generator and a generic database interface, which can be easily extended to support other relational or NoSQL databases. YCSB provides a core package of six pre-defined workloads A-F, which simulate a cloud Online Transaction Processing (OLTP) application (read and update operations). The reported metrics are execution time and throughput (operations per second). The benchmark is open source and available on GitHub (Yahoo: YCSB, 2015).

5.3 Streaming Benchmarks

StreamBench

StreamBench (Lu, 2014) is a benchmark suite containing seven different micro-benchmark programs. These are Identity, Sample, Projection, Grep, Wordcount, DistinctCount, and Statistics. It consists of four workload suites targeting different technical characteristics of the streaming systems. The simulated applications are using data from real-time weblog processing and network traffic monitoring domain.

Yahoo Streaming Benchmark (YSB)

The Yahoo Streaming benchmark (Alexandrov et al., 2013; AMP Lab: Big Data Benchmark, 2013), a streaming benchmark for an end-to-end processing pipeline based on Apache Kafka (distributed streaming platform), Redis [1] (key-value database) and the three computation engines Flink[2], Storm[3] , and Spark Streaming is presented. The idea of this end-to-end pipeline is, to simulate a real-world advertisement analytics pipeline. Their results show, that Storm and Flink have a lower latency compared to Spark. Spark was, however, able to handle the higher throughput. In contrast to this streaming benchmark, the application benchmark presented by us in this paper uses a more adaptive statistical model and uses the Spark machine learning library for a more complex processing.

HiBench

HiBench (Huang et al., 2010; Intel, 2018) developed by Intel, is a comprehensive benchmark suite for Hadoop that initially was consisting of ten workloads including both synthetic micro-benchmarks and real-world applications. Recently, it was extended with four streaming micro-benchmarks listed in Table 8 and implemented for Kafka, Spark, Storm, Flink, and Gearpump. All four workloads are executed in a similar manner: the streaming app reads data from Kafka,

[1] Redis data store: https://redis.io/

[2] Apache Flink: https://flink.apache.org/

[3] Apache Storm: http://storm.apache.org/

process the data and writes it back to Kafka, labeling each record with a timestamp. The benchmarks report the elapsed time (time difference between reading a record from Kafka and writing the result back to Kafka).

Table 8: Streaming Micro-benchmarks in HiBench

Workload	Description
Identity	This workload reads input data from Kafka and then writes the result to Kafka immediately, there is no complex business logic involved.
Repartition	This workload reads input data from Kafka and changes the level of parallelism by creating more or fewer partitions tests. It tests the efficiency of data shuffles in the streaming frameworks.
Stateful Wordcount	This workload counts words cumulatively received from Kafka every few seconds. This tests the stateful operator performance and Checkpoint/Acker cost in the streaming frameworks.
Fixwindow	The workloads perform a window based aggregation. It tests the performance of window operation in the streaming frameworks.

SparkBench

SparkBench[1] (version 2.0) (Li et al., 2015) is another Spark specific benchmark suite developed by IBM, which provides representative workloads in four categories as listed in Table 9. The purpose of the benchmark suite is to help users evaluate and analyze the tradeoffs between different system designs, guide the optimization of workload configurations and cluster provisioning for Spark deployments. SparkBench reports two metrics: job execution time (seconds) and data process rate (MB/second). The job execution time measures the execution time of each workload, whereas the data process rate is defined as the input data size divided by the job execution time.

Table 9: SparkBench Workloads

Category	Workload	Input Dataset	Library/Tool
Machine Learning	Logistic Regression	Wikipedia	MLLib
	Support Vector Machine	Wikipedia	
	Matrix Factorization	Amazon Movie Review	
Graph Computation	PageRank	Google Web Graph	GraphX
	SVD++	Amazon Movie Review	
	TriangleCount	Amazon Movie Review	
SQL Query	Hive	E-commerce	Hive-on-Spark
	RDDRelation	E-commerce	SparkSQL
Streaming Application	Twitter	Twitter	DStream
	PageView	PageView DataGen	

[1] https://github.com/CODAIT/spark-bench

5.4 Benchmarks in the Transport Sector

This section introduces three benchmarks in the transport sector. One may consider taking this benchmark as a best practice implementation if it covers the same use case scenario of what one plans to implement. The overview of three benchmarks is depicted in Table 10. The type of benchmarks and the data characteristics which is described in the LeMO deliverable (Hee et al, 2018).

Table 10: Existing benchmarks in the transport sector

Benchmark name	Type			Data Characteristics					
	Micro benchmark	Application benchmark	Benchmark Specification	Sensor data	Tracking data	Social network data	Crowd-sourced data	Public Data	3rd Party Data
RIoTBench	x	x						x	
Linear Road			x	x					
ShenZhen (SZTS)	x			x					

Linear Road: A Stream Data Management Benchmark

(Arasu, 2004) specifies the Linear Road Benchmark for Stream Data Management Systems (SDMS). The application running on top of this benchmark is called Linear Road. It is a tolling system that uses variable tolling (Yang, 1996), which can dynamically calculate toll charges depending on multiple factors as traffic congestion and accident. SDMS is able to process streaming data, which are the common data source in the transport sector. The goal of this benchmark is to compare the performance of SDMS' relative to each other and to conventional database systems. The results show that a dedicated SDMS can outperform a Relational Database by at least a factor of 5 on streaming data applications.

ShenZhen transportation system (SZTS): a novel big data benchmark suite

ShenZhen Transportation System (SZTS) (Xiong, 2016) is a big data Hadoop benchmark suite comprised of real-life transportation analysis applications with real-life input data sets from Shenzhen in China. SZTS targets a real-life application domain unlike other Hadoop benchmark suites (e.g. HiBench or CloudRank-D) consist of generic algorithms with synthetic inputs. The workloads of SZTS are across several layers namely: the microarchitecture level, the operating system (OS) level, and the job level.

RIoTBench

RIoTBench (Shukla et al., 2017) is a Real-time IoT Benchmark suite, consisting of 27 IoT micro-benchmarks and 4 real-application benchmarks reusing the micro-benchmark components, along with performance metrics. The goal of the benchmark suite is to evaluate the efficacy and performance of Distributed Stream Processing Systems (DSPS) in cloud environments. The benchmark is currently implemented on Apache Strom and available online.

5.5 Recommendations

Utilizing an existing Big Data benchmarking has many advantages. The major benefit is the reusability. One can reuse the benchmark specification and guidelines on how to use the benchmark and how to stress specific technology functionalities in the case of micro-benchmarks. For instance, if one is new to a particular Big Data technology, the best approach will be to refer to an existing Big Data benchmark implemented in the technology. By doing so, you can learn how to deploy, use and understand if it will be the best choice for your application. Similarly, if a benchmark covers the same domain or the same use case scenario of what one plans to implement, one can use it as a best practice implementation and investigate potential problems before starting own implementation.

6 Custom Benchmarking

Numerous benchmarks are available as shown in Chapter 5. However, benchmarks and Big Data technologies are changing fast, thus, it is hard to keep an up-to-date overview of all emerging benchmarks. In addition, often missing guidelines and user documentation are the main factors that hinder the use of the existing benchmark. In other cases, the official documentation is well written, but there is no publicly available implementation. These push developers and practitioners to develop their own benchmarking method. Thus, we propose a guideline on how to set up a Big Data application and how to convert it into a benchmark workload.

6.1 Application: road traffic condition in New York City (NYC)

This section presents an implementation of a system that supports the better transport planning as the demand for the Intelligent Transportation Systems increases. For instance, vehicle control systems and road transportation systems are the most popular Big Data application according to the two studies (An et al., 2011; Zhang et al., 2011). Similarly, Gartner predicted that a quarter billion connected cars will be on the road, becoming one of the major elements in the Internet of Things by 2020 (Gartner, 2015). The following sections explain the proposed application, starting with the application requirements, data description, data modeling, used architecture and the outputs of the application.

