DataBench

Evidence Based Big Data Benchmarking to Improve Business Performance

D1.3 Horizontal Benchmarks – Analytics and Processing

Abstract

This document - deliverable D1.3 of the DataBench project is focusing on benchmarks in the horizontal layers according to the BDVA reference model with data visualization (visual analytics), data analytics and data processing. Visual Analytics is an area that has been less covered in existing benchmarks, but an existing starting point for this can be found in the Hobbit-IV benchmark on visualization and services, which also focuses on question answering and faceted browsing. Data analytics include a level of industrial analytics with descriptive, diagnostic, predictive and prescriptive analytics and the support for this with the use of machine learning. Machine Learning includes supervised and unsupervised learning as well as reinforcement learning and has a strong focus in many ongoing ICT14 and ICT15 projects. Analytics is addressed for graph representations in the Hobbit-II benchmark on Graphalytics, but is also a focus in benchmarks on deep learning like DeepMark and DeepBench. Different analytic benchmarks will typically address different big data types such as time series, spatial, image and text. The area of data processing architectures includes benchmarks for real time processing with stream processing, batch processing and interactive processing and main memory architectures. These are areas covered in many benchmarks such as BigBench, BigDataBench and SparkBench – benchmarking different processing architectures such as MapReduce (Hadoop), SPARK and Flink and others. In this document, the benchmarks will be classified in the following categories:

- Data Visualization (visual analytics),
- Data Analytics - (including Machine Learning and AI benchmarks)
- Data Processing

This "D1.3 Horizontal Benchmarks – Analytics and Processing" document is relating to the public version of the document "D1.2 DataBench Framework – with Vertical Big Data Type benchmarks" and the document "D1.4 Horizontal Benchmarks – Data Management" which have been provided at the same time as this document.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 780966

Deliverable D1.3	Horizontal Benchmarks – Analytics and Processing
Work package	WP1
Task	1.3
Due date	31/12/2019
Submission date	26/05/2021
Deliverable lead	JSI
Version	2.0
Dissemination level	Public
Authors	JSI (Marko Grobelnik) SINTEF (Arne Berre, Volker Hoffman, Kasia Michalowska, Bushra Nazir, Chaudhry Rehan Ikram, Muhammad Shah Zaib, Afroditi Tsalgatidou) GUF (Todor Ivanov, Timo Eichhorn) ATOS (Tomás Pariente, Iván Martínez and Ricardo Ruiz) POLIMI (Chiara Francalanci)
Reviewers	Gabriella Cattaneo Mike Glennon

Keywords

Benchmarking, big data, big data technologies, BDVA Reference Model, AI, Machine Learning, Horizontal benchmarks, architecture, performance metrics

Disclaimer

This document reflects the authors view only. The European Commission is not responsible for any use that may be made of the information this document contains.

Copyright Notice

Copyright belongs to the authors of this document. Use of any materials from this document should be referenced and is at the user's own risk.

Table of Contents

Table of Figures

Table of Tables

Executive Summary

This document addresses the three horizontal layers in the BDVA Big Data Reference architecture from the perspective of benchmarking. The three layers include data visualization & interaction, data analytics, and data processing. The three layers constitute the operational building blocks for most of the Big Data and AI solutions.

This document collects and analyses a number of relevant benchmarking frameworks and approaches found in the literature and on the web. The benchmarks are analysed across several dimensions, including the aim, workloads, metrics, frameworks and the corresponding references. Each of the collected elements will get included in the future work on the DataBench project aligned with the DataBench ontology and knowledge graph.

This document provides also the historical perspective into the benchmarking evolved in the corresponding fields and tries to identify the reasons why particular benchmarks appeared in a certain time frame, reflecting the current state of the technology.

The document touches also on the possible future evolution of benchmarking with the projected developments in BigData and AI technologies. It emphasizes an important direction where benchmarking could have impact in the future, namely the "AI Certification" solutions which would rely on the current benchmarking technology and extend towards technology benchmarking against legal frameworks.

This document refers to the public version of deliverable D1.2, which provides an introduction to the objectives of the Work Package 1 and an extensive catalog of most of the existing benchmarking initiatives and tools. All benchmarks collected in the annexes of D1.2 have been therefore referenced from this document.

1. Introduction and Objectives

The DataBench Framework is based on a combination of both the vertical and horizontal dimensions of the BDVA Reference Model, which uses a set of 6 different Big Data types (Structured data, Time series/IoT, Spatial, Media, Text and Graph) to focus on end-to-end support along the horizontal layers of visualisation, analytics, processing and data protection and management. DataBench aims to identify and unify numerous existing BDT benchmarking initiatives and their business and technical metrics into a common structure. The main objective is to investigate and deliver a single model to import and assess the technical requirements and data coming from existing benchmarking tools and platforms and provide recommended benchmarks in accordance to different dimensions of BDVA reference model and different data types. The first five layers within the horizontal dimension includes data visualisation, analytics, processing, data protection and management. Industry applications and benchmarks belonging/categorized to these layers cover various Big Data types from structured data through time series/real-time streaming. The objective is to provide a model which correlates technical benchmarks to performance and business needs of different sectors and domains.

BDVA reference model includes six horizontal layers out of which the top three (data visualization/interaction, analytics and processing) are covered in this D1.3 deliverable, while the other three (data protection, management and infrastructure) are considered in the parallel deliverable D1.4 sharing the structure and the approach of this document. The top three layers are typically considered as the operational layers constructed from technical building blocks constituting a typical Big Data system, or more broadly an Artificial Intelligence system.

The key operational steps in data science include (a) ingesting the data using some type of data infrastructure, (b) analysing the data resulting into new insights and knowledge hidden in the data, and (c) delivering the insights back to the user either as a static result or through some form of interaction. In this deliverable D1.3 we are analysing different academic and commercial endeavours how to benchmark the building blocks for all three categories.

Next, we provide a quick background and an intuitive introduction of the three areas in focus.

Data visualization and interaction is the end-user facing set of technologies which generally allow to master the complexity hidden in the data via a diverse set of techniques.

Data visualization typically deals with aggregating data and showing them (usually) in two dimensions. Using different visual elements and tricks we can visualize in two dimensions (i.e. a typical screen of paper sheet) more than two dimensions – with some approaches up-to five or six dimensions. This allows visual insights into data complexity beyond what a human can observe from tabular data representation or simple 2D visual charts.

Data interaction techniques are typically delivered through visual interfaces, most often as an interactive on-screen technology, but in special cases an interaction technology can be delivered through specialized interaction devices (e.g. specialized glasses or similar). The main aim of interaction technologies is to allow comprehending a complex multi-dimensional nature of the observed data. This is typically done in a cycle, where a user selects a subset of dimensions in the data, investigates data through these dimensions, and at some point re-selects dimensions, until it finds the useful insights. Data interaction systems are often specific to an application, but some of the systems (evaluated below in the document) try to be generic, targeting certain types of data modalities.

Data analytics techniques are usually considered as the heart of the data management systems. We use here the term 'data analytics' as a cover term for a numerous set of techniques developed in the research areas such as statistics, machine learning, artificial intelligence, database and others. The general principle of data analytic algorithms is (a) to take data of some kind on the input, (b) analyse them with specific operators (most often based on mathematical/numerical analysis), and (c) extracting either aggregates or some kind of new knowledge or insights with (hopefully) some value for the end user. This paradigm in the before-mentioned three steps fits the vast majority of the approaches which we consider today as data analytics. There are exceptions, like 'reinforcement learning' techniques, where the setting is different and the machine extracts insights by extensive simulation whereas a human provides just top-level guidelines.

