

S2CE: A Hybrid Cloud and Edge Orchestrator for Mining Exascale Distributed Streams

Nicolas Kourtellis
nicolas.kourtellis@telefonica.com
Telefonica Research
Barcelona, Spain

Herodotos Herodotou
herodotos.herodotou@cut.ac.cy
Cyprus University of Technology
Limassol, Cyprus

Maciej Grzenda
m.grzenda@mini.pw.edu.pl
Warsaw University of Technology
Warsaw, Poland

Piotr Wawrzyniak
800383@edu.p.lodz.pl
Lodz University of Technology
Lodz, Poland

Albert Bifet
albert.bifet@telecom-paris.fr
LTCI, Telecom Paris, IP-Paris
Paris, France

ABSTRACT

The explosive increase in volume, velocity, variety, and veracity of data generated by distributed and heterogeneous nodes such as IoT and other devices, continuously challenge the state of art in big data processing platforms and mining techniques. Consequently, it reveals an urgent need to address the ever-growing gap between this expected exascale data generation and the extraction of insights from these data. To address this need, this paper proposes *Stream to Cloud & Edge (S2CE)*, a first of its kind, optimized, multi-cloud and edge orchestrator, easily configurable, scalable, and extensible. *S2CE* will enable machine and deep learning over voluminous and heterogeneous data streams running on hybrid cloud and edge settings, while offering the necessary functionalities for practical and scalable processing: data fusion and preprocessing, sampling and synthetic stream generation, cloud and edge smart resource management, and distributed processing.

CCS CONCEPTS

• **Information systems** → **Data stream mining**; • **Computer systems organization** → **Cloud computing**; • **Computing methodologies** → **Machine learning**.

KEYWORDS

data stream analysis, edge analytics, cloud analytics, stream mining, machine and deep learning

1 INTRODUCTION

In the future Internet era, with hundreds of billions of devices, principle factors dominating the continuous utility of the Internet will be: 1) the massive population of devices and their intelligent agents, 2) the big, fast, and diverse data produced from them and their users, and 3) the need for large-scale, adaptive infrastructures to process and extract knowledge from these exascale data in order to help make critical, data-driven decisions. Intelligent agents already exist in different forms, and are well embedded in various ways in our everyday lives, either as passive data collectors, or active producers. They are instantiated in self-driving vehicles [94], phones [84], IoT devices [28], personal artificial intelligence (AI) assistants [21], factory or health sensors [48], smart utility meters [24], sensors in public transportation vehicles [76], chat bots [18], etc. Such entities typically interface with centralized cloud processing systems, responsible for collecting, analyzing, and visualizing the data produced. Finally, through machine learning (ML) and deep learning (DL), they can collect and learn new modalities of data, build new models and functionalities, and operate autonomously, making data-driven, critical decisions.

Human and smart agents' activities already produce big data workloads every day. Billions of messages per day are processed by Facebook Messenger and WhatsApp [41], while the Internet of Things (IoT) (~75 billion objects by 2025 [81]) continuously produces data without human intervention, leading to a dramatic increase of data volume and velocity. These numbers are only expected to exponentially grow through time: billions of devices (autonomous agents or user-devices) will generate big data continuously, as a stream, characterized by at least 4 important dimensions: Volume, Velocity, Variety and Veracity.

All these data are useful only when processed and modeled, and learnings are extracted in time to be used for appropriate decisions. Thus, in the last decade there has been a high demand for data mining tools that allow data practitioners to compute complex ML models on big data streams produced by different sources and collected in centralized locations for processing. To partially satisfy the need for computation power to process these data, we have witnessed an exponential growth of cloud computing, where the market itself is expected to reach 150 billion USD by the early 2020s [56]. Various vendors have introduced different types of clouds (private, public, and hybrid), with heterogeneous resources available in each, varying with respect to computation (CPU/GPU), memory, and network

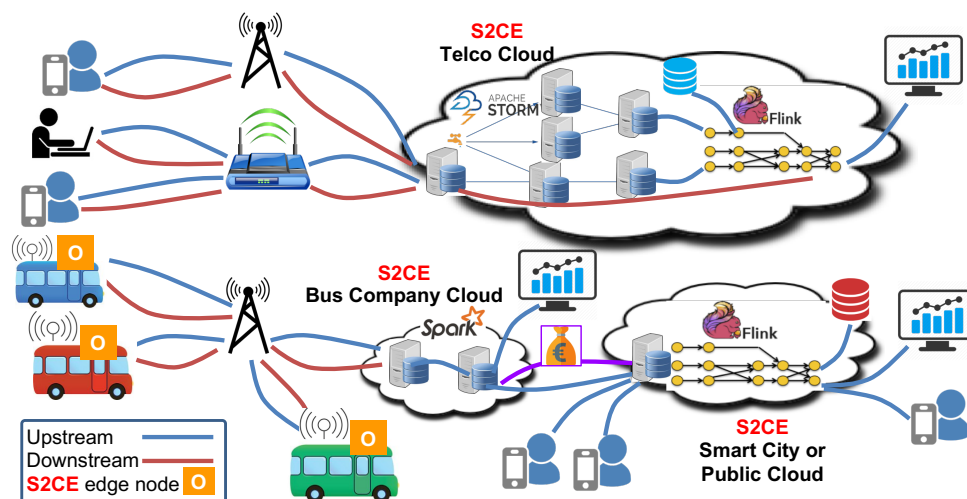


Figure 1: Examples of functionalities and usage of the *S2CE* platform ecosystem.

capacities. However, all such clouds are typically not interoperable and do not facilitate data or computation portability between them. Thus, customers are typically stuck in vendor lock-in, forced to rely on data analytic services of a single cloud provider, leading to lost opportunities in revenue and business from both sides (customers and providers) [61]. Finally, customers who choose private clouds are typically forced to deal with significant overhead in setting, managing, and manually tuning cloud resources.

Furthermore, the current model of data collection and processing is not sustainable. Billions of IoT/edge devices generate tens of times more data than the 30+ million nodes across public and private cloud centers [74]. This influx of data cannot be processed in real time due to latency issues, lack of scalability in some cases, limited bandwidth in wireless connections (e.g., rural or congested areas), or compliance and privacy on data access, and does not allow customers to make real-time, data-driven decisions for their businesses. The cloud industry has recently moved into a hybrid of cloud-edge computing, but this creates a whole set of new challenges regarding interoperability and APIs, managing heterogeneous capacities, workload offloading, data integrity and privacy, storage decentralization, application restructuring, etc. [73].

Proposal. To address the aforementioned challenges, this paper proposes *Stream to Cloud & Edge (S2CE)*, a first of its kind, optimized, multi-cloud and edge orchestrator. In fact, *S2CE* enables machine and deep learning on big data streams using hybrid cloud and edge resources, while offering the necessary functionalities for practical and scalable event-based processing: data fusion and preprocessing, sampling and synthetic data stream generation, cloud and edge smart resource management, and distributed event processing.

Motivating scenarios. Future industrial settings can impose potentially diverse and interdisciplinary constraints and requirements to *S2CE* and its tools. However, its unique architectural design and functionalities enable it to flexibly accommodate different future use-case scenarios. Figure 1 illustrates some of these functionalities and industrial setups. *S2CE* can consume big data from different sources, destined for different types of analysis. These input data

can range from data-in-motion, such as events produced from sensor and other readings from mobile or IoT edge devices, up to fully processed data-at-rest or in-motion from other complete cloud platforms. The distributed and parallel nature of *S2CE*'s components allow the platform to scale gracefully to accommodate future exascale volumes and velocities.

Depending on the capabilities of the source nodes, pre-models could be executed at the edge to alleviate computing pressure from the main cloud platform. Results can be consumed by end-users, the company managing the *S2CE* platform, or shared and even sold to other downstream companies. In fact, interconnection APIs made available by the platform can enable such data to be shared across companies, with appropriate payment schemes in place, while respecting users' privacy. For example, a bus company may be collecting data to optimize bus routes and perform data-driven business decisions, but can also sell such data to a City Council to bootstrap their effort to optimize traffic, and reduce congestion and pollution in the city. This crucial property will drive high business innovation in the data sharing markets, as required and expected by the emerging world Data Economy.

