2023 14th International Conference on Information and Communication Technology Convergence (ICTC) | 979-8-3503-1327-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICTC58733.2023.10392296

A Microservice-based MLOps Platform for Efficient Development of AI Services in an Edge-Cloud Environment

Chorwon Kim, Geon-Yong Kim and Sungchang Kim
Honam Research Center
Electronics and Telecommunications Research Institute (ETRI)
Gwangju, Republic of Korea
{chorwon.kim, gykim, sungchang}@etri.re.kr

***Abstract*—With the advancement of Internet of Things (IoT) and Artificial Intelligence (AI) technologies, outcomes derived from diverse sets of collected data are finding applications in the business domain. In particular, edge AI technology, which combines edge computing and artificial intelligence, emerged to address real-time response limitations, is garnering considerable attention as a solution to the real-time response constraints faced by AI services within operated cloud environments. Nevertheless, the development of AI services within an edge-cloud environment presents numerous challenges, such as limited computing resources at the edge, the portability of learning models, and dependencies on code libraries during the build phase. Consequently, various open source projects and companies are actively advancing MLOps solutions. In this paper, we describe the development of a microservice-based MLOps platform utilizing open source for the development of AI services and the deployment of generated models in an edge-cloud environment built using lightweight edge hardware, software, and designated servers assuming the role of a cloud system.**

Keywords—Edge computing, Cloud computing, MLOps, Kubernetes, Kubeflow

I. INTRODUCTION

In recent times, advancements in technologies such as the Internet of Things (IoT) and artificial intelligence (AI) have led to a proliferation of diverse data types and a vast increase in data volume. This has prompted numerous industries to address challenges by employing a range of AI-driven services and solutions. Specifically, achieving high levels of accuracy through AI techniques in domains like object detection and anomaly detection requires substantial computational resources and memory during both the training and inference phases[1]. As a result, these services are commonly delivered via cloud-based system[2]. Nevertheless, the advent of edge computing technology has surfaced as a solution to address network challenges and real-time responsiveness limitations encountered during data transmission within cloud-based systems, Furthermore the emergence of Edge AI technology, which combines AI services with edge system for data collection, processing, and analysis, is garnering attention[3].

Utilizing edge computing technology enables real-time response to requests, However the development of high-level AI services also needs significant computational resources for data pre-processing and learning. Furthermore, within an edge computing environment comparatively constrained computational resources in contrast to cloud system, the process of learning may lead to service disruptions attributed to the deterioration of performance or system outages. To mitigate this issue, numerous companies offer a platform for training AI models in the cloud system using data gathered from edge system and subsequently deploying the learned models. This capability facilitates flexible handling of novel environments, datasets, specific events, and issues, it implies that AI service developers can engage in effective development tasks without the need to address the problem from computational resource disparities between cloud and edge systems.

With the ongoing trend in the advancement of lightweight edge hardware and software for edge computing, as well as the progress in edge AI technology, our research extend to the range of efficient development and management of AI services within an edge-cloud environment. In this paper, we describes an open source microservice-based MLOps platform for the development of AI services to operate in an edge-cloud environment and the deployment of models generated through them.

II. MLOPS TECHNOLOGY FOR MANAGEMENT OF EDGE COMPUTING SOFTWARE AND AI APPLICATION DEVELOPMENT

A. The Need to Manage Edge Computing Software and AI Applications

The edge computing software we are currently developing is implemented through the utilization of the EdgeX framework, an open source project managed by EdgeX Foundry in the microservice architecture[4]. Each of these components carries out its designated tasks by means of an interface established in the form of an API and operates as a container to provide an independent execution environment. Each component facilitates the application of edge computing technology through the interaction of data acquisition, parsing and storage, and delivery of processed data to the cloud or external server. More specifically, the interface established through APIs facilitate the implementation of data collection and export functionalities, enabling integration with inference engines operating in edge environments or servers dedicated to learning processes.

To address the need for real-time responsiveness, which is lacking in existing cloud-based AI services, the edge computing software is crafted with a set of approximately 14 logically isolated containerized components, and a learning model or inference engine operates for AI services. Due to the

nature of edge computing, which processes data in close proximity to sensors or end devices, the scope of management become broader compared to cloud environments. Consequently, this expanded management scope also encompasses software. Simultaneously, in order for various components to operate without any problems, it is necessary to respond quickly when a service failure occurs while ensuring the same execution environment in other workspaces.

B. MLOps Technology for Efficient AI Service Development

The management of containerized microservices encompasses the entire lifecycle, from the creation to the removal of each operational component. The service developer registers their development outcomes with the management server, which subsequently deploys the outcomes in a specific server meeting the specifications provided by the developer. In case of an issue arising within a running component or when an update, the management server autonomously identifies the problem and initiates a restart or update operation, eliminating the intervention of developer or administrator. This method, referred to as a development methodology called "DevOps", can be extended to the development and deployment of AI applications for AI services[5].

The concept of MLOps, which involves the automated management and operation of machine learning applications by integrating the development and deployment of machine learning models, facilitates the development and management of efficient AI services[6]. To achieve this objective, Google has identified several key components of MLOps, including data analysis and processing, model training, model evaluation, model deployment and validation, and model monitoring to ensure operational status[7]. Specifically, AI applications developed within isolated containers can offer an identical environment as the development phase, preventing issues that may arise due to differences in the infrastructure environment within the MLOps concept. This implies that data analysis and processing, the model development phase, and the deployment and management of models within a commercial environment can be seamlessly conducted. With these advantages, numerous open source projects and companies are actively advancing MLOps solutions.

