AI/ML Service Enablers & Model Maintenance for Beyond 5G Networks

Konstantinos Samdanis
Lenovo
Munich, Germany
ksamdanis@lenovo.com

Aiman Nait Abbou
Aalto University
Espoo, Finland
aiman.naitabbou@aalto.fi

JaeSeung Song
Sejong University
Seoul, South Korea
jssong@sejong.ac.kr

Tarik Taleb
Oulu & Sejong University
Oulu, Finland
tarik.taleb@oulu.fi

***Abstract*—Artificial Intelligence and Machine Learning (AI/ML) can transform mobile communications, enable new applications and services, and pave the way beyond 5G. The adoption of AI/ML may also advance network optimizations and service life-cycle management. This article provides an overview of AI/ML analytics exploring the main concepts in relation to network automation and service-based architectures. Furthermore, it sheds light on AI/ML analytics service enablers by considering a system-level approach elaborating on the key technologies related to service discovery, request, control, and reporting of AI/ML analytics. Finally, it provides an analysis for maintaining the services of AI/ML analytics up-to-date, by considering modifications of the AI/ML model in order to detect, interpret, and compensate for potential performance drifts.**

***Index Terms*—Analytics, AI/ML, B5G**

I. Introduction

Beyond 5G (B5G)[1] is expected to facilitate new services by leveraging the benefits of automation in network and service provisioning. Such services may include advanced network troubleshooting, big data service, robotics, biosensors, mixed-reality experience, auto-driving/piloting, and self-sustainable work-sites [1]. A key ingredient for the success of automation is analytics, as it can turn data into knowledge to improve network and business performance. One common way to realize analytics is by adopting Artificial Intelligence or Machine Learning (AI/ML), which can provide advanced solutions by employing the capability to learn without being explicitly programmed.

AI/ML can bring value for mobile service providers during network planning and operations, by allowing efficient and rapid network optimizations. It can also bring new revenue streams by combining big data and networking to facilitate differentiated customer experience with controlled assurance levels. AI/ML is the foundation of creating innovative services towards vertical segments, e.g., by providing feedback to vehicular applications that allow a proactive switch of the level of automation for self-driving cars, and service sustainability for future car locations. AI/ML services can be utilized in different network segments of 5G, including:

- *Network orchestration,* with the objective to improve network resource management, assure network performance, configuration management, and efficiently analyze failures. The use of AI/ML aims to assist long-term optimizations;
- *Radio* that relies on real-time data to analyze user access patterns and radio conditions, which are highly dynamic in nature. The goal is to optimize the operations of base stations, e.g., scheduling, interference control, etc.;
- *Core network*, to provide control plane analytics for particular users or flows with the objective to analyze or predict users' service experience and behavior, e.g., in terms of communication patterns, mobility, or security;
- *Application*, focusing on performance optimizations, e.g., re-configuring a video codec, policy re-negotiation, and synchronization for assuring service sustainability and Quality or Experience (QoE).

Unlike [2], [3], which concentrates on the learning, algorithms, and application of AI/ML in Self-Organizing Networks (SON) and wireless communications, this paper sheds light on the practice and key enabling technologies of AI/ML. In particular, it explores the service enablers that allow a consumer of AI/ML to discover, request, and control the analytics results. A consumer of AI/ML analytics can be a logical or physical entity of the network, an automation or assurance function, a service optimization tool, a human operator, or an application. In addition, it elaborates on the steps taken by a producer of AI/ML analytics when preparing and determining the requested results. The service maintenance of analytics, including model selection, training, validation, etc., is also considered by taking into account different types of AI/ML algorithms and elaborating on the idea of *concept drift*. In addition, this paper provides a qualitative study that compares among different approaches of concept drift that are linked with distinct AI/ML models with respect to drift detection, interpretation, and adaptation.

The remainder of this paper is organized as follows. Section II provides an overview of the concepts of B5G analytics. Section III reviews the operations of AI/ML analytics, while Section IV elaborates on key enablers. Section V presents the procedures of AI/ML model maintenance, and finally, Section VI concludes the paper.

[1]3GPP Rel.18 and beyond is called B5G (aka 5G-Advanced) based on: 3GPP, *"Advanced plans for 5G"*, Jul. 2021 (https://www.3gpp.org/news-events/2210-advanced$_5g$).

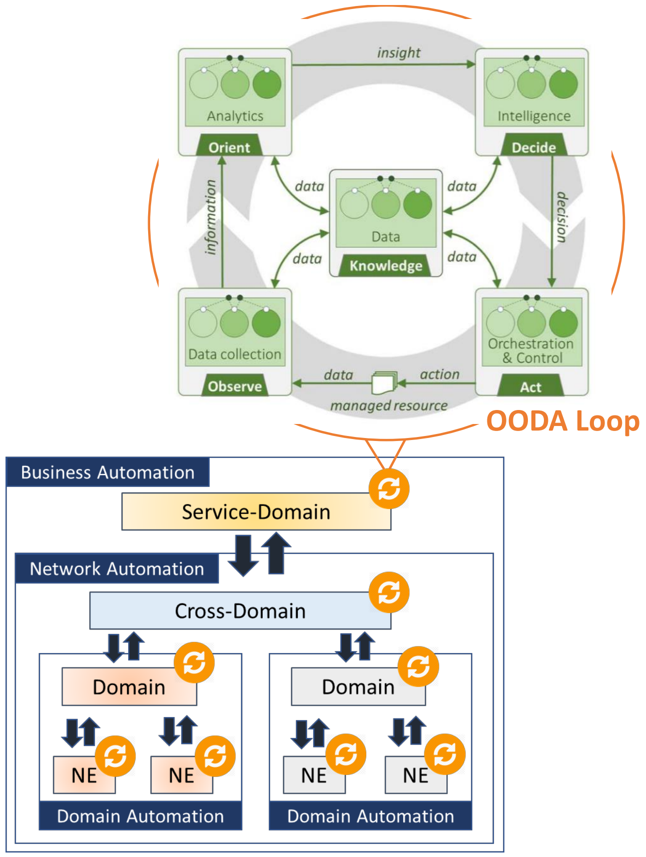


Fig. 1: Analytics scope and closed loops considering the OODA Loop of ETSI ZSM 002.

II. Overview of Analytics Concepts in B5G

Network analytics can be utilized in different parts of the service life cycle management, and their insight results can be made available across distinct network operations planes. Such capabilities enable Network Functions (NFs), Management Services (MnSs), and applications to flexibly gain and combine network intelligence.

A. Analytics and Closed-Loops

AI/ML analytics provide an insight based on the observed data. In other words, AI/ML analyzes and assesses a situation without providing a decision. Such an insight may be used to build knowledge and, at the same time, feed a decision-making entity that intelligently guides orchestration and control to act on a managed resource, user, or application. AI/ML analytics can be a part of a closed-loop, like the Observe, Orient, Decide, and Act (OODA) loop (ETSI ZSM 002) shown in Fig.1, which has the objective of assuring target goals.

Closed loops, and hence AI/ML analytics, can be applied to different network scopes, as illustrated in Fig.1, including:

- *Network elements (NEs)* that focus on mechanisms executed locally in NFs or base stations, e.g., SON functions;
- *Network domains* involving mechanisms executed in the:
  1) Management plane, i.e., Management Data Analytics (MDA) (3GPP TR 28.809, TS 28.104) in the element manager of the radio or 5G core domain;
  2) Control plane, i.e., the Network Data Analytics Function (NWDAF) (3GPP TS 23.288) responsible for providing analytics in the 5G core network;
- *Cross-domains*, which allow coordination, e.g., resource management, across the radio and 5G core domains in the management plane;
- *Communication services* that enable the preparation of network resources based on service requirements and interactions with the consumer for resource adjustments.

AI/ML analytics related to different network scopes may adopt distinct algorithms or models, which are executed independently. The notion of AI/ML analytics and closed loops can be applied to different life-cycle phases of a communication service, including preparation, commissioning, operations, and decommissioning. Preparation focuses on the design, pre-planning, and negotiation of the service requirements. Commissioning converts the communication service to network requirements before it launches a service assurance control loop to guard the boundaries of the target goals. Operations allow run-time assurance by adjusting the allocated resources and decommissioning de-activates them once they are no longer needed.