6.1.1 Requirements

The goal of the application is to provide better services for drivers and passengers on the one hand, and for the city to be aware of the traffic condition considering time information on the other hand. For instance, the application answers questions such as: “which hour of a day has the most pickup call in NYC?” or “which places of NYC have the most taxi pickup demands over 24 hours?” This is the descriptive analytics which provides the answer to the question “What happened?” It aims to offer a better understanding of the given data.

6.1.2 Data description

Uber data is publicly available[1] and the repository contains more than 4.5 million taxi pickups. The geographic coverage is the New York City and the temporal coverage is 6 months from April to September 2014. It consists of four attributes namely: *Date/Time* (the date and time of the Uber pickup), *Lat* (the latitude of the Uber pickup), *Lon* (the longitude of the Uber pickup) and *Base* (the TLC[2] base company code affiliated with the Uber pickup). An example record is: [2014-04-01 00:00:00, 40.729, -73.9422, B02598]

6.1.3 Data modeling

There are numerous methodologies that can be applied for the descriptive analytics. The case study in this section focuses on cluster analysis which is one of the most well-known methods for the descriptive analytics. Specifically, the application utilizes one of the clustering algorithms

[1] http://data.beta.nyc/dataset/uber-trip-data-foiled-apr-sep-2014

[2] http://www.nyc.gov/html/tlc/html/industry/base_and_business.shtml

called K-Means (MacQueen, 1967). K-Means constructs a subset of an input dataset into a predefined number of clusters. The computational complexity of K-Means is known as $O(n^2)$ with n data samples.

6.1.4 Architecture

The proposed architecture is inspired by the Lambda architecture (Marz, 2012). The architecture incorporates numerous Big Data technologies running on Apache Hadoop (section 3.3 and 3.4) as Hadoop Ecosystem has become the de facto platform for Big Data. The applied technologies are Kafka, Spark, Spark MLlib, Spark Streaming, HDFS, Python and Python libraries. Figure 13 illustrates that the proposed architecture with two layers. It is able to store and process a large amount of data in the distributed system. This architecture (in the color of red) is able to cope with the challenges of velocity and volume. In addition, the batch layer (in the color of yellow) can be used as a standalone. It can be executed in an independent manner to obtain a static model.

For our case study, Uber taxis generate the streaming data of the pick-up call. Kafka (the messenger) then ingests the data (the data producer) on the fly, and it pushes the data into two consumer components namely: Spark Streaming and HDFS. The data stored in HDFS are periodically triggered to create a model (the K-Means model). Meanwhile, this periodically updated model is deployed to the Spark Streaming. The former is called the static modeling, while the latter is called adaptive modeling. The Spark Streaming assigns new arriving data to the deployed model and then the visualization component takes the output to plot the taxi pickup data on the map.

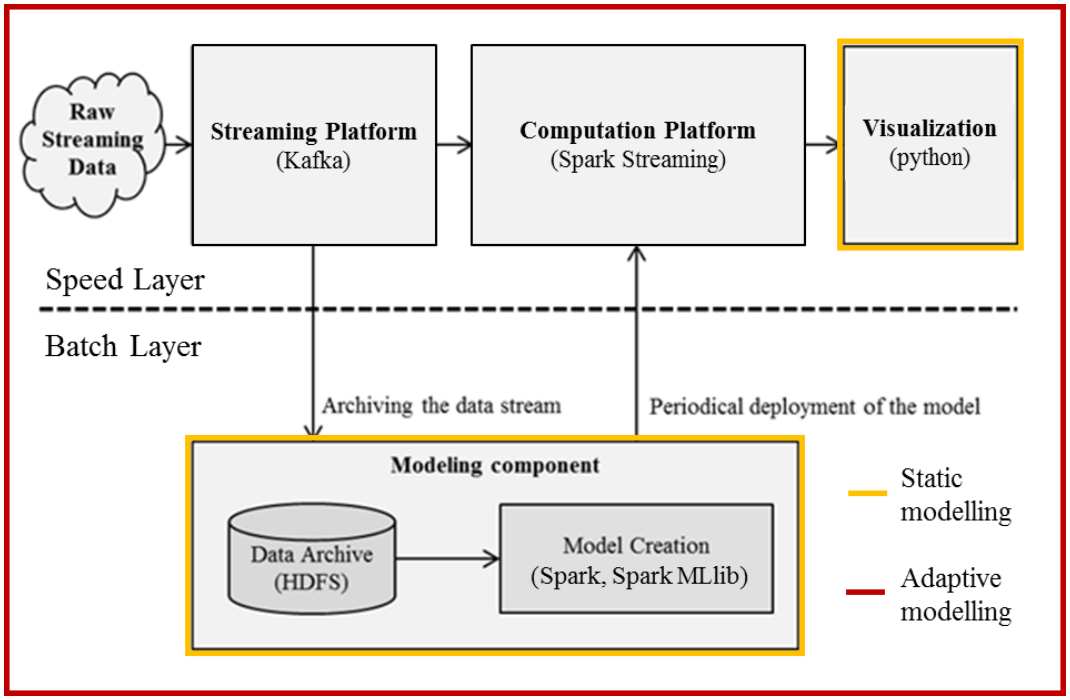

Figure 13: Architecture for Adaptive modeling with two layers

6.1.5 Results and usage

Figure 14 is the K-Means visualization of Uber pickup data in August 2014. A data point on the map represents every pickup location of Uber taxis. The number of the centroid is set as eight (parameter k = 8), thus a data point is highlighted in one of eight colors corresponding to the centroid which the data point belongs to. Note that the proposed architecture is able to generate numerous outputs because it supports the adaptive modeling which is refreshed based on the predefined time interval. Thus, the position of centroids is continuously changing corresponding to the periodical deployment of the static model. Figure 15 shows the summary view of pickup counts for each cluster. The cluster is shown on the x-axis, the y-axis is the count. Most taxis are picked up in the Manhattan borough where is the most densely populated of New York City.

