End-to-End AI Lifecycle Management for 6G: Bridging DataOps and MLOps

Menuka Perera Jayasuriya Kuranage, Ameur Mazene, Abdelkader Mekrache, Adlen Ksentini

***Abstract*—As 6G networks evolve toward a data-centric architecture with full automation capabilities, the softwarization of network functions enables access to a wide spectrum of data across technical, management, and operational domains. This rich data provides a strong foundation for building efficient, dynamic, and scalable network environments, where Artificial Intelligence (AI) and Machine Learning (ML) play a vital role in enabling data-driven intelligence within 6G networks. Consequently, streamlining AI/ML workflows becomes essential to fully harness the potential of 6G. Given the massive volume of data generated and the extensive applicability of AI/ML across various layers of the 6G architecture, including the Radio Access Network (RAN), Core, and Edge, managing AI/ML workflows at scale presents significant challenges. To address these challenges, we propose a novel microservice-oriented DataOps-MLOps framework designed to streamline AI/ML workflows across all domains. The proposed framework natively supports model training, model versioning and incorporates an automated data pipelining approach that facilitates seamless data collection and real-time inference. To validate the framework's effectiveness, we implemented a traffic anomaly detection use case involving end-to-end (E2E) ML model training, deployment, and real-time inference. Experimental results demonstrate that the proposed framework significantly streamlines AI/ML operations, unlocking the full potential of AI/ML integration in future 6G networks.**

***Index Terms*—MLOps, DataOps, Automation, 6G Networks, Service-Based Architecture, Kubernetes.**

I. Introduction

As 6^{th} generation (6G) networks transition from 5^{th} generation (5G) systems, there is a significant paradigm shift toward fully data-driven network management. This transformation is critical to enabling highly automated network management and orchestration, paving the way for more efficient, high-performance networks. It encompasses not only routine operations such as resource allocation, traffic load balancing, and fault management but also predictive functionalities, including resource forecasting, operational cost optimization, proactive congestion control, network function migration, placement, scaling, and more. This data-centric transformation is largely driven by the widespread adoption of softwarization and virtualization across the network stack, spanning the Radio Access Network (RAN), core, and edge networks, all of which are increasingly moving toward cloud-native architectures. As a result, network elements now generate massive volumes of heterogeneous data, including infrastructure-level metrics, application-level telemetry, and service-level information. These data streams are instrumental in supporting intelligent and efficient network management and orchestration.

Analyzing high-dimensional, interdependent data poses significant challenges. AI and ML technologies are critical enablers, offering real-time analytics, prediction, and decision support through pattern recognition in large-scale datasets.

However, managing the AI/ML lifecycle, including training, deployment, monitoring, and updating, is complex. Each use case often requires customized models trained on task-specific, validated data. Post-training, models must be evaluated before deployment and rely on real-time or batch data streams for inference, with outputs post-processed into actionable insights. Additionally, continuous monitoring is essential, as performance can degrade over time due to data or concept drift [1], necessitating retraining and redeployment.

Therefore, ML and data pipelines must be adaptable, with robust versioning for models and datasets, and a flexible, modular framework to allow on-demand updates. This dynamic lifecycle management increases operational complexity, especially in large-scale, data-intensive environments like 6G networks. Integrated MLOps frameworks are crucial to automate retraining, redeployment, and version control. Designing custom ML and data pipelines for such complex networks requires advanced frameworks for scalable, adaptive, and context-aware data processing.

Recognizing the critical importance of AI/ML integration in future network architectures, network standardization bodies have already initiated efforts to define standards for incorporating MLOps and Data Operations (DataOps) into next-generation networks [2]. As recognized by 3^{rd} Generation Partnership Project (3GPP), a fundamental step in this process is to clearly identify and formalize the stages of the AI/ML workflow [3]. This understanding is essential for effectively defining the roles and requirements of MLOps and DataOps within the context of 6G networks.