Data processing and architectures are conceptual and technical infrastructures used to deal with various application requirements. The most traditional architecture and approach is 'batch processing', which is simple and doesn't require any specific infrastructure, but also doesn't satisfy most of the intensive modern applications. Significant changes in the data processing architectures were introduced by optimizing the scale and the speed of data processing, in particular optimizing the low-latency (i.e., fast response). To achieve such application requirements, the data processing architectures had to be adapted in way to deal with many computers and to do processing in the stream. For the scenarios with low latency, 'in-memory' solutions were introduced to avoid accessing slow hard-drives. Nowadays, most of the advanced data processing architectures are available through open-source tools and are widely used in data intensive applications.

The D1.3 document is structured as follows:

- Section 1 provides the introduction to the objectives of WP1 and the deliverable.
- Section 2 describes the DataBench Framework – based on the partners contributions to the BDVA Big Data Reference architecture – and the extensions for various big data types and thus usage of these within different application scenarios. Existing and new benchmarking approaches and challenges are being continuously mapped into the DataBench Framework matrix showing the relationship to the focus aspects of these.
- Section 3 presents different Benchmarking approaches, including types of technical benchmarks and the relationship to business benchmarks. This is followed by a description of different benchmarking organizations, like TPC, SPEC, STAC, LDBC, BenchCouncil, BDVA-TF6-SG7 and Hobbit. Further the section presents application benchmarks and big data types, use cases and application domains, Big Data standards (ISO SC42), Challenges and inducement prices for big data application problems.
- Section 4 provides the conclusions and the discussion of the benchmarking technologies as it was appearing over last decades. We also project possible future evolution of the benchmarking technology and how it might connect to the current popularity and issues related to the field of Artificial Intelligence.

2. The DataBench Framework

2.1 Overview of the DataBench Framework and analytics

The DataBench Framework is further described in the DataBench D1.2 document [73].

This document - deliverable D1.3 - is focusing on benchmarks in the horizontal layers according to the BDVA reference model as follows:

- Data Visualization (visual analytics)
- Data Analytics – (including Machine Learning and AI benchmarks)
- Data Processing

However, we only focus on the most appropriate and relevant benchmarks that satisfy a set of criteria. First, they need to be publicly available in the form of source code or/and execution binaries. Second, they should be regularly updated in terms of bug fixing, usability improvements and new functional extensions. Third, there should be available user documentation, installation and usage guides that accurately describe how to apply the benchmark. Finally, the benchmark should be popular among users (i.e., being actively used and referenced in the relevant literature) in terms of reported results, vendor comparisons and scientific papers, which basically suggests that the benchmark offers a good baseline for comparison and is accepted as a standardized measurement tool.

It is worth mentioning that this document does not contain extensive descriptions of the benchmarks, as most of them have already been described in detail in the annexes of deliverable D1.2. Therefore, this document provides references to those descriptions in the public version of D1.2 and complements them with extra information.

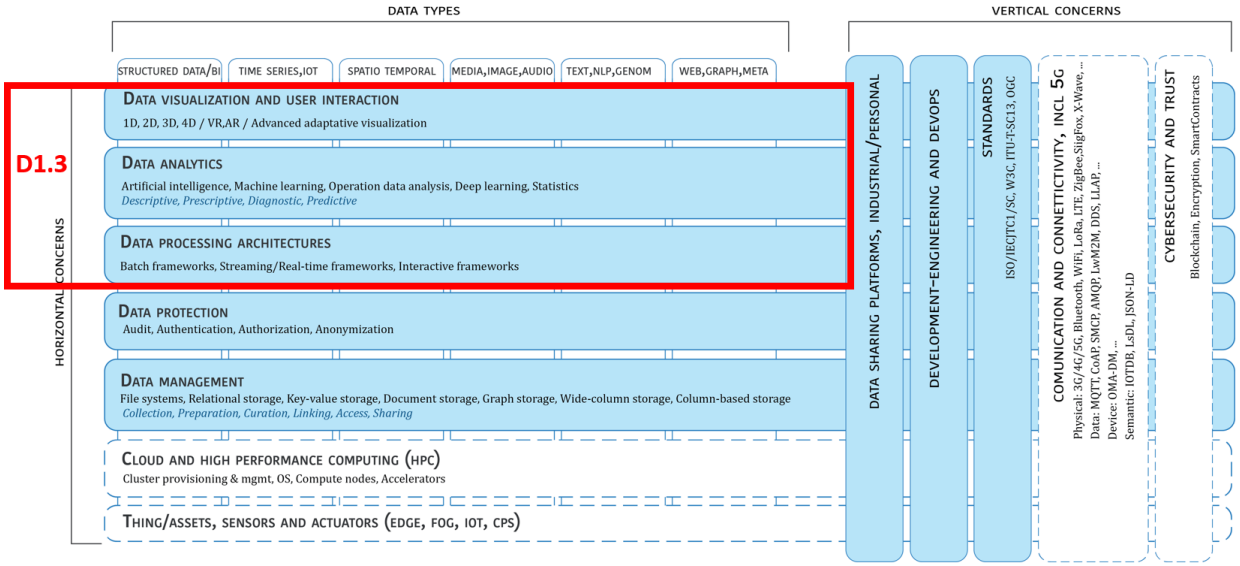

Figure 1 DataBench Framework and the layers covered in the document

3. Horizontal Benchmarks – Analytics and Processing

3.1 Data Visualization and User Interaction

Data visualization and user interaction is an area which is generally less represented among benchmarks primarily due to the 'human in the loop' which cannot be entirely automated. The area and the range of problems include (a) visualization of (typically) complex data via various techniques, (b) interaction via question-answering developed in recent years through chat-bot interfaces, and (c) faceted browsing allowing inspection and search of databases across many dimensions. Each of the listed topics has some element of creativity and is targeting specific user groups with associated skills and aims. Despite some of the traditional patterns in the design, application specifics require new elements to be invented and introduced in the operational frameworks. The variety of terminals for human-computer interaction (including mobile phones, tablets, web interfaces, desktop applications, visualization panels) add additional constraints to the design and benchmarking which is in general not easy to generalize. Indicators on how to measure effectiveness of such frameworks are usually expressing efficiency of data retrieval for a particular application class and communication terminals.

Popular frameworks for data visualization are recently centered around JavaScript libraries which offer flexibility in design and in particular deployment of solutions across devices and integration with back-end frameworks. A good overview of the currently (as for 2019) popular open-source frameworks is provided in the blog [65].

Popular frameworks for Question Answering are numerous. At the present date there are many open-source and commercial frameworks (like Amazon, Microsoft, IBM, IPSoft) offering technology to train and deploy chatbots. The technology to build a question answering system is generally well known, but since each chatbot is mostly specific in its domain, the technology to train or manually design the system, generally requires lots of resources in terms of data (i.e. previous QA interactions). There are numerous open-source frameworks [66] to build a question answering system. Other good overviews of technology and frameworks are available at [67] and [68].

Faceted-search as an interactive query-construction interface is a technology which in the most cases is implemented by the programmer. Still, there are several open source frameworks [69], as well as commercial ones: Algolia, Cludo, ExperRec, Sajari, SearchBox, Site Search 360, SwiftType.

Relevant benchmarks which we selected in the DataBench and described in more details in the follow-up Table 1 are Hobbit IV, VAST Challenges, and IDEBench.