B. Composing AI/ML Analytic Services

AI/ML analytics services may consist of a number of logical components, which are combined to form a service chain that performs AI/ML analytics. A typical AI/ML analytics service chain, according to ITU-T FG-ML5G, may consist of the following logical components:

- *Source* that provides raw data related to performance measurements, Key Performance Indicators (KPIs), UE measurements, configuration, and alarms to feed the *AI/ML Model*. In case of an Internet of Things (IoT) platform (e.g., oneM2M service layer platform) that provides an interworking function with the 3GPP system, it is possible to provide both sensor and service-related data;
- *Collector*, which coordinates the collection of data from various sources;
- *Pre-processor* responsible for preparing the data to fit the *AI/ML Model* by performing different data processing including data cleansing and formatting;
- *AI/ML Model, which* represents an AI/ML logic or algorithm used for inference, i.e., for producing analytics output results based on “live” data input;
- *Policy* that defines rules for forwarding the output of the *AI/ML Model* towards the corresponding *Sinks*, while controlling the regularity and conditions upon which input data is collected;
- *Distributor* that is in charge of identifying the *Sinks* and distributing/forwarding the *Policy* to the corresponding output of the *AI/ML Model*;

- *Sink*, which is the consumer or the target of the *Distributor*.

An AI/ML service chain deployment may span multiple technology domains, i.e., radio, transport, and 5G core, which may belong to different administrative entities. The life-cycle management of an AI/ML service chain relies on the AI/ML service orchestrator, which prepares, configures, re-locates, and optimizes, i.e., scales up/down, the logical components of a corresponding AI/ML service. Variations in the observation data, e.g., due to user mobility, may cause changes in the deployment of an ongoing AI/ML service chain with the orchestrator being responsible for performing the relevant optimizations. AI/ML model maintenance operations, e.g., model re-training, selection, transfer learning, etc., may be optionally handled or triggered by the AI/ML service orchestrator.

C. AI/ML Analytics in 5G Micro-service Architectures

The 5G architecture paradigm adopts micro-services with the introduction of service-based architecture in the 5G core (3GPP TS 23.501, TS 23.502) and network management (3GPP TS 28.533) that enables inter-operation and data exchange among 5G core NFs and MnSs. Service-based architectures rely on representational state transfer interfaces, also called RESTful interfaces, (IETF RFC 7231) to facilitate service acquisition, modification, and termination. RESTful interfaces also enable access for 3rd party applications and vertical segments by leveraging the exposure capability of 5G to assure security, service mapping, and data abstraction.

AI/ML analytics can exploit the benefits of service-based architecture to enrich the quality of analytics, by combining the following observation data from different technology domains:

- *5G core*: control plane or user-centric data, e.g., user mobility, communication patterns, user security risks, etc.;
- *Radio*: near real-time or real-time data, e.g., interference, signal strength, pilot congestion, etc.;
- *Network management*: performance measurements, (e.g., throughput), KPIs, (e.g., end-to-end delay), fault management (e.g., alarms), and configuration management;.
- *Computing and virtualization*: Central Processing Unit (CPU) load, storage, and memory;
- *Application*: QoE, service sustainability, security and privacy, operating environment, etc.

AI/ML analytics results can be also provided across different domains via the means of service-based integration fabric (ETSI ZSM 002). Such integration fabric enables the discovery, selection, and invocation of cross-domain analytics.

III. AI/ML Analytics Operations

AI/ML analytics services follow the producer-consumer paradigm. An AI/ML analytics producer is responsible for providing analytics reporting for interested AI/ML analytics consumers (see Fig. 2). An AI/ML analytics producer relies on AI/ML inference to supply analytics results based on input data from various sources, i.e., NFs, MnS, and application data as discussed in Section II.C. An AI/ML inference may adopt a single or a variety of AI, ML, Reinforcement Learning (RL), or rule-based models, which are prepared by the AI/ML Analytics Model Orchestrator.

According to 3GPP TS 28.104 and TS 23.288, AI/ML analytics operations are related to AI/ML online service provisioning and model life-cycle management. AI/ML online service provisioning allows an AI/ML analytics consumer to request, negotiate, and obtain analytics results. It relies on data collected by the AI/ML analytics producer, which is used for AI/ML inference, to determine such analytic results. The AI/ML model life-cycle management, on the other hand, deals with the AI/ML model maintenance, i.e., ensuring that the AI/ML service is up-to-date by analyzing consumer feedback and data from various regular input sources.

An AI/ML analytics producer may support a set of analytics service types, which are identified using predefined names or identifiers, e.g., $UE_{mobility}$, NF_{load}, or ID_{number}. An AI/ML service is requested using an analytics service type name. Such AI/ML services may operate either within a strict geographical scope, e.g., a specific number of tracking areas, or within a loose scope, e.g., a domain. A mobile operator network may offer a number of AI/ML analytics services for a particular operating scope, which can be combined together in a single network entity, i.e., in a NF or MnS, and can inter-work with each other inside the network to provide global analytics. An AI/ML analytics consumer may either subscribe to an on-going AI/ML analytics service or request the AI/ML analytics producer to set it up, by instantiating the corresponding measurement jobs to collect the required observation data, before producing analytics results. In other words, there are two modes of operation for AI/ML analytic service: synchronous and asynchronous.

The synchronous mode assumes that the AI/ML analytics producer is continuously providing analytics results using specific AI/ML models and regularly consuming input data from predetermined sources. The AI/ML analytics producer feeds the AI/ML inference with input data from a specific set of network objects, e.g., NFs or MnSs, or with data from network objects that reside within a certain geographical area, e.g., a tracking area. Hence, analytics results are constrained in terms of the type and location of input data sources, and due to the AI/ML model in use. In addition, the regularity schedule of input data limits the observation capability type, e.g., non-real-time input data cannot be used to provide real-time analytics results. On the other hand, analytics results are always ready and available immediately for interested consumers.

The asynchronous mode allows an AI/ML consumer to place an analytics service request that selects an AI/ML model and initiates the process of collecting input data. This mode provides a degree of customization, i.e., offering the capability to pick the input data sources with the desired KPIs, location, and observing capability type, i.e., real-time or non-real-time. A consumer can hence select an analytics service and a related AI/ML model from a list offered by the AI/ML analytics producer. The AI/ML analytics producer can then instantiate

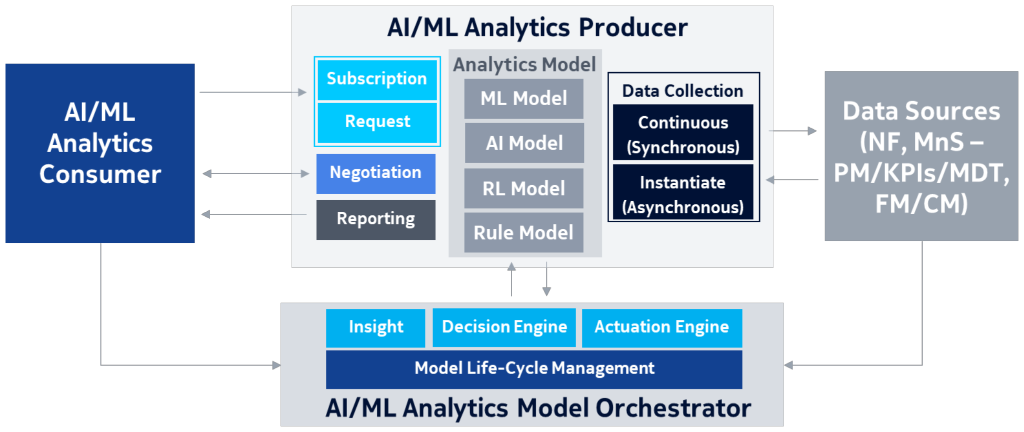


Fig. 2: AI/ML analytics service enablers and operation processes.

an AI/ML analytics service and a corresponding control to allow the consumer to modify the input data, time schedule, and reporting filters and mechanisms as documented in Section IV.B, at any time. However, the collection of data and the AI/ML inference may incur a delay in delivering analytics results for the corresponding consumer.

In synchronous operation, a subscribing consumer can only fine-tune minor time scheduling and filtering parameters. e.g., to receive an analytics report when the network load surpasses a threshold. In contrast, asynchronous operations introduce customization, but also include a service delay that may need to be agreed upon, before the service is instantiated. Hence, a negotiation phase is needed, especially for an asynchronous operation to ensure that the characteristics of the analytics service and service time delay are within the expected bounds.