Contributions. With this paper, we make the following contributions in the domain of distributed, event-based processing systems:

- Analyze state-of-the-art stream data processing methods, libraries, and systems that are widely used in academia and industry.
- Identify needs and challenges faced by – as well as success criteria expected by – the industrial and R&D sectors.
- Propose a novel, hybrid, cloud-edge architecture that can address these challenges, with several key design objectives in mind.
- Discuss the innovation potential of the proposed architecture across different dimensions.

2 STATE-OF-THE-ART ANALYSIS

The current state-of-the-art in the space of distributed data stream processing systems is populous and covers various aspects of the needs that the industry has from such systems. In the next paragraphs, we briefly cover efforts from academia and industry to

address such needs, and identify pending issues and gaps that a new platform, such as our proposed *S2CE*, should address:

- Big data stream processing systems (Section 2.1)
- Cloud resource management and tuning (Section 2.2)
- Distributed stream processing at the edge (Section 2.3)
- Machine and deep learning over data streams (Section 2.4)
- Data transformation techniques (Section 2.5)

2.1 Big Data Stream Processing Systems

There are currently several open source Distributed Stream Processing Engines (DSPE), such as Apache Storm [12], Apache Samza [10], and Heron [54]. While they support developing ML applications, they do not have dedicated ML libraries. Apache Spark [11], originally a batch processing framework, now offers streaming support for micro-batches or continuous streams. Spark ML and MLLib are Spark’s ML libraries with only simple linear classifiers and clustering algorithms for streaming, none of which are state of the art. Also, StreamDM [82] is an Apache-licensed, open source software for mining big data streams using Spark Streaming, and developed by Huawei. Apache Flink [6] is a processing framework that focuses on streaming tasks and arranges them in directed acyclic graphs, similar to Spark’s topologies. Its ML library, FlinkML, supports batch-based ML. Apache Apex [3] and Apache Beam [4] are unified stream and batch processing engines containing basic ML libraries.

In the Cloud front, Google Cloud Dataflow [39] is a streaming data processing system that can emulate batch processing and has some support for ML. MS Azure [59] is Microsoft’s cloud system performing both batch and stream processing. The batch processing engine has ML algorithms, but the Streaming Analytics tool can only compute aggregation and statistics, without any other streaming ML. AWS Kinesis [13] is Amazon’s offering for processing data streams in real-time but, similar to MS Azure, it does not provide native support for streaming ML.

Massive Online Analysis (MOA) [20] consists of well-known online algorithms for streaming classification, clustering, and change detection mechanisms. However, MOA only runs in a single machine and lacks high-performance integration interfaces (e.g., with the widely-used Kafka), making it non-usable in industrial big data deployments. Vowpal Wabbit [88] is a streaming ML framework based on the perceptron algorithm with a specific focus on reinforcement learning and optimized for a multi-core single-node setting. Jubatus [49] is a ML framework for stream mining that establishes tight coupling between the ML library and the underlying custom-built DSPE, which limits the framework’s applicability and extensibility. Finally, Apache SAMOA [9] allows for distributed computation of several ML algorithms over four DSPEs, namely Storm, Flink, Samza, and Apex.

A future processing platform should take advantage of existing engines (be it pure streaming or hybrid) for executing a given streaming ML task, while offering an API for extending ML algorithms available and runnable on the platform. Table 1 summarizes how the desired platform should differ from, and advance the features and functionalities provided by the most relevant prior research projects and large-scale data processing systems used by the industry today.

Table 1: Feature comparison between the desired platform and related large-scale data processing tools and systems.

Features/Capabilities	Apache Storm	Apache Samza	Apache Spark	Apache Flink	Apache Apex	Apache Beam	Google CD	MS Azure ML	MOA	Vowpal Wabbit	Jubatus	Apache SAMOA	Desired Platform
	×	!	✓	!	!	!	!	!	!	!	!	!	
Stream integration components	✓	✓	✓	✓	✓	✓	×	×	×	×	!	✓	✓
Data preprocessing and fusion	!	×	!	!	×	×	×	×	!	!	✓	!	✓
Built-in synthetic data generator	×	×	×	×	!	!	×	×	!	×	×	!	✓
Stream-based machine learning	!	!	!	!	!	!	!	!	!	✓	✓	!	✓
Stream-based deep learning	×	×	×	×	×	×	×	×	×	×	×	×	✓
Resource management	✓	!	✓	✓	✓	✓	×	×	×	×	!	!	✓
Distributed platform	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Open license (Apache preferred)	✓	✓	✓	✓	✓	✓	×	×	!	!	!	!	✓

2.2 Cloud Resource Management and Tuning

Cloud Compute and Storage services are traditionally utilized for creating and provisioning Virtual Machines (VMs) to host batch and stream processing applications [77]. This method offers a tight control of the infrastructure where the software is running, but comes at high cost of maintenance, as each machine has to be provisioned individually. When it comes to scalability and optimization, there is a strong need for careful capacity planning and manual intervention even with automated tools [63]. To alleviate these issues, the recent trend is to use containers instead of VMs as the minimal computation unit in the cloud. Docker makes it easy to generate and run such containers, while Docker Compose is a lightweight solution for orchestrating them [33]. For big workloads, Kubernetes [53] is a better fit for automating application deployment, scaling, and management. Kubernetes supports scale up or down a set of containers, but the decision has to be manually programmed. Also, when nodes fail or in overload cases, there is lack of automated tools for infrastructure management in a Kubernetes cluster.

Data management systems have grown in scale, complexity, and number of installations [14]. Such systems contain 100s of configuration parameters and execute on 1000s of nodes that must be properly configured and managed [42]. Hence, it is crucial to automate the process of resource management, as well as optimization and tuning of application performance. The problem of resource allocation deals mainly with gathering and assigning resources to applications, while scheduling deals with allocating tasks to resources [2]. Past works focused on different aspects of the problem: [38] investigates the scheduling problem satisfying load balancing and cost considerations; [68] employs design-time knowledge and benchmarking method to deal with scheduling on heterogeneous clusters; Re-Stream [83] focuses on energy-efficient resource scheduling; [93] addresses adaptive scheduling. Others focus on automatically optimizing and tuning workloads using various techniques such as cost-based (e.g., [43, 50, 87]) or ML-based (e.g., [16, 89, 90]).

Overall, a future platform should employ new and advanced statistical and machine learning techniques for (i) understanding application behavior and cloud resource usage, (ii) optimizing the

provisioning of cloud resources for applications across different providers in an automated and vendor-neutral manner, and (iii) automatically tuning applications to increase performance and meet SLAs.

2.3 Distributed Stream Processing at the Edge

Systems for distributed stream processing have traditionally been designed to run on clusters or in the cloud. However, processing all the data there can introduce latency delays due to data transfer, which makes near real-time processing difficult to achieve. In contrast, edge computing has become an attractive solution for performing certain stream processing operations, and hence (i) reduce end-to-end latency and communication costs, (ii) enable services to react to events locally, or (iii) offload processing from the cloud [45, 69]. Computing, storage, and network resources located at the network edge, however, are more constrained than those deployed in the cloud. While edge computing is often used to reduce latency of delivering content to mobile end-users, the emergence of application domains such as IoT require data events to be treated locally, under short time delays.

The deployment of data stream processing applications onto heterogeneous infrastructure has been proven to be NP-hard [17]. Moreover, moving operators from cloud to edge devices is challenging due to limitations of edge devices [30]. Existing work often proposes placements strategies considering user intervention [69], whereas many models do not support memory and communication constraints [22]. Studies also consider all data sinks to be in the cloud, with no feedback loop to actuators located at the edge [60]. Thus, no current solution covers scenarios involving smart cities, precision agriculture, and smart homes comprising heterogeneous sensors and actuators and time-constraint applications.