Name	Compares	Workload type	Metrics	Frameworks	Data Types	URL
Hobbit – Benchmark IV – Visualization & Services	*Question Answering*: correctness of answers to natural language queries from knowledge bases *Faceted Browsing*: browsing scenarios performance on the different types of transitions	Multilinguality, Hybrid data sources, Scale	*Question Answering*: Precision, Recall, F-Measure *Faceted Browsing*: Query-per-second, Accuracy, Precision, Recall, F1-Score, Number of remaining instances for future facet selection.	DBPedia	Multilingual Text, knowledge bases (linked data), structured data (heterogeneous data sources)	https://project-hobbit.eu/outcomes/benchmark-iv-visualisation-services/
	Described in D1.2. Section Number 7.13 HOBBIT provides benchmarks for (a) question answering and (b) faceted browsing. The project does not benchmark user interfaces themselves but focusses on the provision of performance and accuracy measurements for approaches used in interfaces.					
VAST Challenges	A series of scenarios performed over annual competition comparing answers to the ground truth	Repository of images and data	Accuracy of a solution compared to the ground truth provided by a particular challenge		Structured data, textual data, images	https://www.cs.umd.edu/hcil/varepository/benchmarks.php
	The Visual Analytics Benchmarks Repository contains resources to improve the evaluation of visual analytics technology. Benchmarks contain datasets and tasks, as well as materials describing the uses of those benchmarks (the results of analysis, contest entries, controlled experiment materials etc.) Most benchmarks contain ground truth described in a solution provided with the benchmark, allowing accuracy metrics to be computed.					
IDEBench: A Benchmark for Interactive Data Exploration	Compares ad-hoc workloads in realistic interactive data exploration (IDE) query engines	Simulated workloads across 4 main IDE scenarios: Independent Browsing, Sequential Linking, 1:N Linking, N:1 Linking	Time Requirement Violated, Missing Bins, Mean Relative Error, Cosine Distance, Mean Margin Error, Out of Margin, Bias	MonetDB, approximated/XDB, IDEA, System X	Structured Data	https://idebench.github.io https://arxiv.org/pdf/1804.02593.pdf
	Described in D1.2. Section Number 7.15 IDEBench is evaluating the performance of database systems for interactive data exploration workloads. As opposed to traditional benchmarks for analytical database systems, the goal is to provide realistic workloads and datasets that can be used to benchmark IDE query engines, with a particular focus on metrics that capture the trade-off between query performance and quality of the result.					

Table 1 Summary of Data Visualization and Usr Interaction benchmarks

Visual analytics benchmarks are relatively diverse in their nature, due to targeting user-facing scenarios. The challenges across different benchmarks concern typical scenarios when visualizing data and interactive data exploration, both present in many applications. The types of data considered in benchmarking scenarios range from structured data, text, knowledge bases/linked data,to images.

The key critique, after summarizing the analyzed benchmarks, is that the authors of the benchmarks and corresponding challenges are struggling how to define a task and which elements of the problem area to target. Typically, only the well-measurable aspects (possible to automate) are targeted and consequently benchmarked.

3.2 Data Analytics & Machine Learning

The core of Big Data technologies and corresponding areas of Data Science, Machine Learning, Data Mining and broader Artificial Intelligence is the data analytics segment. The data analytics typically concentrates on a few key scenarios which propagate its results across technology stacks.

These are the two key scenarios in the data analytics, which in various adaptations shape all other scenarios and methodologies:

(a) ***Supervised analytics***, where the goal is to construct a mapping (typically in a form of an analytic model) between a data (independent variables) and a targeted value (dependent variable). The resulting model is capable of generating values for the targeted value (dependent variable) from yet an unseen data (independent variables). This category of approaches include classification and regression algorithms.

(b) **Unsupervised analytics**, where the goal is to extract the structure in the data. The resulting structure is most typically expressed as a list of (sometimes interconnected) segments (also called clusters or eigen-vectors) in the data. This category of approaches include clustering, eigen-vector-decomposition, and other algorithms without using a specific guidance (through dependent variables), but having some form of metric, to determine proximity of data elements.

Note, that other scenarios in data analytics (across many areas of data related sciences) are almost always a variant of the above two approaches. It is important to emphasize that in its core the methods boil-down to either finding a mapping between different aspects of a, or discovering a structure in a data. The concrete algorithms have always certain specifics in a form of diverse optimization criteria, specific parameters, targeting particular data modalities etc. Algorithms have their implementations integrated in larger frameworks used by the end-users.

Due to their importance, the data analytics algorithms, as the core to the data science related fields, have many benchmarks over the last 30 years (after 1990). The most notable in the early years of Machine Learning was UCI Machine Learning Repository (https://archive.ics.uci.edu/) which served as the key resource for the early years of Machine Learning and Data Mining. After the year 2000, the field of data analytics expanded significantly and many more alternatives appeared and the UCI repository was not capable serving the diversity of required scenarios. The reasons were mainly due to importance of particular challenges (e.g. associated prizes and PR) and ownership of data. In the recent

years (after 2010) there are many platforms where data analytics solutions can compete. Maybe as one of the more visible and influential is the commercial platform Kaggle (https://www.kaggle.com/) with many participants and easy-to-deploy challenge for (typically) commercial scenarios.

What is popular among the tools for data analytics (especially in the AI and Machine Learning part of it) was changing during the years. In 1990s there were only few publicly available tools which most of the researchers (commercial tools were not well developed yet) were using. This included systems implementing particular algorithms like C4.5, SVMLight, CART and some others. After the year 2000, the number of tools (research and commercial) increased significantly and was difficult to attribute top popularity to particular ones. After the year 2012, after the 'deep learning' revolution, the market of the tools consolidated and nowadays (year 2020) there are only few tools for deep learning which stayed in the competition. This happened mainly due to large investments from the big software companies (like Google, Facebook, Microsoft and Amazon) where smaller players cannot compete anymore. It is interesting to observe, the most powerful deep learning tools are all available for free in their full option.

In the more detailed analysis below (Table 2), we will describe only selected deliverables which are in our opinion the most representative and influential for the area. Since the field of data analytics is very active, one can expect new benchmarks and schemas will appear in the future.

Name	Compares	Workload type	Metrics	Frameworks	Data Types	URL
HiBench	Compares the deployed solution to the rest of the systems in the library of solutions across variety of metrics (including scalability and performance)	Micro Benchmarks, Machine Learning, SQL, Websearch Benchmark, Graph Benchmark, Streaming	Metrics are dependent on the type of the workload and correspond to the traditional evaluation metrics used for a particular class of the workload. The emphasis are on the clusters of indicators for usability, scalability and performance.	Hadoop: Apache Hadoop 2.x, CDH5, HDP Spark: Spark 1.6.x, Spark 2.0.x, Spark 2.1.x, Spark 2.2.x Flink: 1.0.3 Storm: 1.0.1 Gearpump: 0.8.1 Kafka: 0.8.2.2	All major data modalities including structured data, text, network data, time series. In general HiBench is data modality neutral.	https://github.com/Intel-bigdata/HiBench http://www.odbms.org/wp-content/uploads/2014/07/hibench-wbdb2012-updated.pdf
	Described in D1.2. Section Number 7.6 HiBench is a big data benchmark suite that helps evaluate different big data frameworks in terms of speed, throughput and system resource utilizations. It contains a set of Hadoop, Spark and streaming workloads, including Sort, WordCount, TeraSort, Sleep, SQL, PageRank, Nutch indexing, Bayes, Kmeans, NWeight and enhanced DFSIO, etc. It also contains several streaming workloads for Spark Streaming, Flink, Storm and Gearpump.					
BigBench	DBMS and MapReduce systems under different workloads	Simulated workloads: Prices of the retailer's competitors, Website logs, Product reviews	Traditional Big Data metric dimensions: Variety (three types of data modalities), Velocity (continuous feed into data store), Volume	Teradata Aster DBMS	Structured, unstructured and semi-structured data	https://dl.acm.org/citation.cfm?id=2463712