Once an AI/ML analytic service is setup, the AI/ML analytic producer prepares the analytics results for the consumer. The AI/ML analytics results may include a combination of the following:

- *Numeric results*, e.g., average, Cumulative Distribution Functions (CDF), standards deviation, etc., or a range, e.g., minimum - maximum;
- *Recommendation options* focusing on: (i) managed objects, e.g., NFs or base stations, (ii) mechanisms, e.g., handover mechanisms, (iii) states, e.g., network state or SON state, (iv) configuration attributes, e.g., for a tilt of a base station, and (v) policy, e.g., handover policy;
- *Root cause analysis* results, e.g., alarm prediction or network state correlation.

Each of these outputs are statistical results based on historical data or predictions for a future time window and are accompanied by a confidence interval. An analytics output or report may contain one or more result types at the same time, e.g., a numeric result, which may involve the average radio load prediction, and a set of recommended handover mechanisms (e.g., too early or too late, dual connectivity, Random Access Channel (RACH)-less to improve latency) eligible for selection.

Once regular input data and consumer feedback indicate that the results are not as expected, e.g., predictions deviate significantly from actual measurements, an AI/ML model update is needed. Network data and non-3GPP, e.g., IoT, data (oneM2M TR-0068, AI-enablement to oneM2M IoT platform), can be collected for the purpose of training. The oneM2M IoT platform allows data collected from a large amount of IoT devices to be managed as training data to develop AI/ML models. The AI/ML analytics producer can create more accurate AI/ML models by collecting such training data from IoT platforms to represent specific service domains or groups of devices (e.g., autonomous vehicles). Model updates are carried out by the AI/ML analytics model orchestrator, which manages the model-life cycle and takes care of AI/ML model re-training, re-selection, and other model modifications. The AI/ML analytics model orchestrator relies on the following logical functions:

- *Insight* that collects input from regular data sources and consumers to correlate predictions with actual measurements to provide a root cause analysis related to the performance deviation of an AI/ML model;
- *Decision engine*, which provides intelligence for adjusting the AI/ML model, e.g., by using different training for certain applications, consumers, target objects, etc., share the AI/ML model training or adopt a different type of AI/ML model;
- *Actuation engine*, which is responsible for performing the AI/ML model updates, i.e., model transfer, model exchange, etc.

Model updates for an AI/ML analytics producer may be performed by transferring a new or re-trained AI/ML model using containers, or via a meta-language that describes a new model or the corresponding updates.

IV. Service Enablers of AI/ML Analytics

The process of AI/ML online service provisioning includes three main steps: (i) discover the capabilities of an AI/ML analytics service, (ii) request an AI/ML analytics service and control its reporting attributes, and (iii) obtain or receive an analytics output that contains the desired results, as specified in 3GPP TS 28.104 and TS 23.288.

A. Discovering AI/ML Analytics

Before a consumer can request an AI/ML analytics service, it needs to know that it can fulfil its needs. AI/ML analytics service discovery allows the consumer to explore the service capabilities in terms of the supported analytics service types, AI/ML model, reporting location or objects, scheduling times, reporting modes, and other filtering information. The discovery process can be centralized or hybrid. A centralized repository allows an AI/ML analytics function to register its serving capabilities. A consumer can then place an AI/ML analytics service discovery request in the repository and receive a set of potential AI/ML analytic functions that can fulfill the request. The consumer can then select the most appropriate function based on the supported service capabilities. In a hybrid approach, a consumer can request a centralized entity, e.g., a Domain Name Server (DNS), to obtain some basic information such as the IP address of a set of AI/ML analytics functions. The consumer can then select an AI/ML analytics service and use this basic information to request its detailed service capabilities, which can be obtained directly.

B. Requesting AI/ML Analytics

When requesting an AI/ML analytic service, a consumer specifies the desired reporting control attributes. Such control attributes may include information related to the AI/ML service, scope of operation, time scheduling, filters, and reporting details, as well as the desired result description. A consumer may indicate the following attributes of an AI/ML service:

- *AI/ML service type name*, which indicates the desired service, e.g., $UE_{mobility}$, NF_{load}, or ID_{number}, etc.;
- *AI/ML model type* that is used to compute the analytics, e.g., convolutional neural network, ML, RL, mathematical function, set of rules, etc.;
- *AI/ML model metrics*, which are used to describe the performance of the analytics service, e.g., true positives, false positives, F1 score, etc.;
- *AI/ML model time stamp* that indicates when the model was trained, last modified or put into service.

A consumer may additionally request AI/ML analytics for a particular operational scope by specifying either the geographical location or specific target objects. The geographical location indicates an area of interest, which can be represented by a cell, a set of one or more tracking areas, or by a set of coordinate points. Target objects may involve a UE, a set of UEs, a flow, a slice, or a combination, e.g., once a UE enters an area of interest. The reporting time scheduling may involve:

- *Immediate reporting flag*, which is a prompt reporting trigger that contains the current AI/ML analytics result;
- *Periodic reporting* that indicates the regular time period;
- *Target time duration* related to an AI/ML analytics service, indicating a desired time duration in the past that a statistics report should consider or in the future that a prediction report should be valid for;
- *Feasibility time* that specifies until when a requested AI/ML analytics service is needed by a consumer.

Filter information specify which conditions should be fulfilled before an analytics report is triggered, for instance upon a particular event, e.g., exceeding a load threshold. Furthermore, the reporting details provide information related to the reporting mode or method. In particular, the reporting mode characterizes whether the reporting type should be continuous, i.e., streaming, in batches, i.e., file-based, or obtained from a specified location once a notification is received. The consumer may also specify various attributes that describe the desired result, which may include:

- *Group reporting*, which indicates that an AI/ML analytics service should be processed or aggregated before being reported to the consumer, specifying also the adopted method (e.g., average);
- *Accuracy ratio* that relates to the sampling ratio of the target objects, e.g., 60% of the UEs or 50% of the base stations in a given area of interest.
- *Reporting results type* that indicates the expected analytics output, which can be numeric, recommendation options or root cause analysis results, or a combination of the three.
- *Reporting analytics type*, which characterizes the reporting result types into statistics for historical measurements or predictions for future events.

Finally, the consumer may request the result to be delivered in a different location than its own, i.e., different from the one where the AI/ML analytics service was requested. In this case, the consumer needs to explicitly indicate the notification address for the AI/ML analytics reporting.

C. Reporting AI/ML Analytics

Once an AI/ML analytics service producer prepares an analytics output or report, it exposes it to the consumer or towards a storage entity. The report includes the following attributes:

- *Reporting results* that contain the output data requested by the consumer;
- *Reporting results type*, which indicates whether the report contains: (i) numeric results, (ii) recommendations of optimal parameters or configurations, or (iii) root cause analysis results;
- *Reporting analytics type*, which indicates whether the reporting result types are statistics or predictions;
- *Validity period* that specifies until when a report is useful, e.g., a NF load prediction that is valid for the next 10 minutes;
- *Timestamp of a report* that defines a record related to the report generation time;
- *Confidence degree* of statistics or prediction indicating the accuracy of the reported results;

- *Reporting expiration information*, which indicates the termination of the AI/ML analytics service in terms of timing or the number of remaining reports.

The AI/ML reports can be provided using file-based reporting that suits the transportation of bigger amounts of data; streaming, which offers low latency related to service assurance; or notification-based reporting to efficiently transport small data, e.g., alarms. Alternatively, the AI/ML analytics service producer may provide the results on, e.g., a server, and then notify the consumer to obtain them.

V. Maintenance of AI/ML Analytic Services

AI/ML analytics producers need a continuous service quality assessment based on consumer requirements. In most cases, consumers are not capable of noticing gradual changes in the quality, state, or performance of analytics results. Specifically, due to the dynamic nature of input data and unexpected usage, the imperceptible change in user experience is a common problem for consumers. For instance, a change in the statistical properties of a target variable over time, which the AI/ML model is predicting, may result in inaccurate predictions, which reduces the overall quality of the service. This is known as *concept drift*, which can drive the AI/ML analytics model orchestrator as shown in Fig.2. The main causes of concept drift are as follows:

- *Unseen input data:* AI/ML models can be trained with insufficient amounts of data and, consequently, they may produce inaccurate or erroneous results.
- *Updated input data:* If the system is continuously fed with data from multiple sources as documented in Section II.C, the stability of the system is tied to the consistency of such data.