A future data stream processing service should be able to orchestrate the deployment of processing tasks and achieve resource elasticity under highly distributed environments comprising edge computing and clouds. A lightweight version of the platform should be able to take advantage of existing solutions such as Apache Edgent [5] in order to decentralize the online ML and mining towards the edge of the ecosystem.

2.4 Machine/Deep Learning over Data Streams

To effectively deal with streaming data, models must be able to adapt to patterns evolving over time by detecting changes in a fast and accurate way [37]. Thus, shift detection mechanisms are necessary in the context of devised ML methods, to render them self-adaptive, similar to DDM [36], EDDM [15], and ADWIN [19]. Past work [26, 92] also demonstrates the need to better understand the ML model structure and claim that such understanding is possible through informative visualization of the model structure. New sophisticated visualization approaches are the Forest Floor [91], interaction importance extraction [34], and the factorMerger [78]. Still the visualization of model structures is less mature than that of raw data provided by popular BI tools such as Tableau [85] and Power BI [65]. Hence, there is a strong need for designing and implementing adaptive distributed algorithms for learning from streaming data, novel tools for visualization of ML models, and

tools for tracing and monitoring the changes in the evolution of streams and of the ML models.

Conventional deep networks do not allow for uncertainty representation, which is key for addressing learning from streaming data. Bayesian inference-based variants [31] offer a solution that can account for two types of uncertainty: (i) heteroscedastic uncertainty, which captures noise inherent in the observations due to temporal irregularities in data sampling; and (ii) model uncertainty, which accounts for uncertainty in the parameters. However, there are two main issues with using Bayesian updating on data streams. First, Bayesian inference computes posterior uncertainty under the assumption that the model is correct, rather than being an approximation. Second, existing approaches either explicitly model the time series, at the cost of low inferential performance, or assume that the data are exchangeable, i.e., that the underlying distribution does not change over time.

To address these issues, a future machine learning stream processing platform should adopt ideas from Bayesian non-parametrics, such as population-driven posteriors [58], and introduce them into the configuration of postulated Bayesian deep networks, and adopt self-adaptive prior assumption mechanisms in the context of approximate variational inference. These adaptations should enable the platform to build more complex, accurate ML and DL models.

2.5 Data Transformation Techniques

Raw data often need to be transformed before processing. Typical architectures use Apache Flume [7] or Kafka [8] to first capture data of interest from distributed sources, and then apply preprocessing such as filtering and format conversion. Given the popularity of these systems, interfaces and methods of Kafka and Flume should be utilized for receiving input from multiple upstream sources and producing output consumable by downstream units. For streams with delayed labels, methods have been proposed such as (i) drift detection applicable in stream classification [57], (ii) classification method inspired by micro-clusters [80], and (iii) analysis of upper loss bound of multiple expert system trained in nonstationary environment with verification latency [32]. Such methods are not available in DSPEs libraries such as Spark ML or FlinkML.

Traditional dimensionality reduction techniques do not apply for stream data as the processed data arrive at real-time and the reduction must happen online, with no-multiple loop, batch-based algorithms. [27] explored statistical inference methods for reducing dimensions of streams using hashing projections to derive efficient estimators of cardinality. [51] discussed different subspace tracking methods for reducing dimension space in streams.

Studies like [1, 23] offered methods on how to produce synthetic streams from real data by inferring underlying statistical distributions. Such approaches do not work for streams with concept drifts, and protecting privacy and confidentiality cannot be done with fixed privacy preserving rules. Data generation allows altering volume, velocity, variety, and proportion of records linked to individual features, while maintaining inter-feature dependencies. Data generators in [29, 67] allow for systematically producing large data volumes. [20] offers several synthetic stream generators for benchmarking ML methods, but without capabilities to scale properly to produce large volume/velocity streams.

Table 2: Expected industrial challenges and how the envisioned platform’s design should address each challenge.

Expected Industrial Challenge	(Objective) How does <i>S2CE</i> address the challenge?
Heterogeneity	(O1) Handling diverse types of cloud computing resources
Scalability	(O1) Distributed and parallelized dynamic analytics for real-time learning
Data-in-motion and data-at-rest	(O1) Processing data seamlessly at the same time without extra system overhead
Hybrid (central+edge) big data architectures	(O2) Optimizing an efficient mixture of central and edge resources
Decentralization & edge	(O2) Computing at edge for faster, more scalable, energy efficient processing
Data/AI/predictive/prescriptive analytics	(O3) Using distributed deep and machine learning
Stream analytics frameworks & processing	(O3) Minimal development effort, scalability, processing speed
Advanced business analytics	(O3) Intelligence to empower companies for accurate, instant, data-driven decisions
Heterogeneity	(O4) Handling diverse data, modeling, and input/output interfaces
Semantic interoperability	(O4) Facilitating data and model exchange between vertical data silos
Data quality	(O4) Providing curation methods for data filtering, quality assessment, improvement
Distributed trust infrastructures	(O4) Managing data in anonymized and decentralized fashion

A future platform should provide instance data preparation for ML, data fusion, methods to deal with evolving, incomplete, or delayed data records and events, delayed labels, and time-spanned joins of streams. Methods should also be explored based on new-found advancements in DL, capturing hidden similarities within streams, compressing more effectively data for efficient processing downstream. Finally, such a platform should provide data generation process to (i) handle non-stationarity of data distributions due to changes in the environment, (ii) scale appropriately to produce required volume and velocity of data, (iii) handle concept drift and skewness observed in real data.

3 INDUSTRIAL CHALLENGES & OBJECTIVES

In the next decade, the big data stream mining community will face several industrial challenges, as summarized in Table 2. In fact, these challenges drive particular industrial needs, for which a future processing platform should adequately address. In the next paragraphs, we first summarize four fundamental industrial needs stemming from our analysis of the current state-of-the-art, as well as success criteria that must be satisfied to address these needs. Then, we outline the desired objectives that the future platform must have by design, to fulfill these needs and success criteria.

3.1 Industrial Needs & Success Criteria

Industrial Needs. Based on the current challenges identified in Section 2, and the expected landscape on data generation and processing systems, we anticipate that the industry in data analytics on cloud infrastructures will have the following four future needs:

- N1:** Computing platforms that can decentralize processing at the source of data during generation (i.e., edge), to alleviate pressure of computation and storage at cloud/centralized infrastructure.
- N2:** Computing platforms that use resources on heterogeneous multi-clouds (public/private) for data mining.
- N3:** Computing platforms that can automatically self-tune and orchestrate their resources for optimal resource allocation and use between cloud and edge.
- N4:** Advanced machine and deep learning tools, capable of pre-processing and analyzing exascale streams at real time, both at

the edge where data are produced and at the cloud where more complex modeling can be done.

Success Criteria. A proposed platform can effectively address the above needs if it satisfies at least the following criteria:

- S1:** The platform should scale appropriately and in a distributed fashion to sustain throughput while processing exascale data streams expected in the next 5-10 years.
- S2:** The platform should produce real-time data insights with microsecond updates, based on advanced machine and deep learning models computed on incoming streams.
- S3:** The platform should shift workload between cloud and edge resources seamlessly; increased latency and reduced model performance should not violate agreed SLAs.
- S4:** The platform should integrate fully with current and future big data processing systems, and facilitate easy adoption and usability by big data practitioners and engineers.

3.2 Desired Platform Design Objectives

The envisioned platform must have key properties embedded in its design to fulfill the following four main objectives, thus satisfying the future industrial needs of big data stream processing. Table 2 summarizes how each objective will address each industrial challenge outlined earlier. In effect, the envisioned platform should:

- Objective O1:** Be a data processing platform capable of scaling in a distributed fashion on cloud resources to ingest exascale streams, and utilize smart cloud resource management and self-tuning for reduced energy consumption, increased efficiency, easy configuration and maintenance.
- Objective O2:** Be a hybrid cloud-edge architecture capable of scaling in a decentralized fashion to preprocess big data streams on edge nodes, by utilizing smart edge resource management for workload migration from cloud to edge nodes, for energy efficiency and reduced latency.
- Objective O3:** Provide the next generation of advanced stream analytics, with distributed machine and deep learning (ML/DL), to support predictive and prescriptive analytics of both data-at-rest and data-in-motion in a unified manner, that empower business intelligence in the new paradigm of Data Economy.