In this paper, we address the critical challenges associated with integrating MLOps and DataOps within the context of 6G networks. To tackle these challenges, we propose a unified framework that brings together MLOps and DataOps to enable scalable, flexible, and efficient support for ML-driven network intelligence. On the MLOps side, the framework supports both model training and inference processes, managing the complete E2E lifecycle of ML models, including training, tracking, evaluation, deployment, and monitoring, ensuring consistent operationalization of ML workflows. From a DataOps perspective, the framework integrates data pipelines from the RAN, core, and edge domains, treating them as primary data sources. A standardized common data model is introduced to harmonize data representation and facilitate unified data processing across disparate sources. The system enables users to define custom data requests easily, selecting from a wide range of metrics, customizing data extraction frequencies, and configuring scheduling parameters for automated data collection and inference operations. To validate our framework, we implemented a traffic anomaly detection use case in which an Long Short-Term Memory (LSTM) based autoencoder model is trained, deployed, and executed for real-time inference using the proposed framework. The results demonstrate that the framework delivers efficient orchestration of both MLOps and DataOps processes within a 6G network environment.

The remainder of this paper is structured as follows: Section II presents the design and implementation of the proposed framework. Section III describes the traffic anomaly detection use case and provides a detailed evaluation of the framework's performance. Section IV concludes the paper and outlines future research directions.

II. Proposed Platform Architecture

To address the challenges of streamlining MLOps operations and DataOps within 6G networks, we propose a novel E2E framework designed to manage both ML model lifecycles and data operation lifecycles across RAN, core, and edge network domains. The proposed framework aims to automate each stage of the MLOps and DataOps pipelines, thereby minimizing manual configurations and reducing complexity for end users. Built on a microservices architecture, the framework is designed with scalability, robustness, and resource efficiency as primary considerations, enabling seamless deployment and management in highly dynamic 6G environments.

A. MLOps Workflow

For the MLOps workflow, the primary objective is to enable both model training and inference capabilities within the 6G network, along with seamless management of the intermediate stages of the ML lifecycle. To achieve this, we introduce a modular architecture, as illustrated in Fig. 1. ML consumer can initiate AI service requests, such as model training, model discovery, or inference execution, through exposed APIs via the orchestrator. The orchestrator follows the Service-Based Management Architecture (SBMA), which includes multiple 6G managers, each responsible for a specific management domain. Among these managers is the AI Manager, responsible for handling AI-related requests. When an ML consumer submits a request to the orchestrator for AI management tasks, the gateway forwards this request to the AI Manager, which then communicates with the AI Fabric. This latter manages both machine learning and data operations.

For model training, users are required to onboard their custom model definitions along with the corresponding training datasets into the framework. Upon initiation of a training request, the AI Fabric triggers an automated training pipeline. As part of this process, the AI Fabric interacts with the Model Registry, which manages experiment tracking, training progress, and evaluation workflows. This functionality is implemented using the open-source platform MLflow [4], which automatically logs critical experiment metadata such as hyperparameters, evaluation metrics, and system logs. Users can access the MLflow user interface through a dedicated endpoint to monitor experiment status in real time, visualize metrics, and analyze model performance. Trained models are subsequently registered in the MLflow Model Registry to support versioning, traceability, and streamlined deployment.

All trained model artifacts are stored in MinIO [5], an open-source object storage platform designed for high-performance workloads. Associated metadata, such as model configurations and training parameters, is stored in a structured relational database powered by PostgreSQL [6], ensuring consistency, efficient data retrieval, and data integrity.

For inference services, users can browse the catalog of available models exposed via the AI Fabric's model discovery endpoints. Based on the provided model metadata, users can select the appropriate model and version for their application. Once a model is selected, it can be deployed to serve inference workloads. The deployment process also includes the instantiation of a dedicated Data Consumer component, which establishes connectivity with the data broker to acquire relevant input data. The Data Consumer is responsible for performing the necessary preprocessing steps to ensure compatibility with the model's input schema, as defined during the training phase. This component is customized for each deployed model to accommodate specific preprocessing logic.

Following preprocessing, the data is forwarded to the deployed model for inference execution. A key feature of this framework is that both the model and its corresponding Data Consumer are deployed as independent microservices, allowing each to scale autonomously under dynamic workloads using Kubernetes Horizontal Pod Autoscaling (HPA). This design ensures high availability, scalability, and isolation between components. Moreover, the architecture supports concurrent deployments, allowing multiple users to deploy their models in parallel and independently access inference outputs.

B. DataOps Workflow

In the DataOps workflow, managing the complete data operations lifecycle is critical to supporting the MLOps pipeline. This includes both data collection for model training and the provisioning of data for inference. The lifecycle encompasses data extraction from various sources, transformation, and delivery to downstream consumers.