Name	Compares	Workload type	Metrics	Frameworks	Data Types	URL
	Described in D1.2. Section Number 7.10 BigBench 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, consisting of 30 queries.					
DeepBench	Comparing Hardware configurations across standard atomic deep learning operators	Different Deep Learning frameworks and libraries across multiple basic operations: Dense Matrix Multiplication, Convolutions, Recurrent Layers, and GPU network topologies	Time in the form of milliseconds, TeraFLOPS for particular Processor (CPU/GPU)	The benchmark itself is evaluating frameworks	The data is generic including images, feature maps, rows and columns	https://svail.github.io/DeepBench/
	Described in D1.2. Section Number 7.12 DeepBench is an open source benchmarking tool that measures the performance of basic operations involved in training deep neural networks. These operations are executed on different hardware platforms using neural network libraries.					
DeepMark	Compares speed and performance among different deep learning frameworks	Workloads are per data modalities where each modality has one or several experiments. An additional workload dimension is per hardware topology.	Specific indicators related to neural networks learning: Round-trip time for 1 epoch of training, Maximum batch-size that fits	Caffe, Chainer, MXNet, Neon, Theano, TensorFlow, Torch	Images, Video, Audio, Text	https://github.com/soumith/convnet-benchmarks/issues/101#
	Described in D1.2. Section Number 7.12 DeepMark is a benchmark comparing popular deep learning frameworks on typical deep learning problems under various hardware configurations. The emphasis is on convolutional networks.					
Fathom	Compares similarity across eight typical deep learning problems (workloads)	Standard Depp Learning Scenarios: Seq2Seq, MemNet, Speech, Autoenc, Residual, VGG, AlexNet, DeepQ	Execution time is the main indicator	TensorFlow	Images, text, dynamic control (Atari Games simulator)	https://github.com/rdadolf/fathom https://arxiv.org/abs/1608.06581
	Described in D1.2. Section Number 7.13 Fathom is a collection of eight typical deep learning problems/workloads. The key aim is to investigate how certain deep learning problems relate to each-other under different conditions of execution.					

Name	Compares	Workload type	Metrics	Frameworks	Data Types	URL
MLPerf	Compares diverse machine learning algorithms under different conditions at the training and inference stages.	MLPerf Training: Image Classification, Object Detection, Translation, Recommendation, Reinforcement Learning MLPerf Inference: Single Stream, Multiple Stream, Server, Offline	MLPerf Training: time required to train a model on the specified dataset to achieve the specified quality target MLPerf Inference: Duration, Samples/query, Latency, specific metrics per workload	No particular framework since platforms themselves are being evaluated	Image and Text data	https://mlperf.org/ https://github.com/mlperf/training https://arxiv.org/pdf/1910.01500.pdf
	Described in D1.2. Section Number 7.15 MLPerf's goal is to build fair and useful benchmarks for measuring training and inference performance of ML hardware, software, and services. It is split into two parts: (1) MLPerf Training is a benchmark suite for measuring how fast systems can train models to a target quality metric, and (2) MLPerf Inference is a benchmark suite for measuring how fast systems can process inputs and produce results using a trained model.					
MLBench	Compares top winning code from Kaggle competition with machine learning solutions in cloud (Microsoft Azure and Amazon AWS)	7 binary classification datasets, 5 multi-class classification datasets, and 5 regression datasets.	The key indicators are: performance (speed) of training the model and quality (precision) of the model	Machine Learning cloud implementations on Microsoft-Azure and Amazon AWS	Multiple data modalities including structured data, text, images	https://arxiv.org/pdf/1707.09562.pdf
	Described in D1.2. Section Number 7.15 He key aim of MLBench is to compare top solutions for a number of Kaggle competitions with corresponding solutions available from the major cloud providers (Microsoft and Amazon). It uses a novel metric based on the notion of "quality tolerance" that measures the performance gap between a given machine learning system and top-ranked Kaggle performers. Currently are available 7 binary classification datasets, 5 multi-class classification datasets and 5 regression datasets.					
OpenML Benchmark Suites	Comparing performance of various machine learning algorithms across many datasets	72 data sets carefully selected from thousands of datasets to be representative across many criteria	Traditional machine learnig metric (like Area Under Curve, Mean Squared Error)	REST API, Java Api Connector, Weka, Phyton, R	Structured data, images, text, network data – all selected to test various	https://docs.openml.org/
	Described in D1.2. Section Number 7.10 The suite offers (a) ease of use through standardized data formats, APIs, and existing client libraries; (b) machine-readable meta-information regarding the contents of the suite; and (c) online sharing of results, enabling large scale comparisons. The OpenML-CC18 is a machine learning benchmark suite of 72 classification datasets carefully curated from the thousands of datasets available on OpenML.org.					

Name	Compares	Workload type	Metrics	Frameworks	Data Types	URL
AdBench	Comparing end-to-end solutions along the pipelines with an emphasis to ad serving tasks	Streaming Analytics on Ad-serving logs, streaming ingestion and updates of various data entities, batch-oriented analytics (e.g. for Billing), Ad-Hoc analytical queries, and Machine learning for Ad targeting. Workload characteristics are found in many verticals, such as Internet of Things (IoT), financial services, retail, and healthcare.	Different Metrics across different stages (Number of events processed per second, Time needed for batch computation & Ad-Hoc Queries, Query Concurrency) Combined Metrics (Latency between Event Generation to Event Processing) Cost to meet SLAs Operational Complexity	No particular framework, but focused on particular architectures (like Lambda, Kappa or Butterfly architectures)	Transactions of structured data	http://www.tpc.org/tpctc/tpctc2016/presentations_2016/session%20009-adbench.pdf
	Described in D1.2. Section Number 7.13 It combines Ad-Serving, Streaming Analytics on Ad-serving logs, streaming ingestion and updates of various data entities, batch-oriented analytics (e.g. for Billing), Ad-Hoc analytical queries, and Machine learning for Ad targeting. While this benchmark is specific to modern Web or Mobile advertising companies and exchanges, the workload characteristics are found in many verticals, such as Internet of Things (IoT), financial services, retail, and healthcare.					
RIoTBench	Compares IoT operators across micro-benchmarks and applications	27 micro-benchmarks (common IoT tasks like data pre-processing, statistical summarization, predictive analytics) 4 real-world stream workloads (in the domain of smart cities and fitness)	Performance metrics: CPU, memory, disk, and network I/O	Storm, Microsoft-Azure	Sensor streams (numeric time series)	https://github.com/dream-lab/riot-bench https://arxiv.org/abs/1701.08530
	Described in D1.2. Section Number 7.14 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.					

Name	Compares	Workload type	Metrics	Frameworks	Data Types	URL
AI-Matrix	Compares deep learning software frameworks and hardware platforms	Micro benchmarks (GEMM as fundamentals of neural networks) Layer-based (basic elements of neural networks) Macro benchmarks (models from different application areas) Syntetic benchmarks	Elapsed time, in the future energy consumption and hardware utilization	TensorFlow, Caffee (deep learning frameworks)	Image, speech, text, structured data	https://aimatrix.ai/
	AI Matrix is a benchmark suite for testing AI software frameworks and hardware platforms. It aims at providing users a means of measuring the performance of different AI software and hardware and comparing their pros and cons. It also helps users gain insights into various factors that affect AI hardware performance and improve hardware design.					
NNBench-X	Compares individual elements and configurations within neural networks		Accuracy (quality of models), efficiency (time) by analyzing neural network operators	TensorFlow	Images, text	https://ieeexplore.ieee.org/document/8637006
	The NNBench-X approach takes as input an application candidate pool and conducts an operator-level and application-level analysis to understand the performance characteristics of both basic tensor primitives (fundamental tensor operators) and whole applications. It conducts a case study on the TensorFlow 'model zoo' (library of models) by using this proposed characterization method.					

Table 2 Summary of Data Analytics and Machine Learning benchmarks

Data Analytics, as the core to the data science tasks, is being evaluated across many benchmarking initiatives and experiments. Since there are many dimensions to be evaluated, benchmarks are typically fixing some aspects (e.g. datasets, tools, data modality) and leaving some others to be tested. In the complexity of data analytics it is not easy to test every possible aspect of the final application setup (which would be the holy grail of data analytics benchmarking), but through diverse benchmarking initiative we can just estimate and approach to the right selection of tools, hardware, data etc. for the final application.