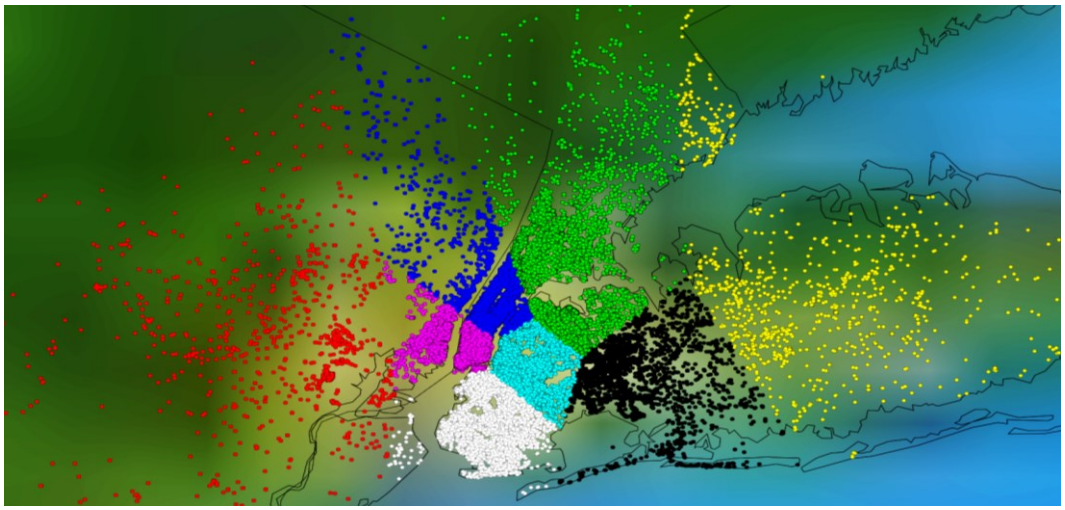

Figure 14: K-means visualization of Uber pickup data in NYC

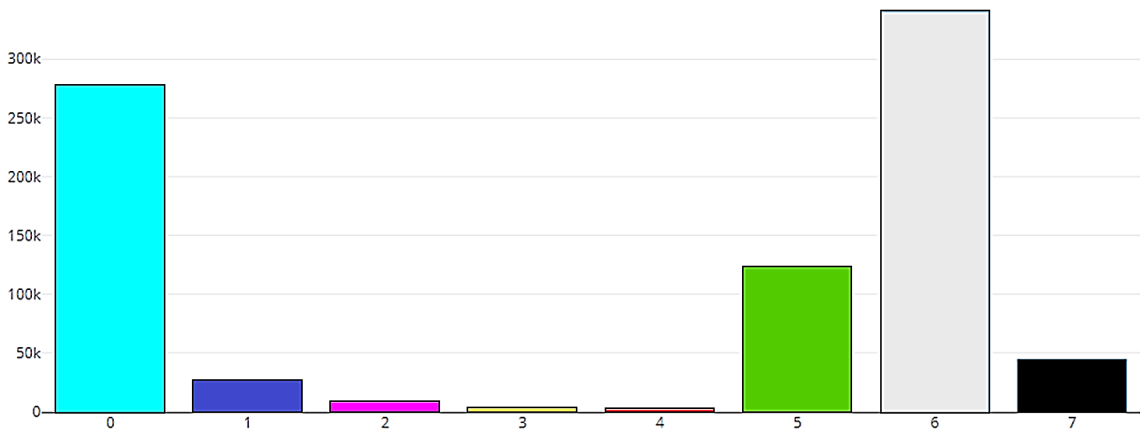

Figure 15: Uber pickups count for each cluster

Figure 16 is another usage of the same Uber data with K-Means model. This output is generated by MapR and this software company is one of the major Hadoop distribution providers. The x-axis shows the hour, the y-axis is the count. This visualization shows that which hours of the day and which cluster the highest number of pickups had. Figure 15 and Figure 16 can provide the actionable value to the taxi driver and the city planner.

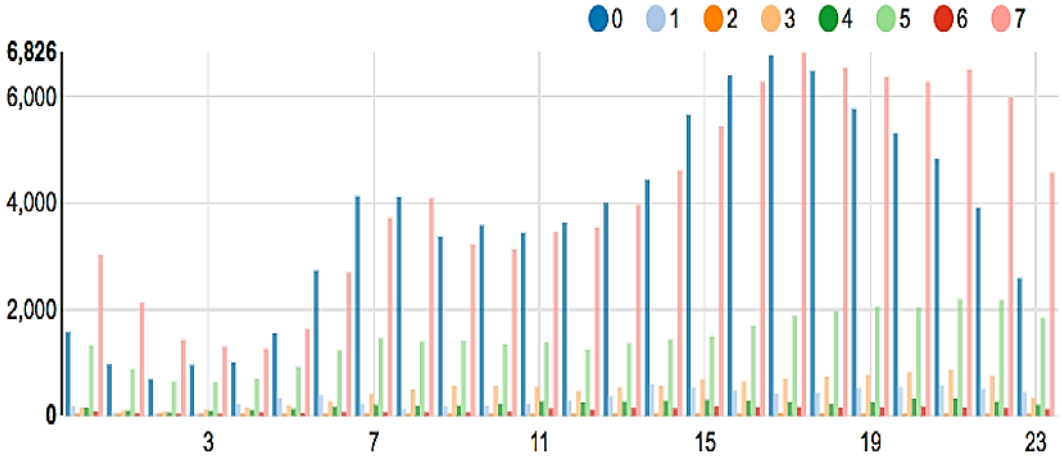

Figure 16: Number of pickups over 24 hours for each cluster [1]

6.2 Turning it into a benchmark

This section converts the proposed application into a benchmark workload. In other words, it is an application benchmark with focus on the Big Data technologies. In particular, the transformed benchmark is designed to analyze the Spark and Spark MLlib which are shown as the modeling component in Figure 13. Spark is chosen to be analyzed because Spark is the de facto choice for stream processing and machine learning (Venugopal & Gualtieri, 2018). Despite Spark users do not need to know the internal procedure behind the API, our application benchmark takes a deep look into each API function call and tries to identify the bottleneck at the function level.

6.2.1 Input datasets

Five data sets are prepared to test the experiment. The entire temporal coverage is 6 months from April to September 2014. And the data description is explained in section 6.

1. (54.8 MB), 2 months (2014/04/01 - 2014/05/31)
2. (84.7 MB), 3 months (2014/04/01 - 2014/06/30)
3. (120.7 MB), 4 months (2014/04/01 - 2014/07/31)
4. (158.0 MB), 5 months (2014/04/01 - 2014/08/31)
5. (204.3 MB), 6 months (2014/04/01 - 2014/09/30)

[1] https://mapr.com/blog/monitoring-real-time-uber-data-using-spark-machine-learning-streaming-and-kafka-api-part-1/

6.2.2 Setup

The experiments were performed on a cluster consisting of 4 nodes connected directly through 1GBit Netgear switch. All 4 nodes are Dell PowerEdge T420 servers. The master node is equipped with 2x Intel Xeon E5-2420 (1.9GHz) CPUs, on the other hand, the worker nodes are equipped with 1x Intel Xeon E5-2420 (2.20GHz) CPU. Besides the CPU, other settings are the same. Each server is equipped with 6 cores, 12 Threads, 32GB of RAM and 1TB (SATA, 3.5 in, 7.2K RPM, 64MB Cache) hard drive. Note that only two worker nodes were used to run the experiments because of the data size.

6.2.3 Metrics

The benchmark metric is mean to evaluate and compare the performance of multiple software or hardware systems stressed with the same benchmark workload. *Throughput* and *Execution time* are used in almost every benchmark, while QphH@SF or QphDS@SF are specific metrics defined for a particular benchmark. Table 11 lists six common benchmark metrics suitable especially for stream processing. There are two Latency metrics namely: *Execution time* and *End-to-end execution.* The former is the elapsed time in seconds for a function, while the latter varies in definition depending on the components that are part of the system under test. For example, in the case of stream processing, it should also include the time taken from data/event creation until the stream processing platform emits the result. Because of the simplicity of our scenario, we report only Execution time.

Table 11: Six benchmark metrics

Metric Name	Description
Execution Time	Execution Time (Latency) is the time between the start and end of the query execution against the streaming data.
End-to-end execution	The End-to-end execution (Latency) include the network and queuing times from sending to receiving a tuple and completely processing it until the final result is outputted.
Throughput	Throughput is the rate of successfully delivered resources per time interval and is measured in messages processed per second. Example from TPC-H: Throughput@Size = S ∗ 22/(TS /3600) ∗SF where S = number of query streams, TS = elapsed time of test (in seconds) and SF = Scale factor.
Jitter	Jitter tracks the variation in the output throughput from the expected output average throughput.
Resource Utilization (CPU%, Memory%, etc.)	Resource Utilization is measured typically by the CPU, Memory, Disk Utilization and Network of the underlying operating system when executing the specified query. Measuring and analyzing these parameters across all cluster nodes helps to identify potential bottlenecks and perform system optimizations.
Price/Performance	$/QphDS@SF where $ = total operating costs including the price of the System under Test, QphDS = Query per hour for decision support, and SF = Scale factor.