Maintaining AI/ML analytics is a challenging task for service providers, as the quality of an analytics report is related to the performance of AI/ML models. After the deployment phase, it is assumed that the accuracy of the reporting results is satisfactory. In the inference phase, the post-deployment accuracy test works as a quality control indicator, which assists the AI/ML analytics model orchestrator by periodically assessing AI/ML models. If the quality degrades, the accuracy test triggers an investigation of the drift. Otherwise, the analysis continues to consider incoming data and forecast potential drifts before feeding the AI/ML models with the collected data.

Analytics can be used in different 5G service domains. For example, in the case of a gyro sensor used to measure the rotation speed of an autonomous vehicle, drift occurs due to system temperature, period of use, mathematical error, etc.; this may cause an accident by slowing the rotation speed. AI/ML analytics can detect such a drift (called a concept drift), in which the operation speed of the gyro sensor is attenuated with data input from various sources (e.g., NF, MnS-PM, and IoT). A result report can be created that recommends fast transmission of a command to the drifted gyro sensor to avoid causing an accident. The process of concept drift consists of: (i) the detection, (ii) the interpretation, by finding out where it happened, what triggered it, and the time of its first occurrence, and (iii) the adaptation, i.e., reaction to address the drift. This process is illustrated in Fig. 3.

A. Concept-Drift Detection

After deploying an AI/ML model, the concept drift detection system provides an insight based on the collected data. Concept drift detection can also be seen as a classifier to control the data consumed by AI/ML models. Assuming that the collected data is robust and safe, the detection framework consists of four main components.

The *monitoring system* collects data chunks from multiple sources and organizes it into blocks to facilitate the pre-processing phase. The monitoring system analyzes the problem of defining a window size to distinguish the old data and new data that need to be verified. The choice of such a window impacts the sensitivity of the drift selection algorithm. A smaller window may be effective against a sudden drift, however, detecting a gradual drift may be difficult. The majority of the existing algorithms use a fixed-based window. Unsupervised Multi-scale Slide Windows (UMSW) [4] and Concept-Drift-Aware Federated Averaging (CDA-FedAvg) [5] are two examples that follow this approach. The main advantage is simplicity, but this method suffers when handling distributions with high-density variations between the regions [6]. Other more dynamic methods may determine the window size based on the proportion of the collected data, (e.g., diverse instance weighting ensemble (DiwE) [6]).

The *pre-processing system* is an optional phase, which may be needed if the size of the collected data is significantly large. In fact, the pre-processing system eliminates unnecessary data, so that only the most sensitive data is kept that is capable of causing a concept drift if changed. Many methods are used to parse the data, including dimensionality reduction and data

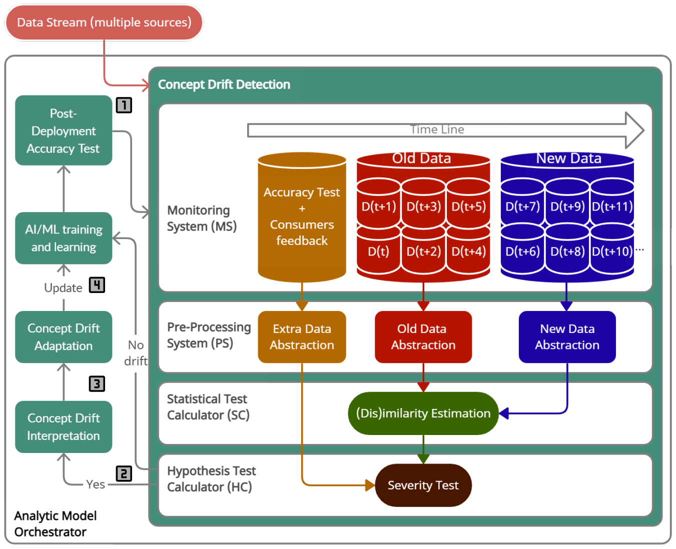

Data Stream (multiple sources)
Concept Drift Detection
Post-Deployment Accuracy Test
1
AI/ML training and learning
Update
4
Concept Drift Adaptation
3
Concept Drift Interpretation
Yes
2
No drift
Analytic Model Orchestrator
Time Line
Monitoring System (MS)
Accuracy Test + Consumers feedback
Old Data
D(t+1)
D(t+3)
D(t+5)
D(t)
D(t+2)
D(t+4)
New Data
D(t+7)
D(t+9)
D(t+11)
D(t+6)
D(t+8)
D(t+10)
...
Pre-Processing System (PS)
Extra Data Abstraction
Old Data Abstraction
New Data Abstraction
Statistical Test Calculator (SC)
(Dis)imilarity Estimation
Hypothesis Test Calculator (HC)
Severity Test

Fig. 3: The concept drift process

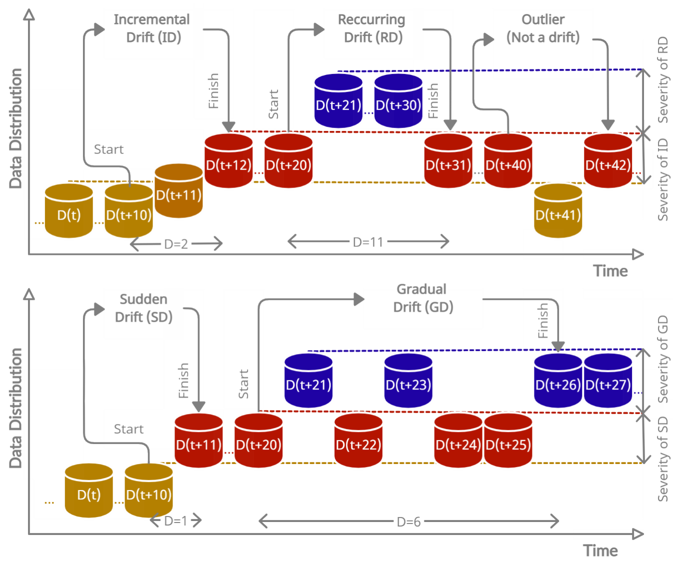


Fig. 4: Concept drift types: An example of the interpretation phase.

normalization [7] and Principal Component Analysis (PCA) [8]. The *statistical test calculator* investigates drift occurrence by comparing the old and new pre-processed data. If a drift is detected, it measures its severity and sends a report to the hypothesis calculator. Many methods can then deploy this phase (e.g., K-means [7] and Kolmogorov–Smirnov (KS) statistics test [9]). The choice may depend on the learning setting, i.e., unsupervised, supervised, [6], or semisupervised [10]. The *hypothesis test calculator* analyzes and questions the accuracy of the report sent by the testing calculator. This phase is important as it makes a final decision regarding whether a drift has occurred, or if it is a false-positive case due to corrupted or insufficient data. Extra data can be collected from the consumers to increase the accuracy of the measurement of the severity of the drift. The majority of the existing algorithms use the significance level approach [9], [11], [12].

B. Concept-Drift Interpretation

Several types of drifts can be detected, including: (i) incremental drift that emerges slowly over time, (ii) sudden drift, which takes place suddenly for a short period of time, (iii) recurring drift, which is a variation of sudden drift that repeats itself over a period of time, and (iv) gradual drift that refers to a new gradual development that dominates the old one over a period of time. Due to the difficulty in manually verifying the collected data, any data set may result in any type, or a combination of types, of drifts. Therefore, there is a need to interpret each drift, which requires the system to comprehend the key elements related to a drift, as shown in Fig. 4. The key elements of a drift are as follows:

- *Criticality (how):* The severity of the concept drift can be assessed by measuring the distance between the previous and the new data. This module may use several methods, including the report sent by the post-deployment accuracy test, to measure the distance. This value will affect the choice of adaptation strategy.
- *Occurrence time (when):* The interpreter introduces an alarm to indicate the exact timestamp when the drift occurred. The time is chosen based on the moment when the two data samples (new and old) were unexpectedly different. This module may also use the accuracy test report to make this estimation.
- *Duration (how long):* This parameter indicates the period of the drift. Measuring the duration of the detected drift can contribute to the confidence of the interpreter, avoid outlier cases, and decrease the ratio of false positive detections.
- *Location (where):* The location where the drift occurred is tied to the mechanism used in the detection phase. Identifying stable regions and distinguishing between drifts can classify the data and allow obsolete or noisy data to be ignored.