The main critique of the data analytic field is lack of standardization of the models produced by machine learning or other data analytics tools and algorithms. The models are the core of any data analytic procedure with two sides of its use: model construction and inference. A standardized way of formulating and reusing the models would simplify many of the issues related to the benchmarking and could easily connect independent efforts to test various aspects of the field. At the present stage, the models have standardized forms only within particular tools or frameworks (like commercial products or e.g. popular Python libraries) forcing developers and data scientist to compare results within localized settings.

3.3 Data Processing Architectures

This section consists from four parts: the historical note, classification of data processing architectures, popular frameworks, and the table 3 with relevant benchmarks.

3.3.1. Background & Historical note

Data processing architectures were developed through the last decades (after late 1980s). Initially, there was no particular need for specialized architectures and most of the solutions were using batch processing. The reason was low demand for intensive data processing and analytic solutions and relatively simple deployments which required model construction in the offline settings from the given datasets, and as a separate activity inference with the pre-constructed models in applications. The power of the available hardware (in terms of CPU speed, memory sizes and disk sizes), availability of data, and network connectivity were relatively low, which prevented discussions of alternative data processing architectures. With the increase on all the previously mentioned elements (hardware, data, connectivity) and in particular with the increases requirements for more intensive data applications (in early 2000s), there also appeared an increased need for alternative data processing architectures and corresponding algorithms to support them. In particular, stream processing applications and increased amounts of data caused developments to support more than just traditional batch processing.

The first line of globally popular data intensive applications associated with interactivity were web search engines (late 1990s and early 2000s), web advertising and social networking platforms (mid 2000s). After 2010 the requirements even increased with the appearance of IoT (intensive data streams), video streaming and recently with the wide-spread usage of deep learning software. The latter moved data architectures from batch processing towards streaming (emphasizing data processing on the fly), in-memory architectures (avoiding touching offline storage, due to too slow disks), and lately GPU and cloud architectures allowing massive parallel processing (either localized or across the network).

3.3.2 Classification of data processing architectures

Therefore, the structure the existing data processing architectures and approaches could be classified among the following main patterns of usage:

- Batch processing – traditional approach to process data, where processing happens offline and results (e.g. in a form of a model or some other kind of aggregates) are used within the application. To operate with a fresh models, the batch procedure needs to be repeated and new models deployed to the application. Batch processing is a simple model and doesn't require much architectural overheads and is popular among data science researchers and developers.
- Stream processing – with the appearance of streaming data and requirements to have immediate feedback, the algorithms had to be adapted to process data 'on the fly'. This typically meant either (a) fast iterative processing on a window of data (mimicking batch processing on small portion of the most recent data) or (b) using specialized streaming algorithms which maintained a model (or other types of aggregates) in memory with fast algorithmic updating (i.e., each new data coming from the stream updated the model or aggregates in a small amount of time, faster than the rate of the incoming data). Versions of streaming solutions across longer pipelines required more than just a change

in the algorithms, but rather change in the architecture where each stage of a pipeline was processed by a different CPU or node in the network – solutions to support such pipelines were MapReduce, Spark, Kafka and others (see below).

- In-memory processing – Low-latency feedback required complete elimination of accessing a disk storage (due to speed). Low prices and availability of large amounts of computer memory allowed development of completely in-memory architectures. This simplified algorithms, decreased a need to parallelize parts of the solutions, but to some degree increased the costs of the infrastructure. Nowadays, many of the existing solutions are entirely in-memory and are using disk space just as a safety backup.
- Interactive/Real-time – Class of applications where either a human user or dependent machine require instant low level latency caused development of interactive and real-time systems. In particular, web applications and mission critical systems require instant feed-back. Solutions to such requirements include variety of approaches including edge-computing, elastic cloud and similar.

3.3.3 Popular frameworks and technologies

The above mentioned data processing architectures evolved into a series of software frameworks being used nowdays in most of the applications. Many of the popular ones are open source and freely available, while some of the bigger companies have their own frameworks, often developed on the top of the open source.

In the following, we list and summarize some of the key features for the most popular frameworks which are still evolving and of course, some new ones might appear in the future. The following summary is prepared as a compilation from [70], [71] and [72]:

- **Apache Hadoop** is a processing framework that exclusively provides batch processing. Hadoop was the first big data framework to gain significant traction in the open-source community. Based on several research papers and presentations by Google about how they were dealing with tremendous amounts of data at the time, Hadoop reimplemented the algorithms and component stack to make large scale batch processing more accessible. Apache Hadoop and its MapReduce processing engine offer a well-tested batch processing model that is best suited for handling very large data sets where time is not a significant factor. The low cost of components necessary for a well-functioning Hadoop cluster makes this processing inexpensive and effective for many use cases. Compatibility and integration with other frameworks and engines mean that Hadoop can often serve as the foundation for multiple processing workloads using diverse technology. Modern versions of Hadoop are composed of several components or layers, that work together to process batch data:
    - HDFS is the distributed filesystem layer that coordinates storage and replication across the cluster nodes. HDFS ensures that data remains available in spite of inevitable host failures. It is used as the source of data, to store intermediate processing results, and to persist the final calculated results.
    - YARN, which stands for Yet Another Resource Negotiator, is the cluster coordinating component of the Hadoop stack. It is responsible for coordinating and managing the underlying resources and scheduling jobs to be run. YARN makes it possible to run much more diverse workloads on a

Hadoop cluster than was possible in earlier iterations by acting as an interface to the cluster resources.

- **Apache Spark** is an open-source distributed general-purpose cluster computing framework. Spark's in-memory data processing engine conducts analytics, ETL, machine learning and graph processing on data in motion or at rest. It offers high-level APIs for the programming languages: Python, Java, Scala, R, and SQL. The Apache Spark Architecture is founded on Resilient Distributed Datasets (RDDs). These are distributed immutable tables of data, which are split up and allocated to workers. The worker executors implement the data. The RDD is immutable, so the worker nodes cannot make alterations; they process information and output results.
- **Apache Storm** has very low latency and is suitable for near real time processing workloads. It processes large quantities of data and provides results with lower latency than most other solutions. The Apache Storm Architecture is founded on spouts and bolts. Spouts are origins of information and transfer information to one or more bolts. This information is linked to other bolts, and the entire topology forms a DAG. Developers define how the spouts and bolts are connected.
- **Apache Samza** uses a publish/subscribe task, which observes the data stream, processes messages, and outputs its findings to another stream. Samza can divide a stream into multiple partitions and spawn a replica of the task for every partition. Apache Samza uses the Apache Kafka messaging system, architecture, and guarantees, to offer buffering, fault tolerance, and state storage. Samza relies on YARN for resource negotiation. However, a Hadoop cluster is needed. Samza has a callback-based process message API. It works with YARN to provide fault tolerance, and migrates your tasks to another machine if a machine in the cluster fails. Samza processes messages in the order they were written and ensures that no message is lost. It is also scalable as it is partitioned and distributed at all levels.
- **Apache Flink** is based on the concept of streams and transformations. Data comes into the system via a source and leaves via a sink. To produce a Flink job Apache Maven is used. Maven has a skeleton project where the packing requirements and dependencies are ready, so the developer can add custom code. Apache Flink is a stream processing framework that also handles batch tasks. Flink approaches batches as data streams with finite boundaries.
- **Amazon Kinesis Streams** is a durable and scalable real time service. It can collect gigabytes of data per seconds from hundreds of thousands of sources, including database event streams, website clickstreams, financial transactions, IT logs, social media feeds, and location-tracking events. The data captured is provided in milliseconds for real time analytics use cases, including real time anomaly detection, real time dashboards, and dynamic pricing.
- **Apache Apex** offers a platform for batch and stream processing using Hadoop's data-in-motion architecture by YARN. The platform provides integration with different data platforms. Apex also provides a framework that is easy to use. Operationally, Apex utilizes native HDFS for persisting state and the YARN features found in Hadoop such as scheduling, resource management, jobs, security, multi-tenancy, and fault-tolerance. Functionally, developers can integrate Apex APIs with other data processing systems. Apex allows for high throughput, low latency, reliability, and unified architecture, for batch and streaming use cases. It can process unbound data sets, which can grow infinitely.