6.2.4 Experiment design

The goal is to measure the execution time of each function call and find out the bottleneck function call using the five different input datasets. Figure 17 is a data pipeline that intuitively visualizes how data flow when the K-Means model is created.

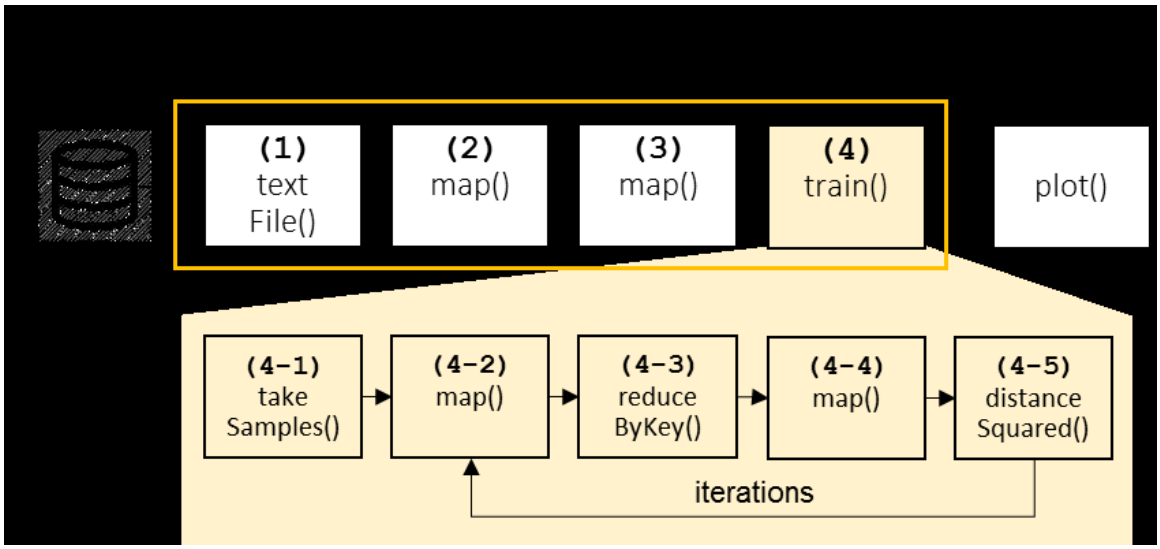

Figure 17: Dataflow of the K-Means modeling

The entire pipeline consists of three Spark functions at the block (1), (2), (3), and one Spark MLlib function at the block (4). The last block (4) can be further broken down into five sub-function calls as shown at the block (4-1), (4-2), (4-3), (4-4), and (4-5). A loop is executed between (4-1) to (4-5) until it merges with the threshold criterion.

The role of each function is described as follows:

(1) textFile(): reads the raw data from HDFS and converts to the RDD format
(2) map(): breaks the raw data into a line by line structure
(3) map(): extracts only relevant fields namely, latitude and longitude
(4) train(): creates the K-Means model over the data

(4-1) takeSamples(): selects random K centroids from the input data
(4-2) map(): finds the nearest centroid for every data sample
(4-3) reduceByKey(): merges the number of data samples for each key (K centroids)
(4-4) map(): find a new centroid by averaging each point of a cluster
(4-5) distanceSquared(): calculates the squared distances between two points (the new centroid and data sample)

The experiment is designed to run three times and averaged by three. The hyper-parameters are set as follows: the number of centroids (K == 8), the threshold distance at the iterations (convergeDistance == 0), and the max number of iterations (maxNumIteration == 10).

6.2.5 Benchmark results

The average of three experiment execution times is shown in Table 12. Except for the first column (Size) and 10th columns (Iterations), the unit is seconds. The last column of Total is the total execution time. It is the sum of the first four functions added to the sum of rest functions multiplied by the number of iterations, namely:

$$Total = \{(1) + (2) + (3) + (4\_1)\} + \{(4\_2) + (4\_3) + (4\_4) + (4\_5)\} * Iterations$$

Table 12: Execution time in seconds

Size (MB)	(1) Load	(2) Map1	(3) Map2	(4-1) take Sample()	(4-2) map()	(4-3) reduce ByKey()	(4-4) map()	(4-5) distance Squared()	Iterations	Total
54.0	1.38	0.00	0.00	32.321	0.000	0.109	10.933	0.000	3	66.827
84.7	1.79	0.00	0.00	37.701	0.000	0.062	16.609	0.000	**8**	**172.859**
120.7	1.16	0.00	0.00	43.698	0.000	0.057	23.315	0.000	**5**	**161.718**
158.0	1.14	0.00	0.00	53.161	0.000	0.047	33.502	0.000	4	188.497
204.3	1.13	0.00	0.00	60.661	0.000	0.047	42.226	0.000	4	230.883

It is noteworthy to mention that the **total execution time is not linearly increasing** unlike the size of the input data. In particular, the results in the red box caught our attention. It is against common sense because the total execution time of the smaller dataset (84.7 MB) is bigger than the larger dataset (120.7 MB). The answer for this phenomenon is **due to the number of Iterations** which is determined based on the randomness of the K-Means clustering algorithm. Thus, we recommend to one to lower the number of hyper-parameter of maxNumIteration. With this setting, the level of randomness will be decreased and more stable IT service can be provided.

6.3 Recommendations

This chapter provided a practical guideline on how one can turn an application to a benchmark. The entire procedure is divided into two parts: (1) setting up a custom application and then (2) validate that it satisfies all the requirements. The biggest advantage of developing customized application benchmark is that it can address the specific domain requirements including the data type and size, best methodologies, architectures and Big Data technologies for this particular scenario. Additionally, one can define custom metrics to monitor particular functions of the implemented components. It is also much easier to deploy, use and extend a software that was developed internally by your team.

In terms of outcomes, based on the results, you will be able to take technical and business decisions about the platform very accurately. However, there are also potential pitfalls. It takes resources and specialized knowledge to develop a customized benchmark. In addition, it is not an industry standard and as such has limited marketing role for a software product or service positioning against competitors.

7 Conclusions

The use of Big Data has the potential to change dramatically the way transportation is done. The technologies needed to store, manage and analyse Big Data are complex and require high skills and expertise. Vendors, especially in the USA, are leading the game, by offering various software data platforms, which integrate both the storage and management of data together with advanced data analytics. To choose the right technology for the domain at hand is a difficult task which requires a high knowledge of the software technologies provided and the requirements of the application.

This document offers the reader a technical insight into existing Big Data technologies at various levels: software management, data platform, and application. In order to evaluate which specific software components in the Big Data stack are more suitable for transport applications, with high volume and high-velocity requirements, a benchmarking approach is presented.

This report introduces general Big Data benchmarks (such as TPC, SPEC, STAC), and streaming benchmarks (such as HiBench, YSB, StreamBench). Few data benchmarks currently exist that are focused on Transport. We recommend in this report to define custom application benchmarks for transportation using the best of the breed Big Data software technologies available. In the report, we have shown an example of such a custom application benchmark, a data-driven application for road traffic evaluation.

The future of data analytics in transportation has many applications and opportunities.

The main challenge is using significantly improved technologies and methods to gather and understand the data in order for business decisions to be informed by better insights.

References

Abouzeid A.; Bajda-Pawlikowski C.; Abadi D.; Silberschatz A.; Rasin A., "HadoopDB: An Architectural Hybrid of MapReduce and DBMS Technologies for Analytical Workloads", Proceedings of the VLDB Endowment, Pages 922-933, Volume 2, Number 1, August 2009.