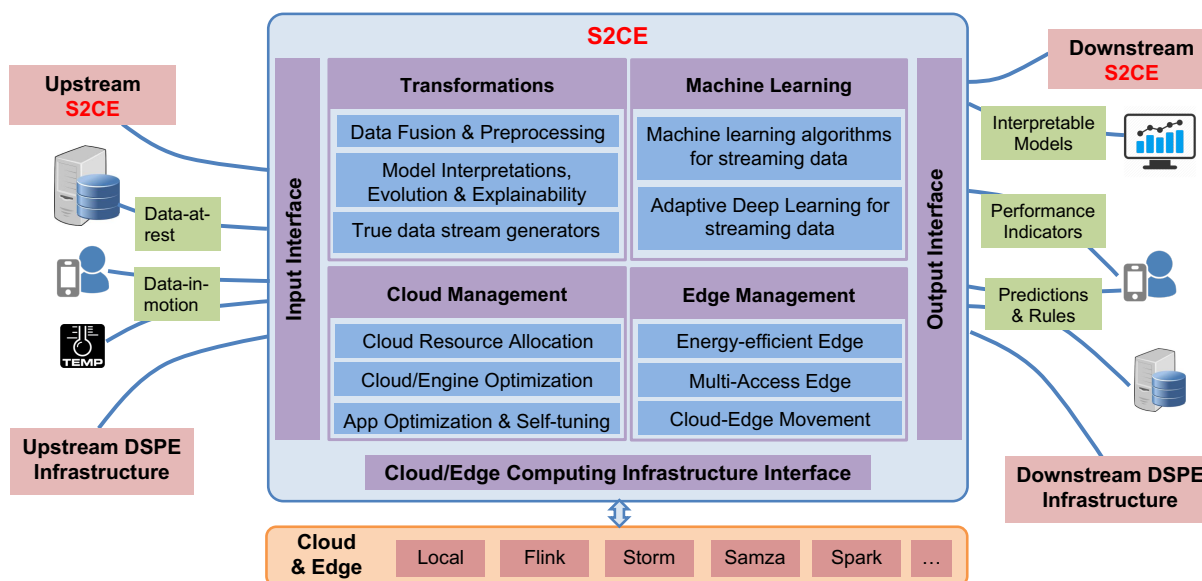


Figure 2: Global architecture of the proposed *S2CE* platform.

Objective O4: Support tools for input/output data transformation and synthetic data stream generation based on real-world statistic distributions, for ensuring data quality, interoperability, and replicability, while enabling the privacy-preserving and confidential sharing of data.

4 PLATFORM DESIGN

Our proposed platform, *S2CE*, is a complete, cloud-edge orchestration platform, based on four key components for input/output transformations, machine and deep learning processing, as well as cloud and edge resource management and tuning. *S2CE* aims to accelerate the building of tools that facilitate and enhance real-time artificial intelligence over voluminous and heterogeneous streams of data from IoT and other sources, in order to extract meaningful knowledge and perform extreme-scale predictive analytics over cloud and edge computing infrastructures. A key objective of this platform is to provide the big data industry with standardized interconnection methods and a stream mining framework within the Apache big data ecosystem, to enhance and automate decision making processes. In the next paragraphs, we provide an overview of the architectural components needed to instantiate the envisioned platform and its desired properties.

4.1 Architectural Components

The high-level vision is to build an integrated platform that includes: (i) input of data from different upstream infrastructures (these can be other *S2CE* pipelines, edge components deployed on top of IoT or mobile nodes to perform data preprocessing, etc.); (ii) cloud- and edge-based modules combining ML algorithms and stream computing infrastructure; and (iii) output modules for data visualization or input to downstream pipelines. Figure 2 shows the architecture of this conceptually novel platform with its various components described in detail below.

Input Interface: APIs providing standardized, secured interconnections will allow the mixing of multi-input data streams. Apart from the edge, such streams can come either directly from IoT-related sensors, or data producers including stream platforms positioned upstream in the pipeline. These data can be of different types (data-in-motion or data-at-rest), formats, and arriving at different velocities and volumes. Due to its adaptive features described later, the platform will be capable of consuming them without penalties in performance (throughput, prediction accuracy, etc.), or security. It will also be capable of fusing, preprocessing, aggregating, or sampling diverse data.

Transformations: The data streams will feed the Transformations component, with the following functionalities:

- *Data preprocessing and fusion:* Data can be processed and transformed to improve the quality of data and learning. The transformations will be either instance- or attribute-based and will allow the imputation of missing data, fusing, and normalizing when multiple data types provided, and dealing with delayed data. This module will allow handling of complex, multi-featured data, with dimensionality reduction techniques either for machine learning or modeling within synthetic stream generators.
- *Visual exploration and model explanation:* ML models built on top of streams are difficult to monitor their structure and performance. This module will contain methods for interpretable ML that efficiently summarizes and visualizes drift, model structure, evolution and performance.
- *Changes in Online Models:* Changes in characteristics of data or models require human intervention and, if not addressed in time, could lead to model performance degradation. This module will also identify and visualize significant changes and trends in data and model performance.
- *Privacy-preserving stream generators:* To test end-to-end platform performance and usability, novel synthetic data generators will be

developed based on game neural networks. These generators can be used for sharing of synthetic data reflecting closed business data while preserving data owners' privacy, for benchmarking of both ML methods and end-to-end applications under varied load.

Machine Learning: This component will take inputted data and perform various algorithms for fast learning. The main challenge will be that such algorithms need to be incremental, use a small amount of time and memory for processing, and adapt to the changes on the streams:

- *ML streaming algorithms:* The platform will contain the necessary abstractions (and a library) so that complex ML algorithms can be implemented for classification, clustering, anomaly detection, frequent pattern mining, and reinforcement learning, designed to scale in a distributed fashion, using cloud deployments to consume very large data streams in real time.
- *Self-adaptive DL algorithms:* DL (Deep Learning) algorithms that evolve and adapt on the streamed data, in a self-enforced fashion, will also be supported.

Cloud Resource Management: This component will manage cloud resources efficiently, starting from the basic mechanics needed for cloud resource provisioning, to algorithmic monitoring and distributed task management, to self-tuning streaming applications automatically:

- *Resource Allocation, Deployment & Monitoring:* Fundamental methods will be deployed to allow *S2CE* to monitor and control available resources at different cloud providers and allocate resources in them as needed, deploying computation tasks and monitoring execution.
- *Cloud/Engine Algorithm Management:* Different cloud and computing engine parameters impact streaming applications in different ways. This module will be responsible for making initial and recurring provisioning and configuration decisions to meet performance, monetary budget, and/or energy efficiency objectives, while employing the most appropriate cloud and computing engine and, if needed, applying data transformation and reduction strategy satisfying these objectives.
- *Optimization & Self-Tuning of Cloud Applications:* Given a ML task to be performed on an input data stream, the platform will be able to self-tune using ML algorithms to pick the best streaming engine and appropriate parameter settings for the execution of the task.

Edge Resource Management: This component will manage edge resources efficiently, by understanding what streaming computation can be offloaded from cloud to edge resources, where (i.e., to which edge nodes), when this offloading should be done (or reversed), as well as how the offloading will be done:

- *Energy-Efficient Edge Placement:* Many streams are simple enough to be preprocessed at their origin (edge) to reduce communication and processing costs at the main cloud platform. Sampling and summarization algorithms will be applied at the edge (e.g., IoT nodes, mobile devices), while guaranteeing property preservation of streams (e.g., via unbiased sampling), and dynamic reconfiguration of processing services according to computing and network availability of edge nodes.

- *Multi-access Edge Computing (MEC):* The platform will contain a MEC module to enable collaborative deployment of applications on the basis of telecommunication and computing resources located closer to user equipment.
- *Computation Movement between Cloud and Edge:* Fundamental blocks will be deployed to monitor utilization of the edge computing infrastructure and, in coordination with the Resource Allocation, Deployment & Monitoring module, enable ML streaming algorithms to offload computation from cloud to edge resources and vice versa.