The data collection and inference process is initiated by the user through AI Fabric via the orchestrator. Upon receiving the request, the AI Fabric coordinates the data acquisition workflow by passing the relevant parameters to the Data Collector component. The Data Collector then communicates with the data broker, implemented using Apache Kafka [7], to create a dedicated topic for the current data stream. This topic acts as a temporary storage and routing layer, allowing extracted and transformed data to be made available for downstream access. Once the topic is created, the Data Collector selects the appropriate data source and forwards the request to the corresponding adapter. This request includes

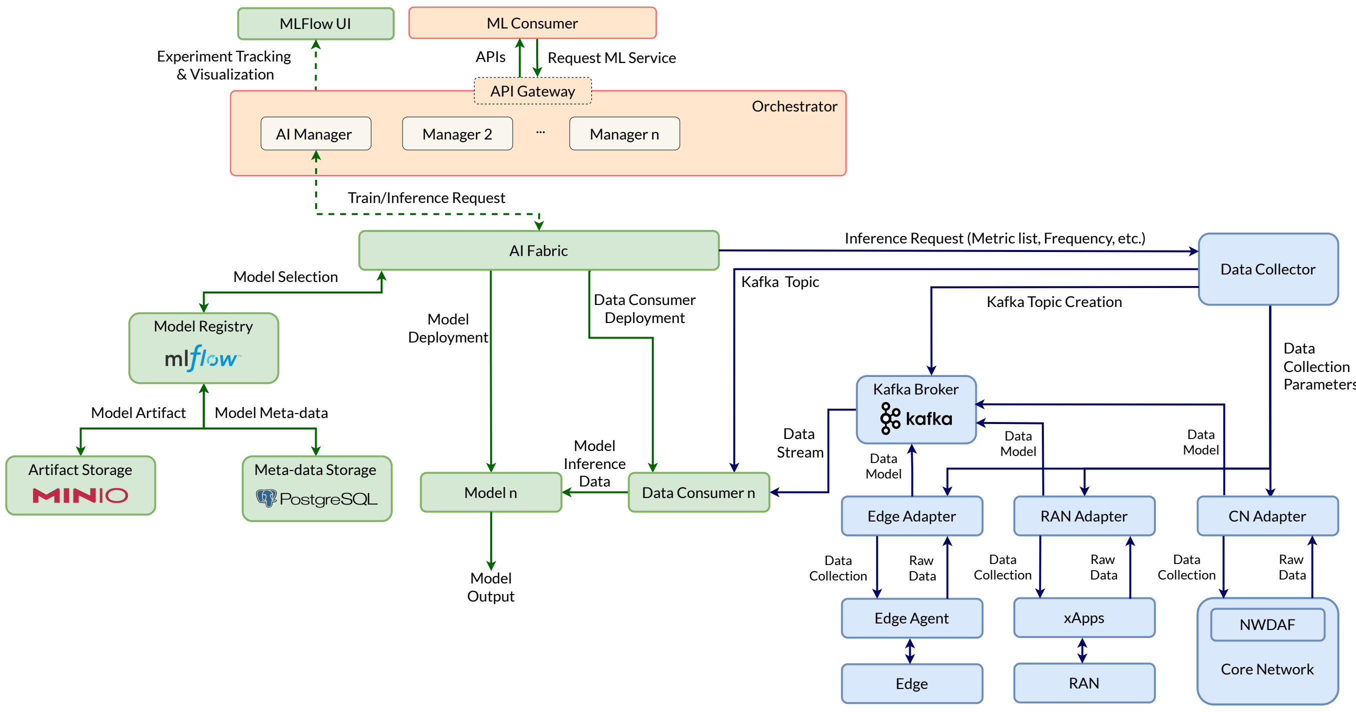

Fig. 1. The proposed MLOps-DataOps framework.

the data collection parameters and the Kafka topic reference. The adapter is responsible for connecting to the specified data source, extracting and transforming the data as required, and publishing it to the assigned Kafka topic. The processed data is then accessible to consumers such as end user or Data Consumer components. The primary data sources in the proposed framework span the Core, RAN, and Edge network domains. Data from the Core Network is collected via the Network Data Analytics Function (NWDAF), which provides access to all 3GPP-compliant metrics. For the RAN domain, an xApp is utilized to gather telemetry through FlexRIC [8]. At the Edge, data is collected from edge applications using a dedicated Edge Agent. In each domain, a specialized adapter is deployed to manage the data acquisition process.