Alexandrov, A., Brucke, C., & Markl, V. (2013). Issues in big data testing and benchmarking. Proceedings of the Sixth International Workshop on Testing Database Systems. DBTest 2013, New York, USA, June 24, pp. 1:1-1:5

Amini, S., Gerostathopoulos, I., & Prehofer, C. (2017, June). Big data analytics architecture for real-time traffic control. In Models and Technologies for Intelligent Transportation Systems (MT-ITS), 2017 5th IEEE International Conference on (pp. 710-715). IEEE.

AMP Lab (2013). AMP Lab Big Data Benchmark. Available at: https://amplab.cs.berkeley.edu/benchmark/

An, S. H., Lee, B. H., & Shin, D. R. (2011, July). A survey of intelligent transportation systems. In Computational Intelligence, Communication Systems and Networks (CICSyN), 2011 Third International Conference on (pp. 332-337). IEEE.

Anand, S. S., Grobelnik, M., Herrmann, F., Hornick, M., Lingenfelder, C., Rooney, N., & Wettschereck, D. (2007). Knowledge discovery standards. Artificial Intelligence Review, 27(1), 21-56.

Andersen B, Pettersen PG (1995) Benchmarking handbook. Champman & Hall

Andrew Pavlo (2011). Benchmark. Available at: http://database.cs.brown.edu/projects/mapreducevs-dbms/

Apache Hadoop (2015). Package org.apache.hadoop.examples.pi. Available at: http://hadoop.apache.org/docs/r0.23.11/api/org/apache/hadoop/examples/pi/package-summary.html

Apache Hadoop (2015). TPC Express Benchmark HS - Standard Specication. Available at: http://hadoop.apache.org/docs/current/api/org/apache/hadoop/examples/terasort/package-summary.html

Apache Software Foundation (2009). Grep. Available at: http://wiki.apache.org/hadoop/Grep

Apache Software Foundation (2010). DataGeneratorHadoop. Available at: http://wiki.apache.org/pig/DataGeneratorHadoop

Apache Software Foundation (2012). Running TPC-H Benchmark on Pig. Available at: https://issues.apache.org/jira/browse/PIG-2397

Apache Software Foundation (2013). GridMix. Available at: https://hadoop.apache.org/docs/stable1/gridmix.html

Apache Software Foundation (2013). Hive performance benchmarks. Available at: https://issues.apache.org/jira/browse/HIVE-396

Apache Software Foundation (2013). PigMix. Available at: https://cwiki.apache.org/conuence/display/PIG/PigMix

Apache Software Foundation (2015). TPC-H and TPC-DS for Hive. Available at: https://github.com/hortonworks/hive-testbench/tree/hive14

Arasu, A., Cherniack, M., Galvez, E., Maier, D., Maskey, A. S., Ryvkina, E., ... & Tibbetts, R. (2004, August). Linear road: a stream data management benchmark. In Proceedings of the Thirtieth international conference on Very large data bases-Volume 30 (pp. 480-491). VLDB Endowment.

Armstrong, T. G., Ponnekanti, V., Borthakur, D., & Callaghan, M. (2013, June). LinkBench: a database benchmark based on the Facebook social graph. In Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data (pp. 1185-1196). ACM.

Baru, C., Bhandarkar, M., Curino, C., Danisch, M., Frank, M., Gowda, B.,Jacobsen, H., Jie, H., Kumar, D., Nambiar, R., Poess, M., Raab, F., Rabl, T., Ravi, N., Sachs, K., Sen, S., Yi, L., & Youn, C. (2014). Discussion of bigbench: A proposed industry standard performance benchmark for big data. Performance Characterization and Benchmarking. Traditional to Big Data - 6th TPC Technology Conference, TPCTC 2014, Hangzhou, China, September 1-5, Revised Selected Papers. pp. 44-63

Baru, C., Bhandarkar, M., Nambiar, R., Poess, M., & Rabl, T. (2013). Setting the Direction for Big Data Benchmark Standards. Nambiar, R., Poess, M. (eds.) Selected Topics in Performance Evaluation and Benchmarking. Lecture Notes in Computer Science, Springer Berlin Heidelberg. Volume 7755, pp. 197-208.

BigBench (2015). Available at: https://github.com/intel-hadoop/Big-Data-Benchmark-for-Big-Bench

BigFrame Team (2013). Available at: https://github.com/bigframeteam/BigFrame/wiki

Bouillet, E., Feblowitz, M., Liu, Z., Ranganathan, A., Riabov, A., Ye, F., ... & Schlosnagle, D. (2007, September). Data stream processing infrastructure for intelligent transport systems. In Vehicular Technology Conference, 2007. VTC-2007 Fall. 2007 IEEE 66th (pp. 1421-1425). IEEE.

BSC (2014). Aloja home page. Available at: http://aloja.bsc.es/

Chang, J., Lim, K.T., Byrne, J., Ramirez, L., & Ranganathan, P. (2012). Workload diversity and dynamics in big data analytics: Implications to system designers. Proceedings of the 2Nd Workshop on Architectures and Systems for Big Data. ASBD '12, ACM, New York, USA. pp. 21-26.

Chaolong, J., Hanning, W., & Lili, W. (2016). Study of Smart Transportation Data Center Virtualization Based on VMware vSphere and Parallel Continuous Query Algorithm over Massive Data Streams. Procedia engineering, 137, 719-728.

Chen, C. Y., Chou, T. Y., Mu, C. Y., Lee, B. J., & Magesh Chandramouli, H. C. (2005, July). Using Data Mining Techniques on Fleet Management System. In 2004 ESRI International User Conference Agenda, USA http://gis2. esri. com/library/userconf/proc04/docs/pap1801. pdf.

Chen, Y. (2012). We dont know enough to make a big data benchmark suite-an academia industry view. Technical Report No. UCB/EECS-2012-71

Chen, Y., Alspaugh, S., & Katz, R. (2012). Iteractive analytical processing in big data systems: A cross-industry study of mapreduce workloads. PVLDB 5(12), 1802-1813

Chen, Y., Ganapathi, A., Grith, R., & Katz, R. (2012). The case for evaluating mapreduce performance using workload suites. In: MASCOTS 2011, 19th Annual IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, Singapore, 25-27 July. pp. 390-399

Chen, Y., Raab, F., & Katz, R. (2012). From TPC-C to big data benchmarks: A functional workload model. In: Specifying Big Data Benchmarks - First Workshop, WBDB 2012, San Jose, CA, USA, May 8-9, and Second Workshop, WBDB 2012, Pune, India, December 17-18, Revised Selected Papers. pp. 28-43 (2012)

Chowdhury, M., Apon, A., & Dey, K. (Eds.). (2017). Data analytics for intelligent transportation systems. Elsevier.

Codd EF, Codd SB, Salley CT (1993) Providing OLAP (On-Line Analytical Processing) to User-Analysis: An IT Mandate. White paper

Cooper, B.F., Silberstein, A., Tam, E., Ramakrishnan, R., & Sears, R. (2010). Benchmarking cloud serving systems with YCSB. Proceedings of the 1st ACM Symposium on Cloud Computing, SoCC 2010, Indianapolis, Indiana, USA, June 10-11, 2010. pp.143-154

Cristóbal, T., Padrón, G., Quesada-Arencibia, A., Alayón, F., & García, C. R. (2018). Systematic Approach to Analyze Travel Time in Road-based Mass Transit Systems Based on Data Mining. IEEE Access.

CVST (June 2018). Connected Vehicles and Smart Transportation, Available at: http://cvst.ca.

Data. 6th TPC Technology Conference, TPCTC 2014, Hangzhou, China, September 1-5. Revised Selected Papers. pp. 1-12

Almeida, J., & Ferreira, J. (2013, May). BUS public transportation system fuel efficiency patterns. In 2nd International Conference on Machine Learning and Computer Science (IMLCS'2013), Kuala Lumpur, Malaysia.