Output Interface: This component will implement standardized, secured interconnections to allow the algorithmic results from the ML module to be outputted for downstream streaming engines, *S2CE* instantiations, and even end-users to consume on their devices in a seamless, secure way through a pipeline. *S2CE* will offer new connectors and interfaces to create stream processing pipelines and output its resulting predictions and models for other engines to use downstream. The output data streams will be splittable to different formats and substreams, depending on the utility of the overall pipeline.

Computing Infrastructure Interface: This component will allow *S2CE* to operate on top of several well-established distributed stream processing engines (DSPEs) such as Apache Flink, Storm, and Spark Streaming. In addition, it will be extensible to work with future DSPEs, exposing flexible APIs for the definition of new data input and output types.

4.2 S2CE Performant Interconnections

The various modules of *S2CE* will be interconnected through two standardized APIs, as shown in Figure 3. These interfaces will be high-performant, secured, capable of exchanging voluminous and fast, exascale data, as well as rich data, depending on the modules consuming them.

The *External API* will be responsible for consuming data inputted to the platform from various upstream sources (other platforms or end-user devices), or outputted to downstream platforms and devices for consumption (further analysis, visualization, and storage). The *Internal API* will be responsible for allowing the various modules to interact and exchange raw data, tuned parameters, models, etc. For example, the Cloud and Edge Management components will tune the computing infrastructure for a given data stream and model applied, and provide such tunings to the Input Interface for adjusting sampling rates. Furthermore, the Transformations and Machine Learning components will consume raw streams or pre-models already built at the edge, to build final, full-blown ML models. Performance metrics will be used to monitor and adjust resource consumption both at the cloud and edge infrastructure.

These APIs allow *S2CE* to have the following key features with respect to big data mining systems:

- High extensibility and flexibility to accommodate future heterogeneous data sources and pools, distributed computing platforms, ML algorithms, tuning algorithms, etc.
- Scalability to consume data from multiple sources with extreme volumes and velocity.
- Federated-ness for connecting and building big data stream pipelines across different clouds and edge.

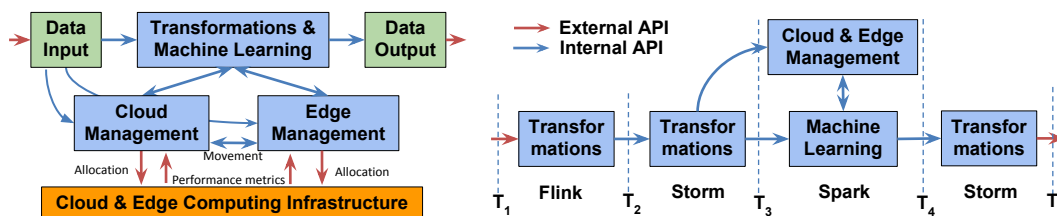


Figure 3: APIs and component interconnection. The components can run in one *S2CE* instance (left), or multiple *S2CE* instances and on different DSPE systems (right).

- Standardized interoperability with other stream or batch platforms, for faster and efficient data sharing.

5 INNOVATION POTENTIAL

The *S2CE* platform will enable the execution of machine and deep learning algorithms on a variety of existing (and future) cloud providers and DSPEs that are widely used in industry. This section outlines the key innovation potential of the *S2CE* platform in various domains.

5.1 Cloud Provisioning and Orchestration

The cloud landscape is becoming quite complex with different cloud providers and offerings that cover the full spectrum from IaaS to PaaS and SaaS with several intermediate solutions [35, 55]. Vendor choosing and careful capacity planning is nowadays inevitable, given the different prices, APIs, tools, and services that may vary from provider to provider [70]. Nowadays, this process is necessary given that once a provider is chosen, many workflows such as provisioning, fault tolerance, development, DevOps, QA, etc., are consequently tied to the offerings of the chosen provider. In fact, all the aforementioned dependencies make the process of changing providers or even collaboration of processes between different providers a very difficult task [61]. It is then necessary to develop solutions that may encompass the provisioning of resources and orchestration of software across different providers in a vendor-neutral manner. That way, splitting processes across several providers or even changing from one to another, becomes a much easier task.

When it comes to data processing and ML, it is even more important, given the myriad of software that is available and may or may not collide with the different managed solutions available in the major cloud providers such as Amazon, Microsoft, and Google [70]. Many enterprises might go for a single product based on a single platform just because of the convenience of such managed solutions, getting into a game of vendor lock-in that makes it difficult to migrate to any other, or even to run some tasks on premises [61]. *S2CE* can be easy to install and manage, as well as optimize resource allocation, software deployment, and runtime tuning across different cloud providers in a vendor-agnostic way.

5.2 Edge Preprocessing and Movement

Given the heterogeneous nature of the Edge, resource optimization and task orchestration is an issue that is difficult to tackle in a single way [72, 86, 95]. Advances in execution paradigms such

as containers and container-orchestration software may make it easier [44], but given the difference in architecture and computation or storage capacity that any node in an Edge infrastructure might have, task placement strategies and resource monitoring become essential tasks to guarantee the correct execution of preprocessing software [30, 75, 79].

Even with that, especially in streaming data analytics, the volume of ingestion might grow unexpectedly, overloading nodes that were not meant to deal with such a volume of incoming data. In some cases, scalability can be achieved when underloaded nodes can collaborate in distributed tasks, or even between different edge infrastructures closely collocated. However, there is a point in which a mechanism is needed to offload this workload to the cloud, where computation power might grow as needed. *S2CE* will provide task placement mechanisms and algorithms across edge nodes, as well as offloading methods for tasks to be migrated to the cloud from edge, seamlessly and without impacting performance or latency beyond agreed SLAs.

5.3 Data-driven Decisions based on Streams

Big Data projects, to be truly beneficial for organizations, have to provide not only data storage and retrieval capabilities, but also support for data-driven business decisions, inevitably necessitating data analytics tools [25, 64]. In the case of big data resources, due to little or no a-priori knowledge on inter-dependencies present in the data, ML techniques are of particular use. In a traditional setting, they are executed on a regular basis in batch mode. However, in many industry cases nowadays, a tool is needed for advanced analytics on data streams coupled with ML, and performed in near-real time. Thus, it is of no surprise that this need for online analytics raised the interest in stream processing engines from major industrial players [40] (e.g., IBM, Twitter, LinkedIn, Google, Amazon). *S2CE* will provide exactly such analytics to support exascale data-driven decisions on business and innovation.

5.4 Innovation Exchange with Apache Ecosystem's Open Source

The big data interest was followed by a rapid development of DSPEs, such as Apache Storm, Samza, and more recently Apache Spark and Apache Flink, which are the dominating solutions in the area of online analytics. These open source products achieved major popularity, exceeding the popularity of commercial, closed solutions. This is clearly confirmed by the offerings of data processing vendors such as Oracle providing its Oracle Table Access for Hadoop

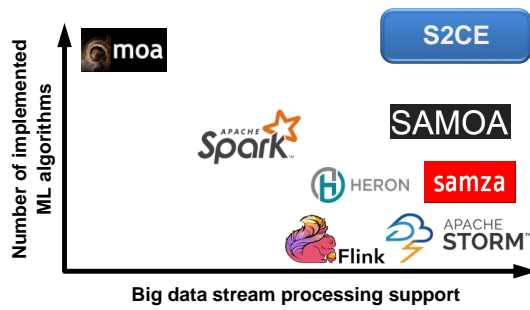


Figure 4: S2CE positioning in the big data stream processing ecosystem.

and Spark components [62]. Other vendors, such as SAS Institute and IBM, developed their own stream-related offerings, such as the SAS Cloud Analytics [71] and IBM Streams [46], respectively. Some of these systems can or even are advised to be blended with open source ML solutions including Spark MLLib [47]. In addition, commercial offerings from Cloudera and other companies integrating big data projects into comprehensive platforms have provided support and integration with these dominant streaming engines and analytics.