C. Concept-Drift Adaptation

The adaptation selection strategy requires high-level understanding and knowledge about data changes. When a drift is detected, the training or AI/ML model adjustment process should be aware. Otherwise, the AI/ML analytics results would continue to produce poor and inaccurate results. The most simple and straightforward approach is re-training, as recommended by Equal Intensity K-Means (EI-kMeans) [7]. This technique may be effective if the post-deployment accuracy test consistently provides poor feedback for the same group of services and the detection mechanism judges the drift to be global. In such case, an AI/ML model should be re-trained with new target variables, then the two models should compete for the main position, and the least accurate model can act as a backup.

Another effective approach against global drift is to first detect the concept drift and then update the model accordingly. This technique may be more effective for certain applications, e.g., in Federated Learning where the size of the locally collected data is manageable. CDA-FedAvg [5] uses long-term memory to keep a record of the data that can be used for locally training the model with a new concept. Re-training is a reactive technique, and it is costly in terms of time and resources (since additional data points may need to be re-labeled). Furthermore, it may require human intervention to supervise the process.

This opens up another area of research, which focuses on designing more adaptive and intelligent techniques, using the global retraining option only when it cannot be avoided. If a detected drift is classified as regional, partially updating the model is also a viable option. This method replaces obsolete decision tree nodes with new ones based on the updated concept, and is more efficient than global training, especially if the drift regions can be accurately detected and interpreted. Passive approaches continuously update the learner each time new data arrives. In this regard, more versatile ensemble-based techniques have been considered to tackle more complex

TABLE I: Summary of the concept drift detection, interpretation, and adaptation algorithms.

App	**Concept Drift Algorithms**	**Drift Detection**				**Drift Interpretation**					**Drift Adaptation**
		Monitoring System	**Pre-Prossing System**	**Statistical Calculator**	**Hypothesis Calculator**	**Drift Types**	**How**	**When**	**Period**	**Where**	
Federated Learning (Distributed)	([8] *(Unsupervised)*	Predefined	PCA	K-Means	Euclidean distance	Sudden		✓		✓	Sends a drift warning to the aggregation agent
	FedConD [11] *(Unsupervised)*	Predefined (queue size)	Local learner	Yates's correction	Pearson's chi-square test	Gradual, Incremental		✓		✓	Added penalty to the local model
	CDA-FedAvg [5] *(Unsupervised)*	Predefined (window length)	Local learner	Method of the moments	Cumulative Sum (CUSUM)	Sudden, Recurring, Gradual		✓	✓		Local training with the new concept on the predefined rounds
Machine Learning (Centralized)	EI-kMeans [7] *(Unsupervised)*	Predefined	Dimensionality reduction or data normalization	K-Means	Pearson's chi-square test	Sudden, Gradual		✓			Retraining the learner
	DiwE [6] *(Unsupervised)*	Dynamic (based on region sample)	Dimensionality reduction	Max-RDD Diversity ensembler selection	Voted by 10 Representative ensembles	Sudden, Gradual, Recurring Incremental	✓	✓		✓	Train the models with the adjusted weight based on the estimated risk of region drift
	CADE [13]	Predefined	Conservative learner	Median Absolute Deviation (MAD)	Euclidean distance	Sudden		✓			N/A
	FW-DDA [12]	Dynamic (fuzzy windowing method)	Learner	Error rate	Significance level	Sudden, gradual, Incremental		✓	✓		Create an adapted learner and replace the current learner
	HHT-CU [9] *(Unsupervised)*	Dynamic	Learner	Classification uncertainty	Layer-I: Significance level (Hoeffding's) Layer-II: Permutation test	Sudden		✓			Update the classifier and employ it to predict labels of incoming data
	HHT-AG [9] *(Unsupervised)*	Predefined	N/A	Kolmogorov-Smirnov (KS) statistics	L-I: KS-Test L-II: 2D-KS-Test	Sudden		✓			
	BNDM [14] *(Unsupervised)*	Predefined	Learner	Bayesian nonparametric test	P-Tree test	Sudden		✓		✓	N/A
	UMSW [4] *(Unsupervised)*	Predefined (Window length)	N/A	K-Means	Cumulative Sum (CUSUM)	Sudden, Gradual		✓			N/A
	ESCR [10] *(S-supervised)*	Dynamic	N/A	K-Means	Jensen-Shannon divergence	Sudden, Gradual		✓		✓	Updates the model using a limited number of labeled data
					Cumulative Sum (CUSUM)	Recurring		✓		✓	

recurring and gradual drifts. In DiwE [6], the severity and occurrence of the drift are voted by 10 representatives. This method also modifies the weights of instances according to newly-emerging concepts and selects the combination of ensembles with the highest diversity based on the disagreement index. If a drift is detected, DiwE trains the models with the adjusted weight based on the estimated risk of region drift.

Single classifier approaches are also a valid option when it comes to large-scale data streams. Another ensemble-based algorithm, called Efficient Semi-supervised Classification over data stream with Recurring concept drift and concept evolution (ESCR) [10], is a semi-supervised technique that updates the model with a limited number of labeled data. Hierarchical Hypothesis Testing framework with Classification Uncertainty (HTT-CU) [9] is a single classifier algorithm. As suggested by its name, HHT-CU measures the classification uncertainty for the current classifier. A normal concept is expected to be stationary. If there is a significant distribution change, the average uncertainty value may suggest a possible drift. If the algorithm determines that a drift has occurred, the machine can request a window of labeled data to update the old classifier and employs the new classifier to predict the labels of incoming data.

D. Analysis of the concept drift algorithms and techniques

Table I summarizes the concept drift detection, interpretation, and adaptation algorithms. In this regard, the choice of the concept drift detection algorithm can be based on the balance of detection accuracy, speed, and costly false positives. For the monitoring system, most of the compared concept drift applications use a predefined window length. This approach is simple and easier to implement. However, depending on how the windows are defined, the boundary between them can be overly rigid because concept drifts may last for a short period or gradually drift from one concept to another. This can cause critical data loss of features divided between the old and new concepts. FW-DDA [12] uses a fuzzy windowing technique where the usage of the shared data is maximized between the windows to achieve better accuracy. That being said, fuzzy windowing methods take more running time than other monitoring approaches, especially with voluminous datasets. This can be translated to higher overhead, yet better accuracy.

The second phase of the detection framework is the Pre-processing system. This module is not always implemented but improves the efficiency of the detection technique. A popular way to reduce the number of variables is Dimension Reduction [6]. Scalability has been a problem, but many approaches, such as PCA, have been developed to overcome this. This contributes to weight reduction, which can lead to better module communication and grant weight updates with independency. The difference between the pre-processing system techniques compared in Table I is the aggressiveness of the algorithm. Data reduction will result in better performance but can lead to less accurate predictions. HHT-AG [9], UMSW [4], and ESCR [10] skipped this phase, which means they prioritize accuracy over speed.

For the statistical calculator, most of the concept drift detection algorithms (ESCR [10]) opted for K-Means. This clustering method is known for its speed and efficiency [8]. However, it still suffers from clustering data with variable size and density and clustering outliers. In FedConD [11], authors opted for Yate's correction which is also known for its efficiency in combination with Pearson's chi-square test in order to determine if there is a significant association between two categorical variables. However, this approach has limited use cases when it can give a correct approximation. Yates's correction may adjust too far, and it is not suitable for small datasets. Other algorithms like DA-FedAvg [5] chose the method of the moments, which is an established procedure for finding point estimators. This method can lead to consistent and unbiased estimators. But their efficiency is questionable or can be improved. DiwE [6] used a custom algorithm called Maximum Region Drift Disagreement (Max RDD) diversity ensembler. As its name states, this algorithm measure the disagreement between ensembles about the existence of a region drift. This method should outperform a random ensembler selection. The algorithm aims to select the most contentious region sets. This can contribute to higher responsiveness without sacrificing the robustness. Another technique called Median Absolute Deviation (MAD) is used in CADE [13] to quantify variation. This method is known for its robustness. However, it suffers from efficiency problems, and can be outperformed by standard deviation. Kolmogorov- Smirnov (KS) statistic test is used in HHT-AG [9] to look for discrepancies between two data distributions. KS test is renowned for being accurate except near the center of the distribution, where deviations can be a problem. HHT-CU [9], on the other hand, uses classification uncertainty where the dramatic change of the uncertainty mean value may suggest a potential concept drift. This algorithm has been shown to be competitive in terms of sensitivity compared to its other unsupervised state-of-the-art counterparts. Unlike KS which is frequentist nonparametric, Bayesian nonparametric methods are noted to have an edge in terms of interpretability and prior knowledge utilization which can benefit detecting and understanding concept drifts.