Dimitrov, M., Kumar, K., Lu, P., Viswanathan, V., & Willhalm, T. Memory system characterization of big data workloads. Proceedings of the 2013 IEEE International Conference on Big Data, 6-9 October 2013, Santa Clara, CA, USA. pp. 15-22

Eurostat (2017). Transport indicators in Energy, transport and environment indicators. European Union. pp. 93-139

Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). The KDD process for extracting useful knowledge from volumes of data. Communications of the ACM, 39(11), 27-34.

Ferdman, M., Adileh, A., Kocberber, Y., Volos, S., Alisafaee, M., Jevdjic, D., Kaynak, C., Popescu, A., Ailamaki, A., & Falsa, B. (2012). Clearing the clouds: a study of emerging scale-out workloads on modern hardware. Proceedings of the 17th International Conference on Architectural Support for Programming Languages and Operating Systems, ASPLOS 2012, London, UK, March 3-7. pp. 37-48

Ferrarons, J., Adhana, M., Colmenares, C., Pietrowska, S., Bentayeb, F., & Darmont,J. (2013) PRIMEBALL: A parallel processing framework benchmark for big data applications in the cloud. Performance Characterization and Benchmarking - 5th TPC Technology Conference, TPCTC 2013, Trento, Italy, August 26, Revised Selected Papers. pp. 109-124

Gandomi, A., & Haider, M. (2015). Beyond the hype: Big data concepts, methods, and analytics. International Journal of Information Management. Volume 35. pp. 137-144

Gartner (2015). Connected Cars Will Form a Major Element of the Internet of Things. Available at: https://www.gartner.com/newsroom/id/2970017.

Ghazal, A., Rabl, T., Hu, M., Raab, F., Poess, M., Crolotte, A., & Jacobsen., H. (2013) BigBench. Towards an Industry Standard Benchmark for Big Data Analytics. SIGMOD

Ghofrani, F., He, Q., Goverde, R. M., & Liu, X. (2018). Recent applications of big data analytics in railway transportation systems: A survey. Transportation Research Part C: Emerging Technologies, 90, 226-246.

Goebel, M., & Gruenwald, L. (1999). A survey of data mining and knowledge discovery software tools. *ACM SIGKDD explorations newsletter*, *1*(1), 20-33.

Gray J (1992) Benchmark Handbook: For Database and Transaction Processing Systems. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA

Greis, N. P., & Nogueira, M. L. (2011). Use of data mining for validation and verification of maritime cargo movement. Institute for Homeland Security Solutions (Research Brief). North Carolina: University of North Carolina.

Guerreiro, G., Figueiras, P., Silva, R., Costa, R., & Jardim-Goncalves, R. (2016, September). An architecture for big data processing on intelligent transportation systems. An application scenario on highway traffic flows. In Intelligent Systems (IS), 2016 IEEE 8th International Conference on (pp. 65-72). IEEE.

Haluzová, P. (2008). Effective data mining for a transportation information system. Acta Polytechnica, 48(1), 24-29.

Hand, D. J., Mannila, H., & Smyth, P. (2001). Principles of data mining (adaptive computation and machine learning) (pp. 361-452). Cambridge, MA: MIT press.

Hashem, I. A. T., Yaqoob, I., Anuar, N. B., Mokhtar, S., Gani, A., & Khan, S. U. (2015). The rise of "big data" on cloud computing: Review and open research issues. Information Systems, 47, 98-115.

Hee, K., Mushtaq, N., Özmen, H., Rosselli, M., Zicari, R., Hong, M., Akerkar, R., Roizard, S., Russotto, R., & Teoh, T., (2018). Deliverable 1.1 Understanding and Mapping Big Data in Transport Sector. [PDF] Leveraging Big Data to Manage Transport Operations (LeMO) project 2018, pp.6-7. Available at: https://lemo-h2020.eu/newsroom/2018/5/13/deliverable-11-understanding-big-data-in-transport-sector.

Heggenberger R., & Mayer, C. (2018). Mobilität 4.0 – neue Geschäftsmodelle für Produkt- und Dienstleistungsinnovationen. Predictive Analytics in der Mobilitätsbranche. Springer Fachmedien Wiesbaden.

Hu, H., Wen, Y., Chua, T. S., & Li, X. (2014). Toward scalable systems for big data analytics: A technology tutorial. IEEE access, 2, 652-687.

Huang, S., Huang, J., Dai, J., & Xie, T., Huang, B. (2010). The hibench benchmark suite. Characterization of the mapreduce-based data analysis. Workshops Proceedings of the 26th International Conference on Data Engineering, ICDE 2010, March 1-6, Long Beach, California, USA. pp. 41-51

IBM (2011). IBM SPSS Modeler CRISP-DM Guide. [PDF] IBM Corporation 1994, pp.1-3. Available at: ftp://public.dhe.ibm.com/software/analytics/spss/documentation/modeler/14.2/en/CRISP_DM.pdf.

ICT (2013). Chinese Academy of Sciences: CloudRank-D. Available at: http://prof.ict.ac.cn/CloudRank/

ICT (2013). Chinese Academy of Sciences: DCBench. Available at: http://prof.ict.ac.cn/DCBench/

ICT (2015). Chinese Academy of Sciences: BigDataBench 3.1. http://prof.ict.ac.cn/BigDataBench/

Intel (2015). HiBench Suite (2015). Available at: https://github.com/intel-hadoop/HiBench

Ivanov, T., Izberovic, S., & Korfiatis, N. (2016). The Heterogeneity Paradigm in Big Data Architectures. In Managing and Processing Big Data in Cloud Computing (pp. 218-245). IGI Global.

Ivanov, T., Izberovic, S., & Korfiatis, N. (2017). The Heterogeneity Paradigm in Big Data Architectures. In Artificial Intelligence: Concepts, Methodologies, Tools, and Applications (pp. 485-511). IGI Global.

Ivanov, T., Rabl, T., Poess, M., Queralt, A., Poelman, J., Poggi, N., & Buell, J. (2015, August). Big data benchmark compendium. In Technology Conference on Performance Evaluation and Benchmarking (pp. 135-155). Springer, Cham.

Ivanov, T., & Zicari, R. V.. (2018). Analytics Benchmarks. To appear in the Encyclopedia of Big Data Technologies, 2018.

Jeske, M., Grüner, M., & Weiß, F. (2013). Big Data in Logistics [PDF]. Troisdorf, Germany: DHL Customer Solutions & Innovation.

Ji, Z., Ganchev, I., O'Droma, M., Zhao, L., & Zhang, X. (2014). A cloud-based car parking middleware for IoT-based smart cities: Design and implementation. Sensors, 14(12), 22372-22393.

Kamel, I. R., Abdelgawad, H., & Abdulhai, B. (2016). Transportation big data simulation platform for the Greater Toronto Area (GTA). In Smart City 360° (pp. 443-454). Springer, Cham.

Karloff H.; Suri S.; Vassilvitskii S. (2010) "Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms", pp. 938-948

Kephart, J.O., Chess, D.M.: The vision of autonomic computing. Computer 36(1), 41–50 (2003)

Khan S., Rahman M., Apon A., Chowdhury M. (2017) Characteristics of Intelligent Transportation Systems and Its Relationship with Data Analytics in Data Analytics for Intelligent Transportation Systems, Elsevier, pp. 1-29,

Khazaei, H., Zareian, S., Veleda, R., & Litoiu, M. (2016). Sipresk: A big data analytic platform for smart transportation. In Smart City 360° (pp. 419-430). Springer, Cham.