- **Apache Flume** is a reliable, distributed service for aggregating, collecting and moving massive amounts of log data. It has a flexible and basic architecture. It is fault-tolerant and hardy with failover and recovery features and tunable reliability. It operates an extensible data model that The key concept behind the design of Flume is to capture streaming data from web servers to Hadoop Distributed File System (HDFS).
- **Apache Kafka** is a distributed data store optimized for ingesting and processing streaming data in real-time. Streaming data is data that is continuously generated by thousands of data sources, which typically send the data records in simultaneously. A streaming platform needs to handle this constant influx of data, and process the data sequentially and incrementally. Kafka provides three main functions to its users: (a) Publish and subscribe to streams of records, (b) Effectively store streams of records in the order in which records were generated, (c) Process streams of records in real time. Kafka is primarily used to build real-time streaming data pipelines and applications that adapt to the data streams. It combines messaging, storage, and stream processing to allow storage and analysis of both historical and real-time data.

Name	Compares	Workload type	Metrics	Frameworks	Data Types	URL
SparkBench	Compares quantitative performance using Spark under diverse hardware and software configurations	Machine Learning, Graph Computation, SQL Streaming, and Other Analytic algorithms Workloads	Performance in terms of speed and memory consumption	Apache Spark	Structured data, graphs, text, streaming data	https://bitbucket.org/lm0926/sparkbench/src/master/
	Described in D1.2. Section Number 7.11 SparkBench, developed by IBM, is a comprehensive Spark specific benchmark suite developed for in-memory data analysis to provide insights into Spark system design and performance optimization and cluster provisioning. The benchmark provides automatic generation of data sets with various scale factors. There are four main workload categories: machine learning, graph processing, streaming and SQL queries.					
Graphalytics	Comparing implementations of graph processing systems on standard graph analysis problems	Breadth-First Search, Community Detection, Local Clustering Coefficient, PageRank, Single-Source Shortest Path, Weakly Connected Components	Edges And Vertices Per Second, Load Time, MakeSpan, Processing Time, Price Per Performance	Giraph, GraphX, PowerGraph, OpenG, GraphMat, nvGRAPH, Gelly, GraphBLAS, GraphLab, Gunrock	Graphs/Networks of various sizes and characteristics	https://graphalytics.org
	Described in D1.2. Section Number 7.11 Graphalytics is an industrial-grade benchmark that enables the objective comparison of graph analysis platforms. It consists of six core algorithms (BFS, CDLP, SSSP, PR, LCC, WCC), standard datasets, synthetic dataset generators, and reference output. The design of the benchmark takes into account that graph processing and is impeded by three dimensions of diversity: platform, algorithms and datasets.					

Table 3 Summary of Data Processing benchmarks

Data processing frameworks were developed after 2000 based on the requirements for the high performance solutions for a spectrum of applications. There were two main transitions:

(a) from a traditional single machine batch processing towards parallelized batch processing (Hadoop and variants), and later

(b) from batch processing to streaming frameworks emphasizing different aspects, in particular low-latency feed-back.

We cannot identify any particular criticism for the existing set of solutions since the academic and commercial market is responding fast to the requirements from the market. A positive element, discussing the data processing frameworks, is the fact that all major platforms are open source, well documented and relatively simple to use. This caused an important up-take in the Big-Data and more recently AI solutions requiring large amounts of data to be processed.

In the future we can expect similar development of such frameworks going along with increasing hardware and connectivity (e.g. 5G) performance. An important step in the future will be inclusion of quantum computers into the ecosystem of data processing.

4. Concluding remarks & Discussion

This deliverable D1.3 presented the first three layers of the BDVA Reference Model along with the structure and the key properties of the corresponding benchmarks for each of the categories (data visualization/interaction, data analytics, data processing). The collected information will serve as a basis for the DataBench Platform and for extending the DataBench ontology and the corresponding knowledge graph.

After analyzing the selected set of benchmarks, it is possible to conclude that each of the three horizontal layers has its own properties and follows the technology developments in each of the corresponding fields.

There is a significant lack of structured benchmarks for the visualization and interaction technologies. The main reason is that such benchmarks or evaluation strategies typically include a human in the loop and many of the technical solutions are subjective, which prevents consistent and relatively inexpensive measurement. While this would be important and relevant in order to carry out a proper evaluation, in many ways the visualization and interaction still remain in the domain of designers who add an artistic touch to the solutions. We can also observe there is an evolution in the front-end technologies of what end-users perceive as acceptable, comprehensible and 'easy-to-use'. Namely, in the 1990es many of the visual elements which are common today (late 2010s), would have been considered as unintuitive. In the course of the years, some of the building blocks were pushed and identified as acceptable ones (either by being promoted by big and popular companies or became part of standard software frameworks). This generated a general acceptance and end-users learned how to use them. At the same time, the average network speed increased a lot and more complex visual solutions were possible. This is reflected also in the benchmarking frameworks which started evaluating more standard and generally popular front-end technologies.

On the side of data analytics benchmarking frameworks, we can follow a relatively long history from the mid 1990s since the early years of machine learning and data mining areas. Initially benchmarks were simple due to the lack of data and compatible with relatively low hardware capacity (i.e. CPU speed and memory sizes). With the evolution of algorithms and in particular the appearance of new products on the market (early 2000s) there was a need to evolve more standardized benchmarking frameworks. With the appearance of deep learning (after 2010) and successes in solving hard AI problems, the area of benchmarking transitioned towards evaluating very specific frameworks, used by deep learning algorithms, mainly running on GPUs, which were previously not in use. Among the benchmarks popular today in data analytics, we can see how some of the remaining few most popular deep learning frameworks (TensorFlow, PyTorch, CNTK, AlexaNet) are running on a rather specialized hardware setups and how many of the parameters (which are often not intuitive in terms of consequences) are influencing the quality and speed of getting an appropriate solution. In the meantime, many of the past benchmarking technologies fell out of use, since the majority of the data analytics area converged into using deep learning technologies.

Data processing technologies serve as an underlying infrastructural layer and evolved significantly especially in the time when the area of Big Data got popular (around 2010). In that time, mainly due to the big volumes of data, the required speed and low latency solutions caused the development of a new series of tools which are capable of operating on a cloud or on distributed architectures. Since this opened several new dimensions to be

tested (including the price of hardware), the corresponding benchmarks appeared where the most popular open-source and commercial tools were to be tested. The complexity of such benchmarks is often relatively high, since they evaluate end-to-end solutions. In many cases, these benchmarks fix certain elements (like big data processing software) while they leave open other parameters to be tested (e.g., hardware configuration or data properties).

In the future, we can expect a further evolution of benchmarking technology, especially with the development of AI as a field moving towards new types of problems like structural problems (causality, knowledge graphs, common sense reasoning and similar), as well as new issues concerning for example the ethical aspects of AI technology.

We predict the increased importance of benchmarking technology within the application of 'AI Certification', where the goal will be to evaluate the (AI) technology with the purpose of establishing trust in the performance and results of particular systems. We expect also benchmarking to get extended towards testing a technology against legal frameworks, ethics etc., which is not the case today. Examples of such initiatives are Council of Europe and OECD which are formulating legal and technical frameworks for 'AI Certification'.