Each adapter receives detailed data collection task parameters, including the start and stop times, a metric list specifying the required data fields, the data extraction frequency, and the target Kafka topic for data publication. A key feature of this framework is its support for configurable extraction frequencies, allowing adaptation to diverse ML model requirements. For instance, anomaly detection models such as those targeting DoS attacks may require fine-grained, high-frequency data, while models used for long-term tasks like capacity planning can function effectively with lower-frequency data.

Another important feature of the framework is its support for topic sharing across models. If multiple models rely on the same data, they can subscribe to a common Kafka topic, thereby eliminating redundant parallel data collection efforts. Any Data Consumer can subscribe to these Kafka topics to retrieve the published data either in real time or in batch mode, depending on the specific application needs. Another critical function of each adapter is to convert raw extracted data into a unified, customized data model. This facilitates consistency and interoperability across the entire DataOps framework. The data model defines several key fields: the origin of the data source, a metric identifier (metric ID), and any associated labels that capture additional subfields or contextual metadata. Furthermore, a timestamp indicating the exact time of data collection and the corresponding value. An example of the data model utilized is presented in Fig. 2.

III. Use case: Core Network Traffic Anomaly Detection

To evaluate the effectiveness of the proposed platform, a proof-of-concept use case was implemented, focusing on the detection of anomalous behavior in 5G core network traffic. This use case was selected based on its alignment with the 3GPP specifications pertaining to the NWDAF [9].

A. *Experimental setup*

The proposed E2E DataOps-MLOps framework was deployed as a set of microservices on a Kubernetes [10] cluster consisting of three nodes. Each node was provisioned as a Virtual Machine (VM) running Ubuntu 20.04, with one designated as the master node and the remaining two as worker nodes. The cluster was configured to use ”Containerd” [12] as the container runtime. The OpenAirInterface (OAI) 5G Core Network [12], including the NWDAF, was deployed separately on a VM outside the Kubernetes cluster using Docker [13]

```
{
  "source": "NWDAF",
  "metric_id": "NUM_OF_UE",
  "labels": "NETWORK_PERFORMANCE",
  "timestamp": "2025-10-23T12:00:00.000Z",
  "value": 1
}
```

(a) Data model for Number of UEs collected from NWDAF

```
{
  "source": "NWDAF",
  "metric_id": "UPLINK_VOL",
  "labels": "UE_COMMUNICATION",
  "timestamp": "2025-10-23T12:00:00.000Z",
  "value": 150
}
```

(b) Data model for uplink data volume collected from NWDAF

Fig. 2. Data models collected from NWDAF for different metrics.

containers. In this setup, the adapter communicates with the NWDAF via APIs to retrieve the required data.

After deploying the framework, a model must be trained and integrated for inference to evaluate the platform's effectiveness. A LSTM-based autoencoder model was implemented for the 5G core network anomaly detection use case. The training process utilized the "Telecommunications - SMS, Call, Internet - MI" dataset, which comprises detailed traffic volume data collected by Telecom Italia over the span of one year [14]. This dataset includes extensive records of user connectivity events, with the analysis focusing specifically on the volume of internet data exchanged with users. To prepare the input data, it was filtered and structured into sequences incorporating temporal features such as weekday, hour, and minute, along with corresponding internet traffic volumes. These features were included to capture daily patterns inherent in network usage behavior.

For the model design, implementation was carried out using TensorFlow and consisted of a two-layer encoder and a two-layer decoder. The encoder processed input sequences of length 12 through stacked LSTM layers with 32 and 16 units, respectively. All input features were standardized using the StandardScaler, and the model was trained to minimize the mean squared error using the Adam optimizer. Training was executed through the proposed framework, which initiated a predefined training pipeline image on the platform. Upon completion, the trained model was automatically registered in the MLflow registry, with the model artifact stored in MinIO and associated metadata saved in a PostgreSQL database.

Following model evaluation, the best-performing version was selected from the model registry for deployment. All models in the registry are exposed via standardized API interfaces, enabling consistent and scalable integration across services.