Kim, K., Jeon, K., Han, H., Kim, S.G., Jung, H., & Yeom, H. (2008). Mrbench. A benchmark for mapreduce framework. 14th International Conference on Parallel and Distributed Systems. ICPADS 2008. Melbourne. Victoria. Australia. December 8-10. pp. 11-18

Krajzewicz, D., Erdmann, J., Behrisch, M., & Bieker, L. (2012). Recent development and applications of SUMO-Simulation of Urban MObility. International Journal On Advances in Systems and Measurements, 5(3&4).

Kunjir, M., Kalmegh, P., & Babu, S. (2014). Thoth. Towards managing a multi-system cluster. PVLDB 7(13). 1689-1692

Li, M., Tan, J., Wang, Y., Zhang, L., & Salapura, V. (2015). Sparkbench. A comprehensive benchmarking suite for in memory data analytic platform spark. Proceedings of the 12th ACM International Conference on Computing Frontiers. CF '15. ACM. New York. USA. pp. 53:1-53:8.

Lipic, T., Skala, K., & Afgan, E. (2014, January). Deciphering big data stacks: An overview of big data tools. In Fifth International Workshop on Big Data Analytics: Challenges, and Opportunities (BDAC-14).

Lu, R., Wu, G., Xie, B., & Hu, J. (2014, December). Stream bench: Towards benchmarking modern distributed stream computing frameworks. In Utility and Cloud Computing (UCC), 2014 IEEE/ACM 7th International Conference on (pp. 69-78). IEEE.

Luckow A., Kennedy K., (2017) Data Infrastructure for Intelligent Transportation Systems in Data Analytics for Intelligent Transportation Systems. Elsevier. pp. 113-129.

Lukácová, A., Babi c, F., & Parali c, J. (2014). Building the prediction model from the aviation incident data. In Applied Machine Intelligence and Informatics (SAMI),2014 I.E. Herl'any, Slovakia: 12th International Symposium on IEEE.

Luo, C., Zhan, J., Jia, Z., Wang, L., Lu, G., Zhang, L., Xu, C., & Sun, N. (2012). Cloudrankd. benchmarking and ranking cloud computing systems for data processing applications. Frontiers of Computer Science 6(4), 347-362

MacQueen, J. (1967, June). Some methods for classification and analysis of multivariate observations. In Proceedings of the fifth Berkeley symposium on mathematical statistics and probability (Vol. 1, No. 14, pp. 281-297).

Manyika, J., Chui, M., Brown, B., Bughin, J., Dobbs, R., Roxburgh, C., & Byers, A. (2011). Big data: The Next Frontier for Innovation, Competition, and Productivity. Tech. rep., McKinsey Global Institute

Matignon, R. (2007). Data mining using SAS enterprise miner (Vol. 638). John Wiley & Sons.

McAfee, A., Brynjolfsson, E., Davenport, T. H., Patil, D. J., & Barton, D. (2012). Big data: the management revolution. Harvard business review, 90(10), 60-68.

Meng, X., Bradley, J., Yavuz, B., Sparks, E., Venkataraman, S., Liu, D., ... & Xin, D. (2016). Mllib: Machine learning in apache spark. The Journal of Machine Learning Research, 17(1), 1235-1241.

Mian, R., Ghanbari, H., Zareian, S., Shtern, M., & Litoiu, M. (2014, September). A data platform for the highway traffic data. In Maintenance and Evolution of Service-Oriented and Cloud-Based Systems (MESOCA), 2014 IEEE 8th International Symposium on the (pp. 47-52). IEEE.

Min Li (2015). SparkBench. Available at: https://bitbucket.org/lm0926/sparkbench

Ming, Z., Luo, C., Gao, W., Han, R., Yang, Q., Wang, L., & Zhan, J.(2013). BDGS: A scalable big data generator suite in big data benchmarking. Advancing Big Data Benchmarks - Proceedings of the 2013Workshop Series on Big Data Benchmarking, WBDB.cn. Xi'an. China. July 16-17. 2013 and WBDB.us. San Jose. CA. USA..October 9-10. Revised Selected Papers. pp. 138-154

Mirabadi, A., & Sharifian, S. (2010). Application of association rules in Iranian railways (RAI) accident data analysis. Safety Science, 48, 1427–1435. https://doi.org/10.1016/j.ssci.2010.06.006

Mohd Selamat, S. A., Prakoonwit, S., Sahandi, R., Khan, W., & Ramachandran, M. (2018). Big data analytics—A review of data-mining models for small and medium enterprises in the transportation sector. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 8(3), e1238.

Moreno-Díaz, R., Pichler, F., & Quesada-Arencibia, A. (2015). Using data mining to improve the public transport in gran Canaria Island (Vol. 9520). Switzerland:Springer International Publishing. https://doi.org/10.1007/978-3-319-27340-2

MRBS (2013). Available at: http://sardes.inrialpes.fr/research/mrbs/index.html

Nambiar, R., Poess, M., Dey, A., Cao, P., Magdon-Ismail, T., Ren, D., & Bond,A. (2014). Introducing tpcx-hs. The rst industry standard for benchmarking big data systems. In: Performance Characterization and Benchmarking. Traditional to Big

Nayak, R., Emerson, D., Weligamage, J., & Piyatrapoomi, N. (2011, March). Road crash proneness prediction using data mining. In Proceedings of the 14th International Conference on Extending Database Technology (pp. 521-526). ACM.

Olston, C., Reed, B., Srivastava, U., Kumar, R., & Tomkins, A. Pig (2008). Latin - a not so foreign language for data processing. Proceedings of the ACM SIGMOD International Conference on Management of Data. SIGMOD 2008. Vancouver. BC. Canada. June 10-12. pp. 1099-1110

Patil, S., Polte, M., Ren, K., Tantisiriroj, W., Xiao, L., Lopez, J., Gibson, G., Fuchs, A., & Rinaldi, B.(2011) YCSB++: benchmarking and performance debugging advanced features in scalable table stores. ACM Symposium on Cloud Computing in conjunction with SOSP. SOCC '11. Cascais. Portugal. October 26-28. p. 9

Pavlo, A., Paulson, E., Rasin, A., Abadi, D., DeWitt, D., Madden, S., & Stonebraker, M. (2009) A Comparison of Approaches to Large-Scale Data Analysis. SIGMOD. pp. 165-178

Piatetsky-Shapiro, G., Smyth, P., & Uthurusamy, R. eds., (1996). Advances in knowledge discovery and data mining (Vol. 21). U. M. Fayyad, P. Smyth, & R. Uthurusamy (Eds.). Menlo Park: AAAI press.

Poggi, N., Carrera, D., Call, A., Mendoza, S., Becerra, Y., Torres, J., Ayguade, E., Gagliardi, F., Labarta, J., Reinauer, R., Vujic, N., Green, D., & Blakeley, J. (2014) ALOJA: A systematic study of hadoop deployment variables to enable automated characterization of cost-e ectiveness. IEEE Intl. Conf. on Big Data. Big Data 2014. Washington. DC. USA. October 27-30. pp. 905-913

Poonawala, H., Kolar, V., Blandin, S., Wynter, L., & Sahu, S. (2016). Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Singapore in Motion: Insights on Public Transport Service Level Through Farecard and Mobile Data Analytics. KDD '16. New York. NY. USA pp. 589-598

Qiu, J., Jha, S., Luckow, A., & Fox, G. C. (2014). Towards HPC-ABDS: an initial high-performance big data stack. Building Robust Big Data Ecosystem ISO/IEC JTC, 1, 18-21.