The risks introduced by developing new solutions, and major costs of big data projects arising from hardware infrastructure are commonly observed. Hence, many companies build their big data solutions based on open source platforms yielding no license cost, or costs for core development of the tools. This is clearly confirmed by the industrial success of Apache big data projects. Naturally, S2CE must integrate organically with the Apache ecosystem projects as a next generation big data solution, and enable innovation exchange with existing and future industrial solutions.

5.5 Unifying Efforts for a Stream ML Library

Many real life big data use cases demand the inclusion of ML techniques in data processing. This requirement was partly answered by open-source libraries extending the functionality of Apache Storm, Spark, and Flink [52]. Unfortunately, major deficiencies of this approach can be already observed. The effort of the open source community is fragmented across different DSPE projects. As the popularity of DSPEs changes significantly over relatively short time periods depending on industrial support, there is a risk that none of the ML libraries accompanying nowadays DSPEs will reach its maturity, before DSPEs are replaced by a competing and more complete framework [66]. To the point, the industry focus already gradually moved from Apache Storm to Spark, and lately, DSPEs such as Flink are appreciated and significant developing effort is applied. Importantly, none of the ML libraries accompanying these DSPEs provide key stream mining techniques developed in the research community such as classification, clustering, or regression.

Hence, on the one hand, there is a growing gap between the constantly expanding portfolio of stream mining techniques in research projects such as MOA [20], and the limited availability of them in DSPEs adopted by the big data industry. On the other hand, projects such as MOA include extensive implementations of state-of-the-art stream ML techniques. However, being a research

project, MOA does not include distributed processing support and integration interfaces or supervision interfaces, which are mandatory for industrial use. S2CE envisions to unify efforts in the open source space to produce a comprehensive ML library for big data stream mining (see Figure 4 for a comparison).

6 CONCLUDING REMARKS

This paper makes the case for the need of an exascale data stream mining platform, that can address the current and future industrial challenges in big data processing. Such a platform must scale seamlessly between available cloud and edge infrastructures to tackle the ever increasing volume, velocity, variety, and veracity of data expected in the future, and can compute complex machine and deep learning models needed for future, data-driven business decisions. The paper proposes an architectural design for S2CE, a platform that addresses all these needs and industrial challenges in a one-stop-shop solution.

S2CE is a first of its kind, optimized, multi-cloud and edge orchestrator, easily configurable, scalable, and extensible, while utilizing cloud and edge smart resource management and distributed processing. S2CE does not need to be linked to a single cloud provider or DSPE. It has been inspired by the various big data projects in the Apache ecosystem.

The author team has already started the design and development of the needed components, by bootstrapping on functionalities of Apache SAMOA. S2CE will be developed and continuously refined, irrespective of evolving popularities of existing and future DSPEs and cloud infrastructures.

Moreover, S2CE aims to bridge the gap between industry-selected solutions and research projects by offering innovative approaches to stream mining through:

- (1) The ability to optimize the use of available cloud and edge resources by deploying stream mining tasks using existing DSPE clusters preferred by individual organizations. S2CE can go beyond current approaches by automatically tuning the platform, the available DSPEs and cloud (public, private, or hybrid) infrastructure, breaking down the streaming processing task at hand into subcomponents for efficient execution and minimization of computing resources by offloading essential preprocessing to the edge. Importantly, this will let the research community to contribute to the unified ML library of S2CE, rather than multiple native libraries of different DSPEs. Moreover, the platform will include algorithms estimating the performance and DSPE settings suitable for ML.
- (2) Extensive set of high-impact tools on the ultimate results of ML process, including: (i) stream sampling methods to control the data volume used to update ML models in order to prevent performance issues; (ii) synthetic stream generation methods to preserve privacy and confidentiality of true data, while unlocking the value of otherwise frequently not used or available data streams; (iii) a visualization module revealing inference rules present in ML models, making them transparent and promoting business use of stream mining via increased understanding of the performance of models.
- (3) The adoption of existing industry standards. In particular, the platform can provide integration components with popular

systems such as Kafka, making it possible to create efficient and high throughput data pipelines using *S2CE*. The *S2CE* should be provided together with thoroughly designed integration patterns for using *S2CE* with third-party modules such as data storage platforms (e.g., Apache HBase), data ingestion (e.g., Apache Flume), and DSPs (e.g., Apache Storm, Apache Flink). Finally, *S2CE* can be coupled with extensive monitoring and management abilities, easing *S2CE* adoption across companies.

In conclusion, we envision *S2CE* to become the go-to, one-stop-shop platform for mining of big data streams over cloud and edge, and adopted by big data practitioners for its easy pipeline integration and management, and extended by machine learning experts.