5. References

[1] BDVA SRIA version 4. [Online]. Available: http://bdva.eu/sites/default/files/BDVA_SRIA_v4_Ed1.1.pdf (15 December 2019).

[2] "The Benchmark Handbook, Second Edition." [Online]. Available: https://jimgray.azurewebsites.net/BenchmarkHandbook/TOC.htm. [Accessed: 04-Jun-2019].

[3] P. K. Ahmed and M. Rafiq, "Integrated benchmarking: a holistic examination of select techniques for benchmarking analysis," *Benchmarking Qual. Manag. Technol.*, vol. 5, no. 3, pp. 225–242, Sep. 1998.

[4] M. J. Spendolini and M. J. Spendolini, *The benchmarking book*, vol. 4. Amacom New York, NY, 1992.

[5] G. H. Watson, *Strategic benchmarking: How to rate your company's performance against the world's best*. Wiley, 1993.

[6] C. J. McNair and K. H. Leibfried, *Benchmarking: A tool for continuous improvement*. John Wiley & sons, 1992.

[7] N. Poggi, "Microbenchmark," in *Encyclopedia of Big Data Technologies.*, S. Sakr and A. Y. Zomaya, Eds. Springer, 2019.

[8] R. Han, L. K. John, and J. Zhan, "Benchmarking big data systems: A review," *IEEE Trans. Serv. Comput.*, vol. 11, no. 3, pp. 580–597, 2017.

[9] "TPC-Homepage V5." [Online]. Available: http://www.tpc.org/. [Accessed: 04-Jun-2019].

[10] "TPC-H - Homepage." [Online]. Available: http://www.tpc.org/tpch/. [Accessed: 04-Jun-2019].

[11] "TPC-DS - Homepage." [Online]. Available: http://www.tpc.org/tpcds/. [Accessed: 04-Jun-2019].

[12] "TPCx-BB - Homepage." [Online]. Available: http://www.tpc.org/tpcx-bb/default.asp. [Accessed: 04-Jun-2019].

[13] Intel, "HiBench," 04-Jun-2019. [Online]. Available: https://github.com/Intel-bigdata/HiBench. [Accessed: 04-Jun-2019].

[14] IBM, "SparkBench — Bitbucket." [Online]. Available: https://bitbucket.org/lm0926/sparkbench/src/master/. [Accessed: 04-Jun-2019].

[15] M. Ferdman *et al.*, "Clearing the Clouds: A Study of Emerging Scale-Out Workloads on Modern Hardware," Aug. 2018.

[16] "BigDataBench | A Scalable Big Data and AI Benchmark Suite, ICT, Chinese Academy of Sciences." .

[17] "pumabenchmarks - Faraz Ahmad." [Online]. Available: https://engineering.purdue.edu/~puma/pumabenchmarks.htm. [Accessed: 04-Jun-2019].

[18] "SPEC - Standard Performance Evaluation Corporation." [Online]. Available: https://www.spec.org/. [Accessed: 04-Jun-2019].

[19] F. Raab, "TPC-C - The Standard Benchmark for Online transaction Processing (OLTP)," in *The Benchmark Handbook for Database and Transaction Systems (2nd Edition).*, J. Gray, Ed. Morgan Kaufmann, 1993.

[20] T. Hogan, "Overview of TPC Benchmark E: The Next Generation of OLTP Benchmarks," in *Performance Evaluation and Benchmarking, First TPC Technology Conference, TPCTC 2009, Lyon, France, August 24-28, 2009, Revised Selected Papers*, 2009, vol. 5895, pp. 84–98.

[21] M. Pöss and C. Floyd, "New TPC Benchmarks for Decision Support and Web Commerce," *SIGMOD Rec.*, vol. 29, no. 4, pp. 64–71, 2000.

[22] M. Poess, T. Rabl, and H.-A. Jacobsen, "Analysis of TPC-DS: the first standard benchmark for SQL-based big data systems," in *Proceedings of the 2017 Symposium on Cloud Computing, SoCC 2017, Santa Clara, CA, USA, September 24-27, 2017*, 2017, pp. 573–585.

[23] M. Pöss, R. O. Nambiar, and D. Walrath, "Why You Should Run TPC-DS: A Workload Analysis," in *Proceedings of the 33rd International Conference on Very Large Data Bases, University of Vienna, Austria, September 23-27, 2007*, 2007, pp. 1138–1149.

[24] R. O. Nambiar and M. Poess, "The Making of TPC-DS," in *Proceedings of the 32nd International Conference on Very Large Data Bases, Seoul, Korea, September 12-15, 2006*, 2006, pp. 1049–1058.

[25] M. Poess, T. Rabl, H.-A. Jacobsen, and B. Caufield, "TPC-DI: The First Industry Benchmark for Data Integration," *PVLDB*, vol. 7, no. 13, pp. 1367–1378, 2014.

[26] P. Sethuraman and H. R. Taheri, "TPC-V: A Benchmark for Evaluating the Performance of Database Applications in Virtual Environments," in *Performance Evaluation, Measurement and Characterization of Complex Systems - Second TPC Technology Conference, TPCTC 2010, Singapore, September 13-17, 2010. Revised Selected Papers*, 2010, vol. 6417, pp. 121–135.

[27] R. Nambiar, "Benchmarking Big Data Systems: Introducing TPC Express Benchmark HS," in *Big Data Benchmarking - 5th International Workshop, WBDB 2014, Potsdam, Germany, August 5-6, 2014, Revised Selected Papers*, 2014, vol. 8991, pp. 24–28.

[28] A. Ghazal *et al.*, "BigBench: towards an industry standard benchmark for big data analytics," in *Proceedings of the 2013 ACM SIGMOD international conference on Management of data*, 2013, pp. 1197–1208.

[29] "Home | STAC - Insight for the Algorithmic Enterprise | STAC." [Online]. Available: https://www.stacresearch.com/. [Accessed: 04-Jun-2019].

[30] "LDBCouncil |." [Online]. Available: http://www.ldbcouncil.org/. [Accessed: 04-Jun-2019].

[31] Y. Liu, H. Zhang, L. Zeng, W. Wu, and C. Zhang, "MLbench: benchmarking machine learning services against human experts," *Proc. VLDB Endow.*, vol. 11, no. 10, pp. 1220–1232, 2018.

[32] S. Karim, T. R. Soomro, and S. M. A. Burney, "Spatiotemporal Aspects of Big Data," *Appl. Comput. Syst.*, vol. 23, no. 2, pp. 90–100, Dec. 2018.

[33] M. Stonebraker, J. Frew, K. Gardels, and J. Meredith, "The SEQUOIA 2000 Storage Benchmark," in *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, New York, NY, USA, 1993, pp. 2–11.

[34] J. Patel *et al.*, "Building a Scaleable Geo-spatial DBMS: Technology, Implementation, and Evaluation," in *Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data*, New York, NY, USA, 1997, pp. 336–347.

[35] P. Werstein, "A performance benchmark for spatiotemporal databases," in *In: Proc. of the 10th Annual Colloquium of the Spatial Information Research Centre*, 1998, pp. 365–373.

[36] C. S. Jensen, D. Tiešytė, and N. Tradišauskas, "The COST Benchmark—Comparison and Evaluation of Spatio-temporal Indexes," in *Database Systems for Advanced Applications*, 2006, pp. 125–140.

[37] C. Düntgen, T. Behr, and R. H. Güting, "BerlinMOD: a benchmark for moving object databases," *VLDB J.*, vol. 18, no. 6, p. 1335, Apr. 2009.

[38] J. Yu, Z. Zhang, and M. Sarwat, "Spatial data management in apache spark: the GeoSpark perspective and beyond," *GeoInformatica*, vol. 23, no. 1, pp. 37–78, Jan. 2019.