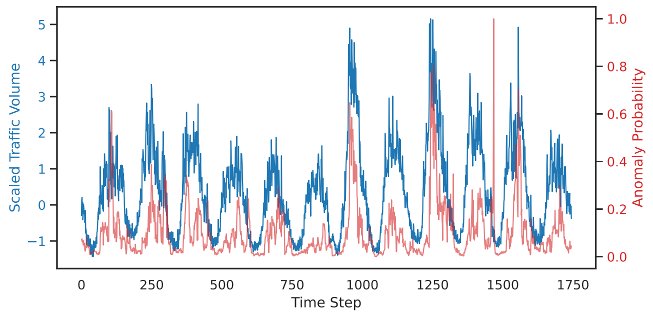

Fig. 3. Probabilities of abnormal traffic behaviors in the core network.

Upon deployment, a dedicated Data Consumer component was instantiated to receive input data from the data broker. This component performs preprocessing operations on the incoming data and forwards the processed input to the deployed model, which outputs the corresponding anomaly probability as the inference result.

B. Evaluation Results

1) Anomaly Detection Model Validation: Fig. 3 illustrates the relationship between scaled 5G core traffic volume (blue line) and the corresponding anomaly probability (red line) over time. The LSTM autoencoder model effectively captures regular traffic patterns, with higher anomaly probabilities consistently aligning with sharp spikes or irregular fluctuations in traffic volume. These peaks in anomaly probability suggest the model's sensitivity to deviations from learned temporal patterns, indicating potential abnormal behavior. Notably, periods of steady traffic correspond to low anomaly scores, reinforcing the model's ability to distinguish normal from anomalous patterns. Overall, the figure confirms that the model can accurately detect unusual traffic behavior in alignment with significant deviations in the input sequence.

2) Scalability of the framwork: Fig. 4 and 5 present the CPU and memory utilization across the MLOps and DataOps namespaces during the deployment of varying numbers of models and data consumers. Three configurations were assessed: (1) one model with one consumer, (2) two models with two consumers, and (3) three models with three consumers. As shown in Fig. 4, CPU consumption increases with workload in both namespaces. In the MLOps namespace, CPU usage rises from approximately 45 millicores to over 70 millicores as the number of models and consumers increases. In contrast, CPU usage in the DataOps namespace grows more modestly, increasing from around 8 millicores to just above 12 millicores. Fig. 5 illustrates memory usage trends, which also scale with workload. In the MLOps namespace, memory usage increases from about 3200 MB to over 4000 MB. DataOps memory usage grows slightly as well, ranging from around 180 MB to just under 250 MB. Overall, these results demonstrate that the MLOps namespace incurs significantly higher resource utilization compared to DataOps. However, both namespaces show

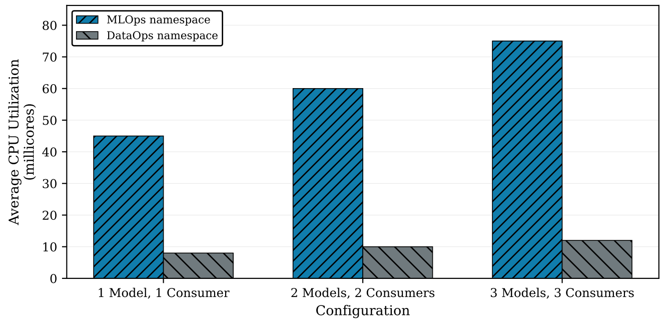

Fig. 4. CPU utilization trends in MLOps and DataOps namespaces during concurrent deployments of models and data consumers on the platform.

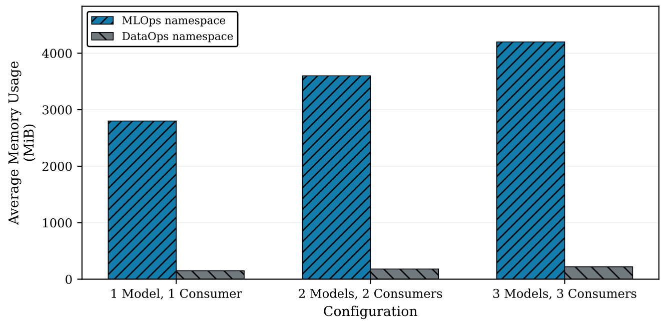

Fig. 5. RAM utilization trends in MLOps and DataOps namespaces during concurrent deployments of models and data consumers on the platform.

consistent and predictable scaling, confirming the platform's ability to handle increasing deployment demands efficiently while maintaining operational stability.