For the hypothesis calculator, most of the methods compared in this research study are based on Cumulative Summation (e.g, ESCR [10], UMSW [4], and CDA-FedAvg [5]). This severity test method enables sensitive and continuous monitoring of a trainee's performance to objectively determine competency and the degree of the drift if occurred. Another severity test computes the Euclidean distance between the two clusters. This method is used in FCDC [8] and CADE [13]. Other methods, like significance level (e.g, FW-DDA [12], HHT-CU [9]), focus on the probability of making the wrong decision when the null hypothesis is true. The problem with this approach is the possibility of rejecting a null hypothesis that is true, which can lead to inaccurate predictions. It is noted that most concept drift detection algorithms try to solve only some types of concept drift, but they mostly focus on sudden and gradual drift.These types are the most common and the easiest to detect and make a counter-measurement to avoid irreversible drifts proactively. Additionally, all the algorithms compared in this quantitative study answer the question of when the drift occurred. The other questions are only sometimes acknowledged, which can be a result of a trade-off to satisfy the performance and lower the overhead.

Finally, for the drift adaptation, the analytic model is notified with the drift to be trained with the new concept (e.g, DA-FedAvg [5], DiwE [6], FW-DDA [12], EI-kMeans [7]). Retraining can be costly in terms of resources, especially, if the drift was not discovered in the early stages or in case of a false-positive. Nevertheless, adapting and maintaining the AI/ML analytical models at the producer level are necessary after the deployment, and this phase has to be counted for regarding the added overhead and the consumed resources.

VI. Conclusion

AI/ML is a key enabler of B5G networks providing communication service assurance and network optimization. This paper sheds light on AI/ML analytics service enablers that leverage the benefits of micro service architecture to facilitate data collection, service discovery, and request and reporting control, even across different network segments. It also provides an analysis of various AI/ML models with respect to the fundamental phases of an AI/ML model lifecycle management to detect and interpret performance drift, providing appropriate adaptations and corrective actions.

The choice of the adaptation strategy should be tailored to the type of AI/ML analytics services, CPU availability, desired sensitivity, false-positive and false-negative borderlines, number and nature of consumers, and level of security and robustness of the collected data. In this regard, many open questions need to be answered including the amount of training and data to be considered in case of concept drift, where the retraining process should take place and the desired level of automation. In addition, AI/ML enablers currently aim to facilitate edge (e.g., ETSI MEC) and IoT services (e.g., oneM2M), allowing for intelligent services. Therefore, it is necessary to develop standards for interworking between AI/ML models in different layers and for federated learning.

Acknowledgment

This research work is partially supported by the European Union's Horizon 2020 Research and Innovation Program through the Charity and aerOS projects under Grant No. 101016509 and 101069732, respectively; and the Academy of Finland IDEA-MILL project under Grant No. 352428. Prof. Song was supported by the Technology Innovation Program (RS-2022-00154678, Development of Intelligent Sensor Platform Technology for Connected Sensor) funded By the Ministry of Trade, Industry & Energy. Prof. Song and Prof. Taleb are co-corresponding authors of this work.

References

[1] K. Samdanis and T. Taleb, "The Road beyond 5G: A Vision and Insight of the Key Technologies," *IEEE Network*, vol. 34, no. 2, pp. 135–141, Mar/Apr. 2020.

[2] P. V. Klaine, M. A. Imran, O. Onireti, and R. D. Souza, "A survey of Machine Learning Techniques applied to Self-organizing Cellular Networks," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, p. 2392–2431, 4th Quart. 2017.

[3] Y. Sun, M. Peng, Y. Zhou, Y. Huang, and S. Mao, "Application of Machine Learning in Wireless Networks: Key Techniques and Open Issues," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3072–3108, 4th Quart. 2019.

[4] Y. Yuan, Z. Wang, and W. Wang, "Unsupervised Concept Drift Detection based on Multi-scale Slide Windows," *Ad Hoc Networks*, vol. 111, p. 102325, 2021.

[5] F. E. Casado, D. Lema, M. F. Criado, R. Iglesias, C. V. Regueiro, and S. Barro, "Concept Drift Detection and Adaptation for Federated and Continual Learning," *Multimedia Tools and Applications*, vol. 81, p. 3397–3419, 2022.

[6] A. Liu, J. Lu, and G. Zhang, "Diverse Instance-weighting Ensemble based on Region Drift Disagreement for Concept Drift Adaptation," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 1, pp. 293–307, 2020.

[7] ——, "Concept Drift Detection via Equal Intensity k-Means Space Partitioning," *IEEE Transactions on Cybernetics*, vol. 51, no. 6, pp. 3198–3211, 2021.

[8] D. M. Manias, I. Shaer, L. Yang, and A. Shami, "Concept Drift Detection in Federated Networked Systems," *IEEE GLOBECOM*, Mardid, Dec. 2021.

[9] S. Yu, X. Wang, and J. C. Príncipe, "Request-and-Reverify: Hierarchical Hypothesis Testing for Concept Drift Detection with Expensive Labels," *27th International Joint Conference on Artificial Intelligence (IJCAI'18)*, p. 3033–3039, Stockholm, Jul. 2018.

[10] X. Zheng, P. Li, X. Hu, and K. Yu, "Semi-supervised Classification on Data Streams with Recurring Concept Drift and Concept Evolution," *Knowledge-Based Systems*, vol. 215, p. 106749, 2021.

[11] Y. Chen, Z. Chai, Y. Cheng, and H. Rangwala, "Asynchronous Federated Learning for Sensor Data with Concept Drift," *IEEE International Conference on Big Data*, Orlando, Dec. 2021.

[12] A. Liu, G. Zhang, and J. Lu, "Fuzzy Time Windowing for Gradual Concept Drift Adaptation," in *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE, 2017, pp. 1–6.

[13] L. Yang, W. Guo, Q. Hao, A. Ciptadi, A. Ahmadzadeh, X. Xing, and G. Wang, "CADE: Detecting and Explaining Concept Drift Samples for Security Applications," in *30th USENIX Security Symposium)*, 2021, pp. 2327–2344.