REFERENCES

- [1] Jason W Anderson, KE Kennedy, Linh B Ngo, Andre Luckow, and Amy W Apon. 2014. Synthetic Data Generation for the Internet of Things. In *IEEE International Conference on Big Data (Big Data)*. IEEE, 171–176.
- [2] VP Anuradha and D Sumathi. 2014. A Survey on Resource Allocation Strategies in Cloud Computing. In *International Conference on Information Communication and Embedded Systems (ICICES)*. IEEE, 1–7.
- [3] *Apache Apex* 2019. <https://apex.apache.org/>.
- [4] *Apache Beam* 2019. <https://beam.apache.org/>.
- [5] *Apache Edgent* 2019. <https://edgent.apache.org/>.
- [6] *Apache Flink* 2019. <https://flink.apache.org/>.
- [7] *Apache Flume* 2019. <https://flume.apache.org/>.
- [8] *Apache Kafka* 2019. <https://kafka.apache.org/>.
- [9] *Apache Samoa* 2019. <https://samoa.incubator.apache.org/>.
- [10] *Apache Samza* 2019. <https://samza.apache.org/>.
- [11] *Apache Spark* 2019. <https://spark.apache.org/>.
- [12] *Apache Storm* 2019. <https://storm.apache.org/>.
- [13] *AWS Kinesis* 2019. <https://aws.amazon.com/kinesis/>.
- [14] Shivnath Babu and Herodotos Herodotou. 2013. Massively Parallel Databases and MapReduce Systems. *Foundations and Trends® in Databases* 5, 1 (2013), 1–104.
- [15] Manuel Baena-Garcia, José del Campo-Ávila, Raúl Fidalgo, Albert Bifet, R Gavalda, and R Morales-Bueno. 2006. Early Drift Detection Method. In *Fourth International Workshop on Knowledge Discovery from Data Streams*. 77–86.
- [16] Zhendong Bei, Zhibin Yu, Huiling Zhang, Wen Xiong, Chengzhong Xu, Lieven Eeckhout, and Shengzhong Feng. 2016. RFHOC: A Random-Forest Approach to Auto-Tuning Hadoop's Configuration. *IEEE Transactions on Parallel and Distributed Systems (TPDS)* 27, 5 (2016), 1470–1483.
- [17] Anne Benoit, Alexandru Dobrila, Jean-Marc Nicod, and Laurent Philippe. 2013. Scheduling Linear Chain Streaming Applications on Heterogeneous Systems with Failures. *Future Generation Computer Systems* 29, 5 (2013), 1140–1151.
- [18] Morgan Benton et al. 2017. Quality in Chatbots and Intelligent Conversational Agents. *Software Quality Professional Magazine* 19, 3 (2017).
- [19] Albert Bifet, Eibe Frank, Geoffrey Holmes, and Bernhard Pfahringer. 2010. Accurate Ensembles for Data Streams: Combining Restricted Hoeffding Trees using Stacking. In *Proceedings of 2nd Asian Conference on Machine Learning*. 225–240.
- [20] Albert Bifet, Geoff Holmes, Richard Kirkby, and Bernhard Pfahringer. 2010. Moa: Massive online analysis. *Journal of Machine Learning Research* 11, May (2010), 1601–1604.
- [21] Nil Goksel Canbek and Mehmet Emin Mutlu. 2016. On the Track of Artificial Intelligence: Learning with Intelligent Personal Assistants. *Journal of Human Sciences* 13, 1 (2016), 592–601.
- [22] Bin Cheng, Apostolos Papageorgiou, and Martin Bauer. 2016. Geelytics: Enabling on-demand Edge Analytics over Scoped Data Sources. In *IEEE International Congress on Big Data (BigData Congress)*. IEEE, 101–108.
- [23] Peter Christen and Agus Pudjijono. 2009. Accurate Synthetic Generation of Realistic Personal Information. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 507–514.
- [24] Mehmet Hazar Cintuglu, Osama A Mohammed, Kemal Akkaya, and A Selcuk Uluagac. 2016. A Survey on Smart Grid Cyber-physical System Testbeds. *IEEE Communications Surveys & Tutorials* 19, 1 (2016), 446–464.
- [25] Cloud Customer Architecture for Big Data and Analytics V2.0 2019. <https://www.omg.org/cloud/deliverables/CSCC-Cloud-Customer-Architecture-for-Big-Data-and-Analytics.pdf>.
- [26] Paulo Cortez and Mark J Embrechts. 2013. Using Sensitivity Analysis and Visualization Techniques to Open Black Box Data Mining Models. *Information Sciences* 225 (2013), 1–17.
- [27] Ioana Ada Cosma. 2009. *Dimension Reduction of Streaming Data via Random Projections*. Ph.D. Dissertation. Oxford University, UK.
- [28] Rory Coulter and Lei Pan. 2018. Intelligent Agents Defending for an IoT World: A Review. *Computers & Security* 73 (2018), 439–458.
- [29] *DataGenerator* 2019. <https://finraos.github.io/DataGenerator/>.
- [30] Marcos Dias de Assuncao, Alexandre da Silva Veith, and Rajkumar Buyya. 2018. Distributed Data Stream Processing and Edge Computing: A Survey on Resource Elasticity and Future Directions. *Journal of Network and Computer Applications* 103 (2018), 1–17.
- [31] Armen Der Kiureghian and Ove Ditlevsen. 2009. Aleatory or Epistemic? Does it Matter? *Structural Safety* 31, 2 (2009), 105–112.
- [32] Gregory Ditzler, Gail Rosen, and Robi Polikar. 2014. Domain Adaptation Bounds for Multiple Expert Systems under Concept Drift. In *2014 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 595–601.
- [33] *Docker: Enterprise Container Platform* 2019. <https://www.docker.com/>.
- [34] John Ehrlinger. 2015. ggRandomForests: Visually Exploring a Random Forest for Regression. *arXiv preprint arXiv:1501.07196* (2015).
- [35] Ayman Gabarin. Overcoming Cloud Complexity 2019. <https://www.comparethecloud.net/articles/overcoming-cloud-complexity>.
- [36] Joao Gama, Pedro Medas, Gladys Castillo, and Pedro Rodrigues. 2004. Learning with Drift Detection. In *Brazilian Symposium on Artificial Intelligence*. Springer, 286–295.
- [37] João Gama, André Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. 2014. A Survey on Concept Drift Adaptation. *ACM Computing Surveys (CSUR)* 46, 4 (2014), 1–44.
- [38] Javad Ghaderi, Sanjay Shakkottai, and Rayadurgam Srikant. 2015. Scheduling Storms and Streams in the Cloud. *ACM SIGMETRICS Performance Evaluation Review* 43, 1 (2015), 439–440.
- [39] *Google Cloud Dataflow* 2019. <https://cloud.google.com/dataflow/>.
- [40] Lawrence E Hecht. Vendors Compete for Users of Stream Processing Technologies 2019. <https://thenewstack.io/vendors-compete-for-users-of-stream-processing-technologies/>.
- [41] Jessi Hempel. This Is the Smartest Thing Facebook Ever Did 2016. <https://backchannel.com/this-is-the-smartest-thing-facebook-ever-did-e25404b79c77>.
- [42] Herodotos Herodotou. 2017. Business Intelligence and Analytics: Big Systems for Big Data. In *Analytics, Innovation, and Excellence-Driven Enterprise Sustainability*, Elias G. Carayannis and Stavros Sindakis (Eds.). Palgrave Macmillan US, 7–49.
- [43] Herodotos Herodotou, Harold Lim, Gang Luo, Nedyalko Borisov, Liang Dong, Fatma Bilgen Cetin, and Shivnath Babu. 2011. Starfish: A Self-tuning System for Big Data Analytics. In *5th Biennial Conference on Innovative Data Systems Research (CIDR)*. 261–272.
- [44] S. Hoque, M. S. d. Brito, A. Willner, O. Keil, and T. Magedanz. 2017. Towards Container Orchestration in Fog Computing Infrastructures. In *IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)*, Vol. 2. 294–299.
- [45] Yun Chao Hu, Milan Patel, Dario Sabella, Nurit Sprecher, and Valerie Young. 2015. *Mobile Edge Computing - A Key Technology Towards 5G*. White paper 11. ETSI. 1–16 pages.
- [46] *IBM Streaming Analytics for IBM Cloud* 2019. <https://www.ibm.com/cloud/streaming-analytics>.
- [47] *IBM Streams* 2019. <https://ibmstreams.github.io/>.
- [48] Sajid Iqbal, Wasif Altaf, Muhammad Aslam, Waqar Mahmood, and Muhammad Usman Ghani Khan. 2016. Application of Intelligent Agents in Health-care. *Artificial Intelligence Review* 46, 1 (2016), 83–112.
- [49] *Jubatus: Distributed Online Machine Learning Framework* 2019. <http://jubatus.us/en/>.
- [50] Faria Kalim, Thomas Cooper, Huijun Wu, Yao Li, Ning Wang, Neng Lu, Maosong Fu, Xiaoyao Qian, Hao Luo, Da Cheng, et al. 2019. Caladrius: A Performance Modelling Service for Distributed Stream Processing Systems. In *35th IEEE International Conference on Data Engineering (ICDE)*. IEEE, 1886–1897.
- [51] Chandrika Kamath, I Gorton, and DK Gracio. 2013. Dimension Reduction for Streaming Data. In *Data-Intensive Computing: Architectures, Algorithms, and Applications*. Cambridge University Press, 124–156.
- [52] Jeyhun Karimov, Tilmann Rabl, Asterios Katsifodimos, Roman Samarev, Henri Heiskanen, and Volker Markl. 2018. Benchmarking Distributed Stream Data Processing Systems. In *IEEE 34th International Conference on Data Engineering (ICDE)*. 1507–1518.
- [53] *Kubernetes: Production-Grade Container Orchestration* 2019. <https://kubernetes.io/>.
- [54] Sanjeev Kulkarni, Nikunj Bhagat, Maosong Fu, Vikas Kedigehalli, Christopher Kellogg, Sailesh Mittal, Jignesh M Patel, Karthik Ramasamy, and Siddharth Taneja. 2015. Twitter Heron: Stream Processing at Scale. In *Proc. of the 2015 ACM SIGMOD Intl. Conf. on Management of Data*. ACM, 239–250.
- [55] David Linthicum. How to Deal with Cloud Complexity 2019. <https://www.infoworld.com/article/3409089/how-to-deal-with-cloud-complexity.html>.
- [56] Shanhong Liu. Size of the Cloud Computing and Hosting Market Worldwide 2018. <https://www.statista.com/statistics/500541/worldwide-hosting-and-cloud-computing-market/>.
- [57] E. Lughofer, E. Weigl, W. Heidl, C. Eitzinger, and T. Radauer. 2015. Drift Detection in Data Stream Classification without Fully Labelled Instances. In *IEEE International Conference on Evolving and Adaptive Intelligent Systems (EAIS)*. IEEE, 1–8.