3) Deployment Time Analysis: To evaluate the performance of the platform's deployment mechanisms, Table I reports the average deployment times and their standard deviations for both the model and data consumer components. The results show that model deployment takes an average of approximately 7.58 seconds, while the data consumer is deployed in about 3.35 seconds. When both components are deployed sequentially, the total time averages around 8.35 seconds, reflecting minimal orchestration overhead between the two stages. The relatively low standard deviation values across all cases indicate stable and predictable deployment behavior, which is crucial for maintaining operational efficiency in dynamic and scalable environments.

TABLE I: Statistics of deployment times for model and data consumer.

Component	**Mean Time (ms)**	**Std. Dev. (ms)**
Model Deployment	7579.97	1471.40
Data Consumer Deployment	3349.62	1070.79
Combined Sequential Deployment	8350.36	1392.41

IV. Conclusion

In this work, we proposed a novel DataOps/MLOps architecture tailored for 5G and 6G networks. The proposed solution integrates AI/ML pipeline stages with DataOps workflows across the RAN, core, and edge domains. The architecture is modular and adheres to a microservices-based design, enabling high availability, scalability, and operational flexibility. The system was implemented and validated through the deployment of a core network traffic anomaly detection model. Evaluation results demonstrate the architecture's capability to support seamless E2E model training and real-time inference. As future work, we aim to extend the solution by adding model performance monitoring capabilities and seamless model re-deployment support to the framework, as well as a data space layer that facilitates dataset storage, metadata management, and version-controlled dataset lifecycle operations.

Acknowledgment

This paper was supported by the European Union's Horizon Program under 6G-DALI (Grant Agreement No. 101192750) and SUNRISE-6G projects (Grant No. 101139257).

References

[1] A. S. Iwashita and J. P. Papa, "An Overview on Concept Drift Learning," IEEE Access, 2019.
[2] J. Naumova, "MLOps for Highly Autonomous Networks," NGMN. Accessed: May 02, 2025. [Online]. Available: https://www.ngmn.org/publications/mlops-for-highly-autonomous-networks.html
[3] 3GPP, Technical Specification Group Services and System Aspects; Management and orchestration; Artificial Intelligence / Machine Learning (AI/ML) management, 28.105, Version 19.1.0, 2024.
[4] MLflow, MLflow: An Open Source Platform for the Machine Learning Lifecycle, Version 2.11.1, The Linux Foundation. Accessed: May 02, 2025. [Online]. Available: https://mlflow.org
[5] MinIO — S3 Compatible Storage for AI. MinIO, Inc. Accessed: May 02, 2025. [Online]. Available: https://min.io
[6] PostgreSQL, Version 17.4, The PostgreSQL Global Development Group, Accessed: May 02, 2025.[Online]. Available: https://www.postgresql.org/
[7] Apache Kafka, The Apache Software Foundation, Version 4.0.0, Accessed: May 02, 2025. [Online]. Available: https://kafka.apache.org/
[8] FlexRIC, OpenAirInterface. Accessed: May 02, 2025. [Online]. Available: https://gitlab.eurecom.fr/mosaic5g/flexric
[9] Mekrache, A., Boutiba, K., & Ksentini, A., Combining network data analytics function and machine learning for abnormal traffic detection in beyond 5G. GLOBECOM, 2023.
[10] Kubernetes, The Linux Foundation, Version 1.33, Accessed: May 02, 2025. [Online]. Available: https://kubernetes.io/
[11] Containerd, The Linux Foundation, Version 1.6.2, Accessed: May 02, 2025. [Online]. Available: https://containerd.io/
[12] OAI-5G Core, OpenAirInterface, Version v2.0.0, Accessed: May 02, 2025. [Online]. Available: https://openairinterface.org/
[13] Docker, Docker Inc., Version 27.5.1, Accessed: May 02, 2025. [Online]. Available: https://www.docker.com/
[14] Italia Telecommunications - SMS, Call, Internet - MI (V1 ed.). Harvard Dataverse, 2015.