Rabl, T. & Poess, M. (2011) Parallel data generation for performance analysis of large, complex RDBMS. DBTest '11. p. 5

Rabl, T., Frank, M., Sergieh, H., Kosch, H. (2010). A data generator for cloud-scale benchmarking. Performance Evaluation, Measurement and Characterization of Complex Systems - Second TPC Technology Conference. TPCTC 2010. Singapore, September 13-17. Revised Selected Papers. pp. 41-56

Rockart JF, Ball L, Bullen CV (1982) Future role of the information systems executive. MIS Q 6(4):1–14

Sakr, S. & Casati, F. (2011) Liquid benchmarks: Towards an online platform for collaborative assessment of computer science research results. Proceedings of the Second TPC Technology Conference on Performance Evaluation. Measurement and Characterization of Complex Systems. TPCTC'10. Springer-Verlag. Berlin. Heidelberg pp. 10-24

Sakr, S., Shafaat, A., Bajaber, F., Barnawi, A., Batar, O., & Altalhi, A. (2015). Liquid benchmarking: A platform for democratizing the performance evaluation process. Proceedings of the 18th International Conference on Extending Database Technology. EDBT 2015. Brussels. Belgium. March 23-27. pp. 537-540

Sangroya, A., Serrano, D., & Bouchenak, S. (2012). MRBS: A Comprehensive MapReduce Benchmark Suite. Tech. rep. LIG. Grenoble. France

Sangroya, A., Serrano, D., & Bouchenak, S. (2012). MRBS: towards dependability benchmarking for hadoop mapreduce. Euro-Par 2012. Parallel Processing Workshops - BDMC. CGWS. HeteroPar. HiBB. OMHI. Paraphrase PROPER. Resilience. UCHPC. VHPC. Rhodes-Islands. Greece. August 27-31. Revised Selected Papers. pp. 3-12

Schoenherr, T., &, Speier-Pero, C. (2016). Data Science, Predictive Analytics, and Big Data in Supply Chain Management: Current State and Future Potential. Journal of Business Logistics. Volume 36. pp. 120-133

Sherif S. (2015). Liquid benchmarking. Available at: http://wiki.liquidbenchmark.net/doku.php/home

Shim JP, Warkentin M, Courtney JF, Power DJ, Sharda R, Carlsson C (2002) Past, present, and future of decision support technology. Decision Support Systems 33(2):111–126

Shin, I.-H., Park, G.-L., Saha, A., Kwak, H.-Y., & Kim, H. Analysis of moving patterns of moving objects with the proposed framework. Computational Science andIts Applications–ICCSA 2009, 2009, 5593:443–452, doi:https://doi.org/10.1 007/978-3-642-02457-3_38

Shtern, M., Mian, R., Litoiu, M., Zareian, S., Abdelgawad, H., & Tizghadam, A. (2014, September). Towards a multi-cluster analytical engine for transportation data. In Cloud and Autonomic Computing (ICCAC), 2014 International Conference on (pp. 249-257). IEEE.

Shukla, A., Chaturvedi, S., & Simmhan, Y. (2017). RIoTBench: An IoT benchmark for distributed stream processing systems. Concurrency and Computation: Practice and Experience, 29(21).

Stonebraker, M., Abadi, D, DeWitt, D., Madden, S., Paulson, E., Pavlo, A., Rasin, A.: Mapreduce and parallel dbmss: friends or foes? Commun. ACM 53(1), pp. 64-71

TPC (2018) https://www.tpc.org

Transaction Processing Performance Council. TPC Benchmark DS – Standard Specification (2015), version 1.3.1

Transaction Processing Performance Council. TPC Benchmark H - Standard Specification (2014), version 2.17.1

Transaction Processing Performance Council. TPC Express Benchmark HS - Standard Specification (2015), version 1.3.0

Venugopal, A., & Gualtieri, M. (2018, June 1). Apache Spark Empowering the Real-time, Data Driven Enterprise: The De Facto Choice for Stream Processing and Machine Learning [Webinar]. Retrieved from: https://www.streamanalytix.com/webinar/apache-spark-empowering-real-time-data-driven-enterprise-de-facto-choice-stream-processing.

Viglioni, G. M. C., Cury, M. V. Q., & Silva, P. A. L. D. (2007). Methodology for railway demand forecasting using data mining. In Proceedings of the SAS GlobalForum, Brazil.

Wang, G., Gunasekaran, A., Ngai, E., & Papadopoulos, T. (2016) Big data analytics in logistics and supply chain management: Certain investigations for research and applications. International Journal of Production Economics. Volume 176. pp. 98-110

Wang, L., Zhan, J., Luo, C., Zhu, Y., Yang, Q., He, Y., Gao, W., Jia, Z., Shi, Y., Zhang, S., Zhen, C., Lu, G., Zhan, K., Li, X., & Qiu, B. (2014) BigDataBench: a Big Data Benchmark Suite from Internet Services. HPCA

Wong, J.-Y., & Chung, P.-H. (2007). Managing valuable Taiwanese airline passengers using knowledge discovery in database techniques. Journal of Air TransportManagement, 13, 362–370. https://doi.org/10.1016/j.jairtraman.2007.07.001

Wu, L., Yuan, L., & You, J. (2015). Survey of large-scale data management systems for big data applications. Journal of computer science and technology, 30(1), 163-183.

Xiong, W., Yu, Z., Bei, Z., Zhao, J., Zhang, F., Zou, Y., Bai, X., Li, Y., & Xu, C.: A characterization of big data benchmarks. Proceedings of the 2013 IEEE International Conference on Big Data. 6-9 October. Santa Clara. CA. USA. pp. 118-125

Xiong, W., Yu, Z., Eeckhout, L., Bei, Z., Zhang, F., & Xu, C. (2016). ShenZhen transportation system (SZTS): a novel big data benchmark suite. The Journal of Supercomputing, 72(11), 4337-4364.

Yahoo (2015). YCSB, Available at: https://github.com/brianfrankcooper/YCSB

Yang, Q., & Koutsopoulos, H. N. (1996). A microscopic traffic simulator for evaluation of dynamic traffic management systems. Transportation Research Part C: Emerging Technologies, 4(3), 113-129.

Yanpei C. (20139. Statistical Workload Injector for MapReduce (SWIM). Available at: https://github.com/SWIMProjectUCB/SWIM/wiki

Zareian, S., Veleda, R., Litoiu, M., Shtern, M., Ghanbari, H., & Garg, M. (2015, June). K-Feed-a data-oriented approach to application performance management in cloud. In 2015 IEEE 8th International Conference on Cloud Computing (CLOUD) (pp. 1045-1048). IEEE.

Zeng, G. (2015). Application of Big Data in Intelligent Traffic System. IOSR Journal of Computer Engineering, 17(1), 01-04.

Zhang, C., Huang, Y., & Zong, G. (2010). Study on the application of knowledge discovery in data bases to the decision making of railway traffic safety in China. In:2010 International Conference on Manageme nt and Service Science (MASS). https://doi.org/10.1109/ICMSS.2010.5577012

Zhang, J., Wang, F. Y., Wang, K., Lin, W. H., Xu, X., & Chen, C. (2011). Data-driven intelligent transportation systems: A survey. IEEE Transactions on Intelligent Transportation Systems, 12(4), 1624-1639.

Zhu, L., Yu, F. R., Wang, Y., Ning, B., & Tang, T. (2018). Big Data Analytics in Intelligent Transportation Systems: A Survey. IEEE Transactions on Intelligent Transportation Systems.

Zicari, R. V., Rosselli, M., Ivanov, T., Korfiatis, N., Tolle, K., Niemann, R., & Reichenbach, C. (2016). Setting up a Big Data project: Challenges, opportunities, technologies and optimization. In Big Data optimization: Recent developments and challenges (pp. 17-47). Springer, Cham.