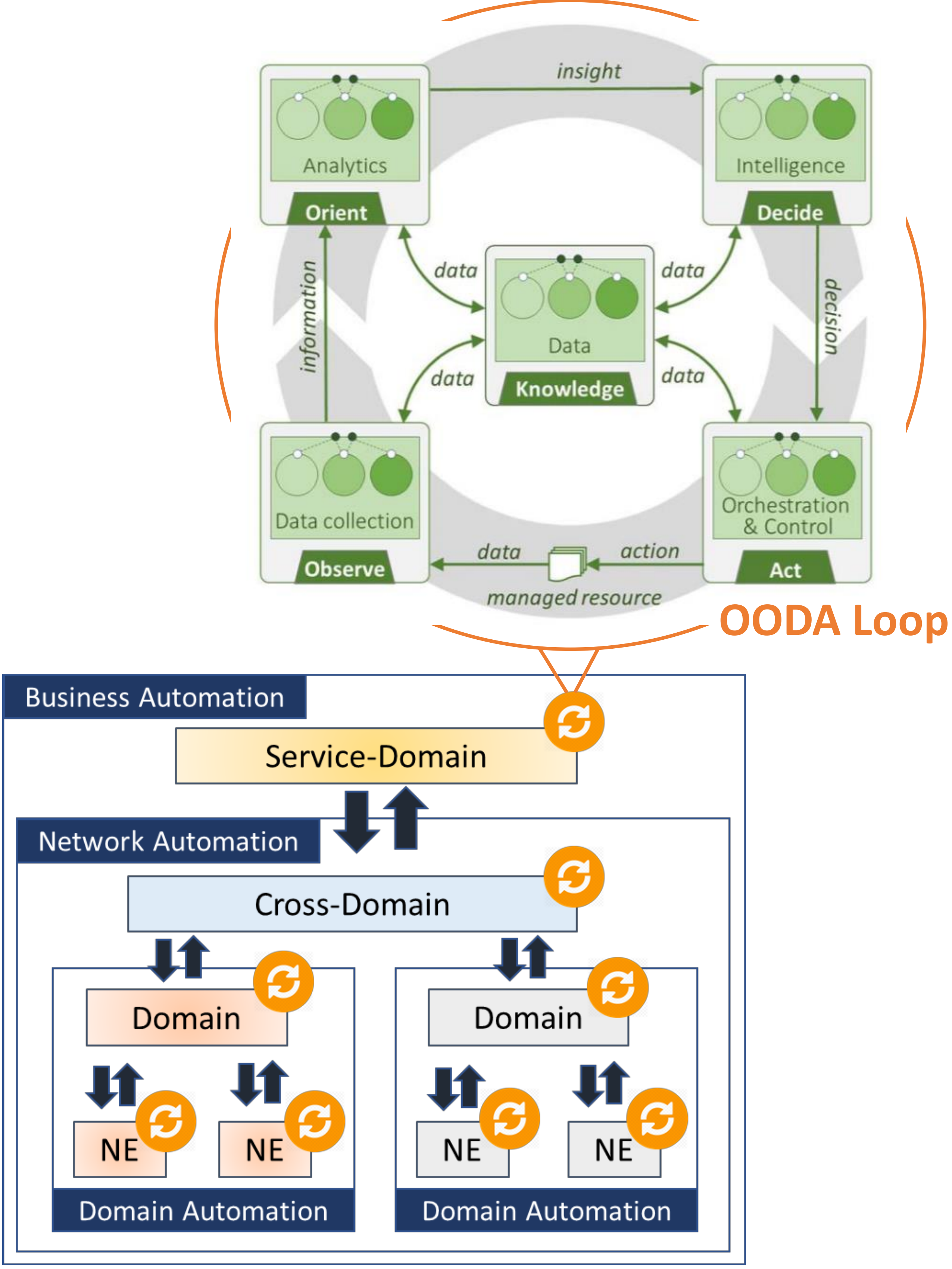
[14] J. Xuan, J. Lu, and G. Zhang, "Bayesian Nonparametric Unsupervised Concept Drift Detection for Data Stream Mining," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 12, no. 1, pp. 1–22, 2020.

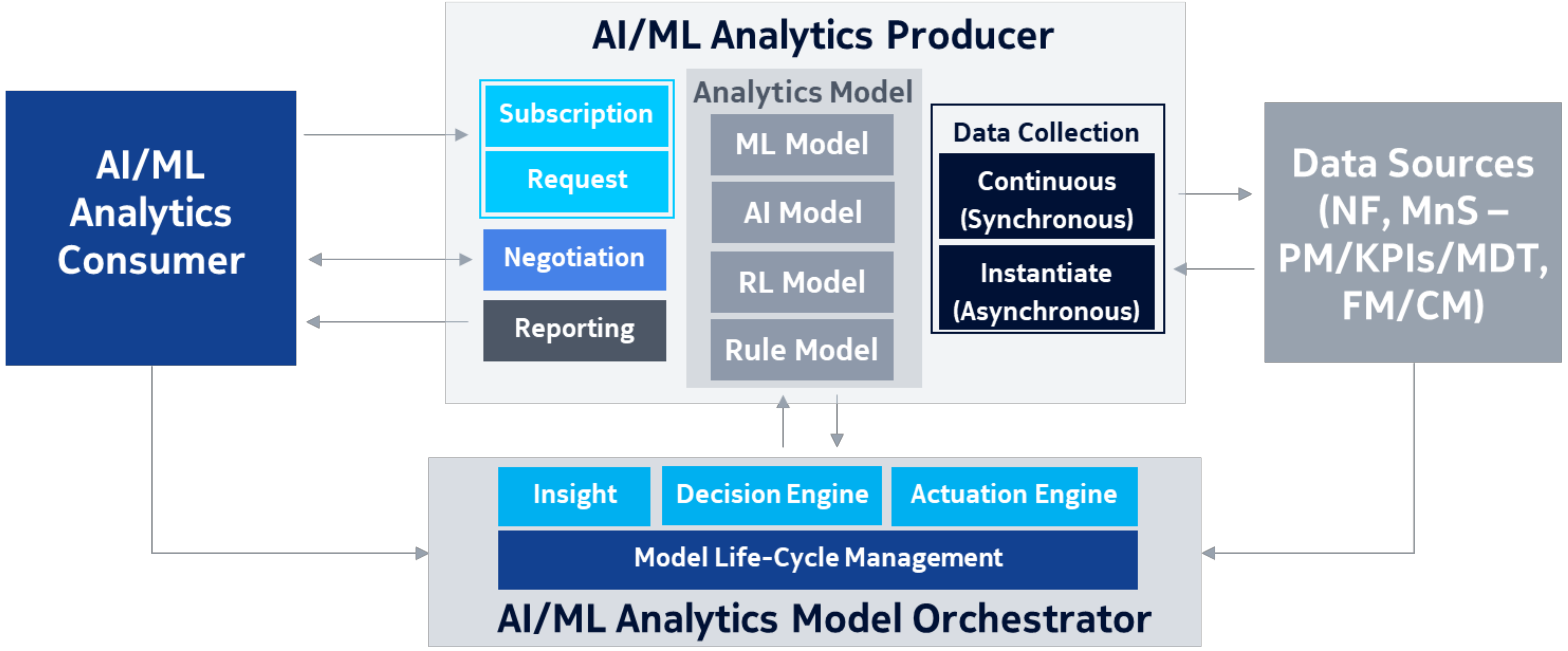
Konstantinos Samdanis is a 5G standardization expert at Lenovo. He was appointed as a project manager on network automation at Nokia. Before he was employed at Hauwei as a principal researcher and previously worked for NEC Europe as a senior researcher and broadband specialist. He received his Ph.D from Kings College London.

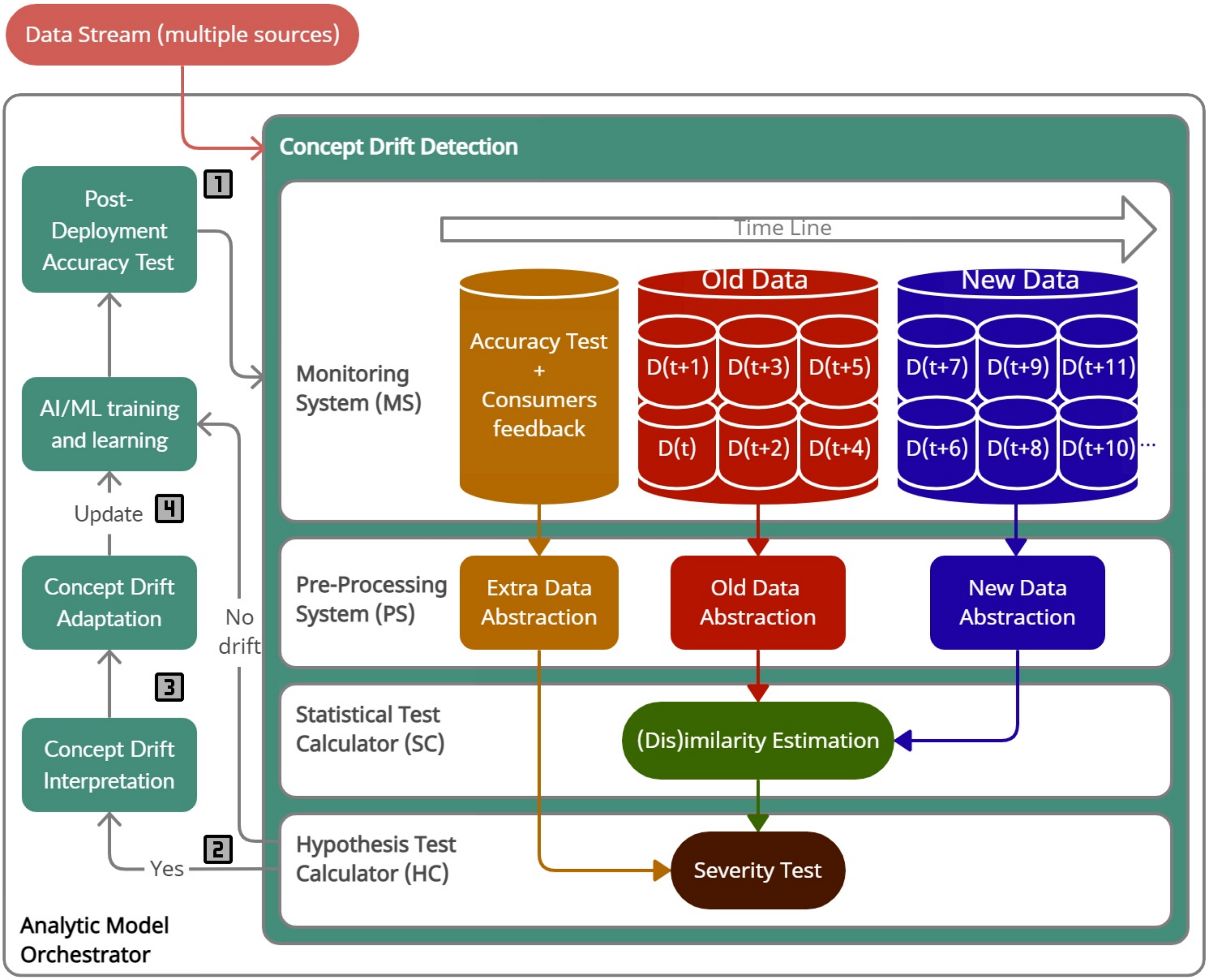
Aiman Nait Abbou is currently a doctoral student at Aalto University. He obtained his B.E. degree in telecommunications and computer networking from the Faculty of Sciences Dhar El Mahraz (FSDM) in 2014, and his M.Sc. degree in computer networking from the Faculty of Science and Techniques (FST) in 2016.