[39] S. Hagedorn, P. Götze, and K.-U. Sattler, "Big Spatial Data Processing Frameworks: Feature and Performance Evaluation," in *EDBT*, 2017.

[40] M. Lissandrini, M. Brugnara, and Y. Velegrakis, "Beyond macrobenchmarks: microbenchmark-based graph database evaluation," *Proc. VLDB Endow.*, vol. 12, no. 4, pp. 390–403, Dec. 2018.

[41] D. Yan, Y. Bu, Y. Tian, A. Deshpande, and J. Cheng, "Big Graph Analytics Systems," in *Proceedings of the 2016 International Conference on Management of Data*, New York, NY, USA, 2016, pp. 2241–2243.

[42] G. Aluç, O. Hartig, M. T. Özsu, and K. Daudjee, "Diversified Stress Testing of RDF Data Management Systems," in *The Semantic Web – ISWC 2014*, 2014, pp. 197–212.

[43] G. Bagan, A. Bonifati, R. Ciucanu, G. H. Fletcher, A. Lemay, and N. Advokaat, "gMark: Schema-Driven Generation of Graphs and Queries," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 4, pp. 856–869, Apr. 2017.

[44] R. Angles *et al.*, "The Linked Data Benchmark Council: A Graph and RDF Industry Benchmarking Effort," *SIGMOD Rec*, vol. 43, no. 1, pp. 27–31, May 2014.

[45] A. Iosup *et al.*, "LDBC Graphalytics: A Benchmark for Large-scale Graph Analysis on Parallel and Distributed Platforms," *Proc VLDB Endow*, vol. 9, no. 13, pp. 1317–1328, Sep. 2016.

[46] M. Stonebraker, U. Cetintemel and S. Zdonik, "The 8 Requirements of Real-Time Steam Processing," ACM SIGMOD Record, vol. 34, pp. 42-47, 2005..

[47] Yahoo!, "Github," [Online]. Available: https://github.com/yahoo/streaming-benchmarks. [Accessed May 2019].

[48] M. Li, Y. Wang, Z. Li, v. Salapura and A. Biven, "SparkBench: A Comprehensive Spark Benchmarking Suite Characterizing In-memory Data Analytics".

[49] R. Lu, G. Wu, B. Xie and J. Hu, "Stream Bench: Towards Benchmarking Modern Distributed Stream Computing Frameworks," in IEEE/ACM 7th International Conference on Utility and Cloud Computing, London, UK, 2014.

[50] M. Arlitt, M. Marwah, G. Bellala, A. Shah, J. Healey og B. Vandiver, «IoTAbench: an Internet of Things Analytics Benchmark,» ICPE´15 Proceedings of the 6th ACM/SPEC International Conference on Performance Engineering, pp. 133-144, Jan 28 - Feb 04 2015.

[51] A. Shukla, S. Chaturvedi and Y. Simmhan, "RIoTBench: A Real-time IoT Benchmark for Distributed Stream Processing Platforms," Concurrency and Computation; Software Practice and Experience, Volume 29, Issue 21, 10 November 2017

[52] N. Shalom, *The Common Priniciples Behind The NoSQL Alternatives,* 2009. Blog: https://natishalom.typepad.com/nati_shaloms_blog/2009/12/the-common-principles-behind-the-nosql-alternatives.html (Accessed May 2019).

[53] Benchmarking Deep Learning Operations on Different Hardware: Baidu-Research/DeepBench. baidu-research. URL: https://github.com/baidu-research/DeepBench.

[54] Tensorflow Benchmarks. URL: https://www.tensorflow.org/performance/benchmarks. (Accessed May 2019)

[55] The Deep Learning Benchmarks. Contribute to DeepMark/Deepmark Development by Creating an Account on GitHub. DeepMark. URL: https://github.com/DeepMark/deepmark.

[56] Soumith Chintala. Easy Benchmarking of All Publicly Accessible Implementations of Convnets: Soumith/Convnet-Benchmarks. URL: https://github.com/soumith/convnet-benchmarks.

[57] Robert Adolf et al. 'Fathom: Reference Workloads for Modern Deep Learning Methods'. In: 2016 IEEE International Symposium on Workload Characterization (IISWC) (Sept. 2016), pp. 1–10. DOI: 10.1109/IISWC.2016.7581275. arXiv: 1608.06581. URL: http://arxiv.org/abs/ 1608.06581.

[58] Cody Coleman et al. 'DAWNBench: An End-to-End Deep Learning Benchmark and Competition', https://cs.stanford.edu/~deepakn/assets/papers/dawnbench-sosp17.pdf

[59] Reference Implementations of Training Benchmarks. Contribute to Mlperf/Training Development by Creating an Account on GitHub. MLPerf. URL: https://github.com/mlperf/training.

[60] MLPerf. URL: https://mlperf.org/.

[61] Hongyu Zhu et al. 'TBD: Benchmarking and Analyzing Deep Neural Network Training'. In: (16th Mar. 2018). arXiv: 1803.06905 [cs, stat]. URL: http://arxiv.org/abs/1803.06905.

[62] Jinhua Tao et al. 'BENCHIP: Benchmarking Intelligence Processors'. In: (23rd Oct. 2017). arXiv: 1710.08315 [cs]. URL: http://arxiv.org/abs/1710.08315.

[63] Getting Started - Deep Learning Bechmarking Suite. URL: https://hewlettpackard.github.io/ dlcookbook-dlbs/#/index?id=deep-learning-benchmarking-suite.

[64] Wanling Gao et al. 'Data Motif-Based Proxy Benchmarks for Big Data and AI Workloads'. Published in IEEE International Symposium on Workload, 2018, DOI: 10.1109/IISWC.2018.8573475

[65] Jonathan Saring: '11 Javascript Data Visualization Libraries for 2019' - https://blog.bitsrc.io/11-javascript-charts-and-data-visualization-libraries-for-2018-f01a283a5727

[66] Anush Fernandes: 'The Best Open Source Chatbot platforms in 2019' - https://blog.verloop.io/the-best-open-source-chatbot-platforms-in-2019/

[67] Bolanle Ojokoh Emmanuel Adebisi: 'A Review of Question Answering Systems', Journal of Web Engineering (JWE) 17(8):717-758 DOI: 10.13052/jwe1540-9589.1785, 2019

[68] Lorena Kodra, Elinda Kajo Meçe: Question Answering Systems: A Review on Present Developments, Challenges and Trends, (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 8, No. 9, 2017

[69] Stephanie Lemieux: Open-Source Options for Faceted Search, for the Budget Conscious User - https://www.cmswire.com/cms/information-management/opensource-options-for-faceted-search-for-the-budget-conscious-user-012110.php

[70] Justin Ellingwood: Hadoop, Storm, Samza, Spark, and Flink: Big Data Frameworks Compared - https://www.digitalocean.com/community/tutorials/hadoop-storm-samza-spark-and-flink-big-data-frameworks-compared, 2016

[71] Eran Levy: 7 Popular Stream Processing Frameworks Compared - https://www.upsolver.com/blog/popular-stream-processing-frameworks-compared, 2019

[72] Wissem Inoubli, Sabeur Aridhi, Haithem Mezni, Mondher Maddouri, Engelbert Mephu Nguifo: A Comparative Study on Streaming Frameworks for Big Data, LADaS 2018 - Latin America Data Science Workshop (http://ceur-ws.org/Vol-2170/paper3.pdf), 2018

[73] DataBench Deliverable "D1.2 DataBench Framework – with Vertical Big Data Type benchmarks" - available at https://www.databench.eu/public-deliverables/

[74] DataBench Deliverable "D1.3 Horizontal Benchmarks – Analytics and Processing" - this document - available at https://www.databench.eu/public-deliverables/

[75] DataBench Deliverable "D1.4 Horizontal Benchmarks – Data Management" - available at https://www.databench.eu/public-deliverables/