- [58] James McInerney, Rajesh Ranganath, and David Blei. 2015. The Population Posterior and Bayesian Modeling on Streams. In *Advances in Neural Information Processing Systems*. 1153–1161.
- [59] Microsoft Azure. 2019. <https://azure.microsoft.com/en-us/>.
- [60] Lina Ni, Jinquan Zhang, Changjun Jiang, Chungang Yan, and Kan Yu. 2017. Resource Allocation Strategy in Fog Computing based on Priced Timed Petri Nets. *IEEE Internet of Things Journal* 4, 5 (2017), 1216–1228.
- [61] Justice Opara-Martins, Reza Sahandi, and Feng Tian. 2016. Critical analysis of vendor lock-in and its impact on cloud computing migration: a business perspective. *Journal of Cloud Computing* 5, 4 (2016).
- [62] Oracle Table Access for Hadoop and Spark (OTA4H) 2018. <https://docs.oracle.com/bigdata/bda45/BIGUG/ota4h.htm#BIGUG76783>.
- [63] René Peinl, Florian Holzschuher, and Florian Pfitzer. 2016. Docker Cluster Management for the Cloud - Survey Results and own Solution. *Journal of Grid Computing* 14, 2 (2016), 265–282.
- [64] Darren Perucci. 2016. Cloud Computing + Data Analytics = Instant Business Intelligence 2017. <https://dzone.com/articles/cloud-computing-data-analytics-instant-business-in>.
- [65] Power BI 2019. <https://powerbi.microsoft.com/en-us/>.
- [66] Chandan Prakash. Spark Streaming vs Flink vs Storm vs Kafka Streams vs Samza : Choose Your Stream Processing Framework 2018. <https://medium.com/@chandanbaranwal/spark-streaming-vs-flink-vs-storm-vs-kafka-streams-vs-samza-choose-your-stream-processing-91ea3f04675b>.
- [67] Tilmann Rabl, Michael Frank, Hatem Mousselly Sergieh, and Harald Kosch. 2010. A Data Generator for Cloud-scale Benchmarking. In *Technology Conference on Performance Evaluation and Benchmarking*. Springer, 41–56.
- [68] Marek Rychly et al. 2014. Scheduling Decisions in Stream Processing on Heterogeneous Clusters. In *Eighth International Conference on Complex, Intelligent and Software Intensive Systems (CISIS)*. IEEE, 614–619.
- [69] Hooman Peiro Sajjad, Ken Danniswara, Ahmad Al-Shishtawy, and Vladimir Vlassov. 2016. Spanedge: Towards Unifying Stream Processing over Central and Near-the-edge Data Centers. In *2016 IEEE/ACM Symposium on Edge Computing (SEC)*. IEEE, 168–178.
- [70] Mark Samuels. How to Choose your Cloud Provider: AWS, Google or Microsoft? 2018. <https://www.zdnet.com/article/how-to-choose-your-cloud-provider-aws-google-or-microsoft/>.
- [71] SAS Cloud Analytics 2019. https://www.sas.com/en_us/solutions/cloud-analytics.html.
- [72] D. SchÄdfer, J. Edinger, S. VanSyckel, J. M. Paluska, and C. Becker. 2016. Tasklets: Overcoming Heterogeneity in Distributed Computing Systems. In *IEEE 36th International Conference on Distributed Computing Systems Workshops (ICDCSW)*. 156–161.
- [73] Jim Scott. How Orchestration, Edge Computing, and Serverless Computing Impact Your Cloud Strategy 2018. <https://mapr.com/blog/how-orchestration-edge-computing-and-serverless-computing-impact-your-cloud-strategy/>.
- [74] Arun Shankar. Nutanix Revamps Platform for Multi-cloud Orchestration and IoT Edge Management 2018. <http://www.biznesstransform.com/nutanix-revamps-platform-for-multi-cloud-orchestration-and-iot-edge-management/>.
- [75] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu. 2016. Edge Computing: Vision and Challenges. *IEEE Internet of Things Journal* 3, 5 (Oct 2016), 637–646.
- [76] Bhupendra Singh and Ankit Gupta. 2015. Recent Trends in Intelligent Transportation Systems: A Review. *Journal of Transport Literature* 9, 2 (2015), 30–34.
- [77] Sukhpal Singh and Indrveer Chana. 2016. Cloud Resource Provisioning: Survey, Status and Future Research Directions. *Knowledge and Information Systems* 49, 3 (2016), 1005–1069.
- [78] Agnieszka Sitko and Przemyslaw Biecek. 2017. The Merging Path Plot: Adaptive Fusing of k-groups with Likelihood-based Model Selection. *arXiv preprint arXiv:1709.04412* (2017).
- [79] O. Skarlat, M. Nardelli, S. Schulte, and S. Dustdar. 2017. Towards QoS-Aware Fog Service Placement. In *IEEE 1st International Conference on Fog and Edge Computing (ICFEC)*. 89–96.
- [80] Vinicius MA Souza, Diego F Silva, Gustavo EAPA Batista, and João Gama. 2015. Classification of Evolving Data Streams with Infinitely Delayed Labels. In *IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*. IEEE, 214–219.
- [81] Statista Research Department 2019. <https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/>.
- [82] streamDM: Data Mining for Spark Streaming 2019. <http://huawei-noah.github.io/streamDM/>.
- [83] Dawei Sun, Guangyan Zhang, Songlin Yang, Weimin Zheng, Samee U Khan, and Keqin Li. 2015. Re-Stream: Real-time and Energy-efficient Resource Scheduling in Big Data Stream Computing Environments. *Information Sciences* 319 (2015), 92–112.
- [84] Ming Sun, Yun-Nung Chen, Zhenhao Hua, Yulian Tamres-Rudnicki, Arnab Dash, and Alexander Rudnicki. 2016. AppDialogue: Multi-app Dialogues for Intelligent Assistants. In *Tenth International Conference on Language Resources and Evaluation (LREC'16)*. 3127–3132.
- [85] Tableau 2019. <https://www.tableau.com/>.
- [86] B. Varghese, N. Wang, S. Barbhuiya, P. Kilpatrick, and D. S. Nikolopoulos. 2016. Challenges and Opportunities in Edge Computing. In *IEEE International Conference on Smart Cloud (SmartCloud)*. 20–26.
- [87] Shivaram Venkataraman, Zongheng Yang, Michael J Franklin, Benjamin Recht, and Ion Stoica. 2016. Ernest: Efficient Performance Prediction for Large-Scale Advanced Analytics. In *13th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*. USENIX Association, 363–378.
- [88] Vowpal Wabbit 2019. https://github.com/VowpalWabbit/vowpal_wabbit.
- [89] Chunkai Wang, Xiaofeng Meng, Qi Guo, Zujian Weng, and Chen Yang. 2017. Automating Characterization Deployment in Distributed Data Stream Management Systems. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 29, 12 (2017), 2669–2681.
- [90] Guolu Wang, Jungang Xu, and Ben He. 2016. A Novel Method for Tuning Configuration Parameters of Spark based on Machine Learning. In *18th International Conference on High Performance Computing and Communications (HPCC)*. IEEE, 586–593.
- [91] Soeren H Welling, Hanne HF Refsgaard, Per B Brockhoff, and Line H Clemmensen. 2016. Forest Floor Visualizations of Random Forests. *arXiv preprint arXiv:1605.09196* (2016).
- [92] Hadley Wickham, Dianne Cook, and Heike Hofmann. 2015. Visualizing Statistical Models: Removing the Blindfold. *Statistical Analysis and Data Mining: The ASA Data Science Journal* 8, 4 (2015), 203–225.
- [93] Jielong Xu, Zhenhua Chen, Jian Tang, and Sen Su. 2014. T-Storm: Traffic-Aware Online Scheduling in Storm. In *International Conference on Distributed Computing Systems (ICDCS)*. IEEE Computer Society, 535–544.
- [94] J Yang and JF Coughlin. 2014. In-vehicle Technology for Self-driving Cars: Advantages and Challenges for Aging Drivers. *International Journal of Automotive Technology* 15, 2 (2014), 333–340.
- [95] Daniel (Yue) Zhang, Tahmid Rashid, Xukun Li, Nathan Vance, and Dong Wang. 2019. HeteroEdge: Taming the Heterogeneity of Edge Computing System in Social Sensing. In *International Conference on Internet of Things Design and Implementation*. 37–48.