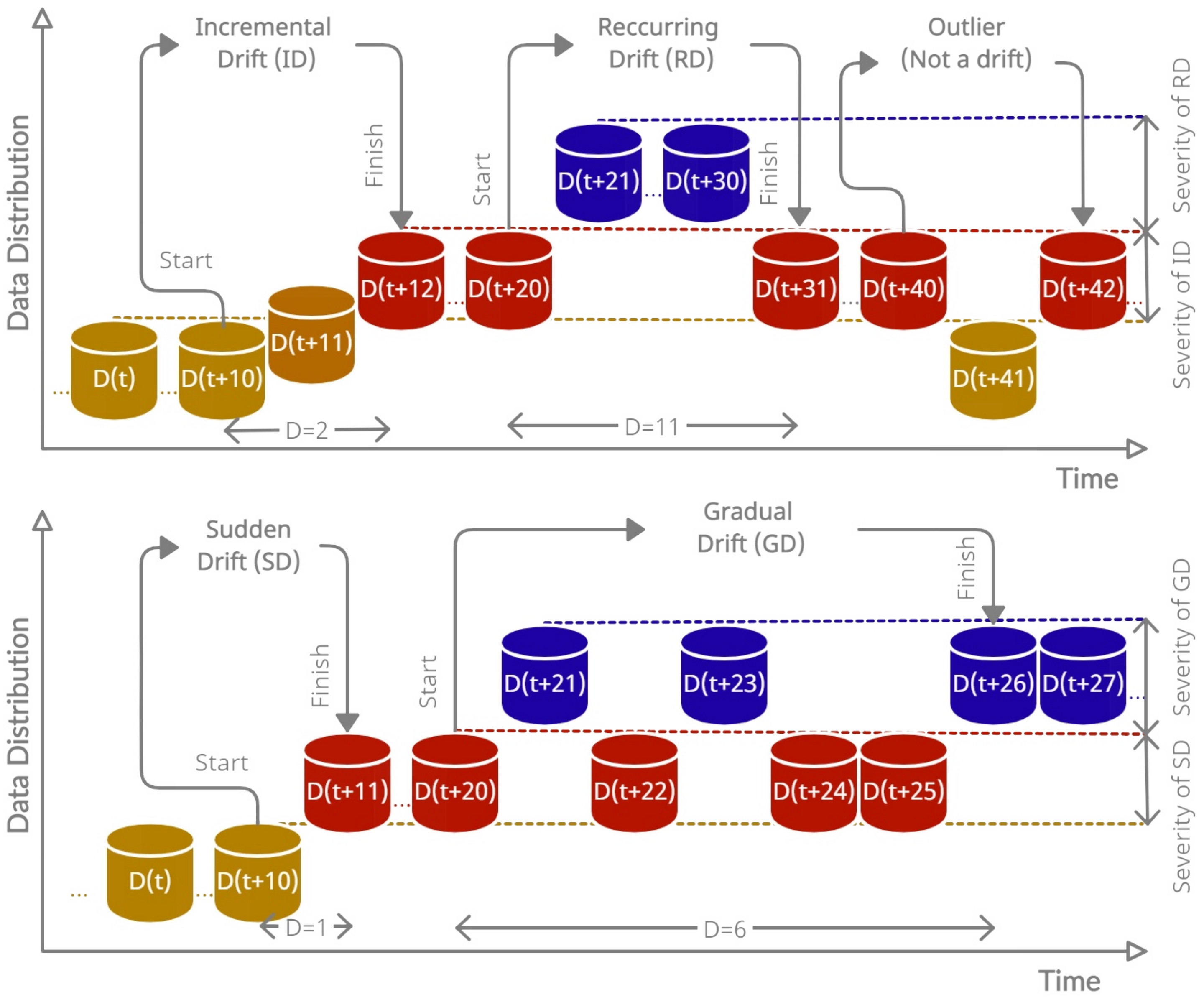
JaeSeung Song is a professor at Sejong University. He received a Ph.D. from Imperial College London in the Department of Computing, United Kingdom. He holds B.S. and M.S. degrees in computer science from Sogang University.

Tarik Taleb is a Professor at University of Oulu, Finland. Between Oct. 2014 and Dec. 2021, he was a Professor at Aalto University. He also worked as assistant professor at Tohoku University. He holds a B.E. degree in information engineering, and M.Sc. & Ph.D. degrees in information sciences from Tohoku University.


insight
Analytics
Orient
Intelligence
Decide
information
data
data
Data
Knowledge
decision
data
data
Data collection
Observe
Orchestration & Control
Act
data
action
managed resource
OODA Loop
Business Automation
Service-Domain
Network Automation
Cross-Domain
Domain
Domain
NE
NE
NE
NE
Domain Automation
Domain Automation


AI/ML Analytics Producer
AI/ML Analytics Consumer
Subscription
Request
Negotiation
Reporting
Analytics Model
ML Model
AI Model
RL Model
Rule Model
Data Collection
Continuous (Synchronous)
Instantiate (Asynchronous)
Data Sources (NF, MnS – PM/KPIs/MDT, FM/CM)
Insight
Decision Engine
Actuation Engine
Model Life-Cycle Management
AI/ML Analytics Model Orchestrator


Data Stream (multiple sources)
Concept Drift Detection
1
Post-Deployment Accuracy Test
Time Line
Monitoring System (MS)
Accuracy Test + Consumers feedback
Old Data
D(t+1) D(t+3) D(t+5)
D(t) D(t+2) D(t+4)
New Data
D(t+7) D(t+9) D(t+11)
D(t+6) D(t+8) D(t+10) ...
AI/ML training and learning
Update
4
Pre-Processing System (PS)
Extra Data Abstraction
Old Data Abstraction
New Data Abstraction
Concept Drift Adaptation
No drift
3
Statistical Test Calculator (SC)
(Dis)imilarity Estimation
Concept Drift Interpretation
2
Yes
Hypothesis Test Calculator (HC)
Severity Test
Analytic Model Orchestrator



Incremental Drift (ID)
Reccurring Drift (RD)
Outlier (Not a drift)
Start
Finish
Start
Finish
D(t)
D(t+10)
D(t+11)
D(t+12)
D(t+20)
D(t+21)
D(t+30)
D(t+31)
D(t+40)
D(t+41)
D(t+42)
D=2
D=11
Severity of RD
Severity of ID
Data Distribution
Time
Sudden Drift (SD)
Gradual Drift (GD)
Start
Finish
Start
Finish
D(t)
D(t+10)
D(t+11)
D(t+20)
D(t+21)
D(t+22)
D(t+23)
D(t+24)
D(t+25)
D(t+26)
D(t+27)
D=1
D=6
Severity of GD
Severity of SD
Data Distribution
Time