III. Framework and Environment Design

By deploying AI models as inference engines within edge computing environments, analysis result using AI service can be quickly provided. Nonetheless, only edge system gives rise to challenges, such as constraints on computing resources for executing the complete tasks from AI model training to deployment. Hence, the process of AI model training occurs on the cloud server via the edge-cloud environment, and subsequently, the trained AI model is deployed to the edge system or server as an inference engine. Furthermore, the depicted platform, as illustrated in Figure 1, was designed to apply the MLOps concept to manage the lifecycle of AI application development and deployment.

Figure 1 is a diagram illustrating an MLOps concept designed for AI application within an edge-cloud environment. Developers utilize data samples from the edge environment to create AI services. Developers have the flexibility to undertake development tasks either on an assigned PC or within a cloud system, following which the developed model is uploaded to the cloud environment at the code level. The code uploaded to the cloud environment is trained and verified

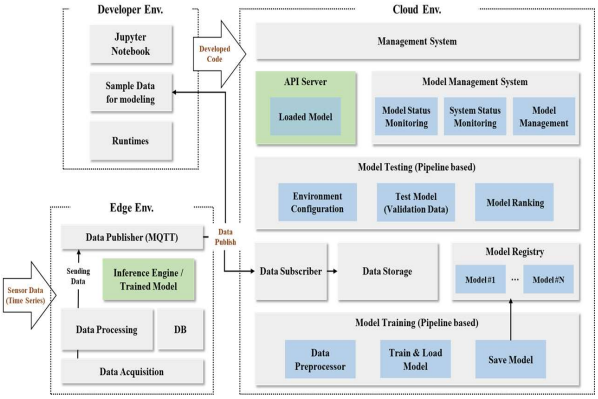

Figure 1. MLOps concept designed for AI Application development within an edge-cloud environment

within a containerized virtual environment constructed to replicate the model development. Additionally, we employ a structure known as a pipeline to encompass processed ranging from data preprocessing to model training and validation, storing the model with optimal weight values within the Model Registry. Furthermore, this pipeline, created using this approach, facilitates detection of data trend change within edge environment and enables automated model re-training and deployment.

This trained model offers two ways for delivering inference outcomes in response to data requests through the API server within the cloud environment (referred to as the offloading approach) or functioning within the edge environment as inference engine[8,9]. The mentioned method can be selected according to the computing resources available in the edge environment. For example, if it is determined that AI service operations within the edge environment could potentially disrupt the overall functioning of edge computing due to inadequate computational resources, the adoption of an offloading method becomes a viable option.

IV. Experimental Result

In order to implement an MLOps platform design applied for the edge-cloud environment, the configuration of the experimental setup is delineated as depicted in Figure 2.

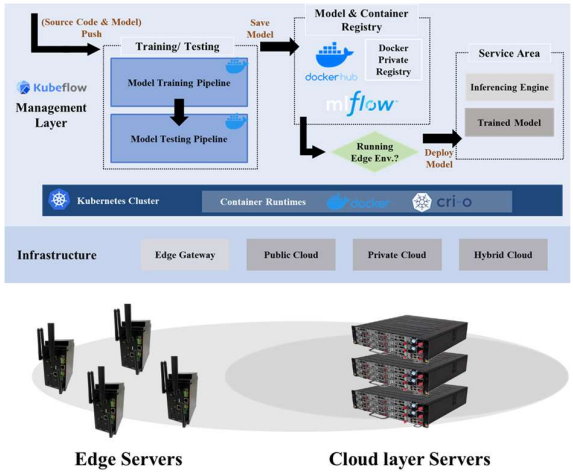

Figure 2. An edge-cloud environment and configured software built to apply MLOps platform design

In the case of the edge server, we leverage a gateway integrated with edge computing software[10]. The gateway was designed based on NVIDIA Jetson Xavier NX with 48 tensor cores to accelerate machine learning algorithms for data analysis, and Cloud layer Servers featuring Xeon CPUs and dual NVIDIA A40 GPUs, were employed to fulfill the role of the cloud computing environment.

In terms of software, we leverage various open source software to apply the MLOps platform. First, we employ Kubeflow, an AI platform built on Kubernetes serving as a container orchestration tool utilized for efficient management of edge computing software[11,12]. Kubeflow offers a comprehensive suite of tools and environments for all tasks, ranging from the training of AI models to their deployment within an configured Kubernetes environment. Following the learning process, the model with finely tuned weight values is deployed onto edge servers or cloud instances, thereby facilitating AI services. In this process, the task of preserving the trained model along with its optimized weight values is achieved through the utilization of MLflow which furnishes functionalities such as packaging and saving the model. Also a registry server was constructed with the intention of being deployed after wrapping the model in container images. Saved models or container images are deployed within a Kubernetes environment in accordance with the conditions of the edge-cloud environment.

For the validation of the MLOps platform designed and implemented for the development and deployment of AI services operating in an edge-cloud environment, we use a deep learning-based time-series data forecasting model. The mentioned model examines sensor data gathered at the edge to assess standard operation status, concurrently identifying abnormal data through data forecasting then take actions quickly. Hence, the designed and implemented MLOps platform was employed to enhance the efficiency of the model development process, with the outcomes depicted in Figure 3.

Figure 3 illustrates the outcomes of establishing and executing a pipeline processing model development, validation, storage and deployment, also the model is saved within MLflow, subsequently operating in the edge-cloud environment. When an AI service developer executes such a defined pipeline, the model undergoes training using stored data, and the trained model is either persisted as an object type or transformed into a container image for deployment in a containerized environment. With the saved results, it is possible to verify that Kubernetes can choose and deploys an operational environment within the edge-cloud environment, ensuring seamless AI service by considering the current environment status.

V. Conclusion

In this paper, we present the design and implementation of an MLOps platform utilizing Kubeflow and diverse open source tools for the creation of AI services within an edge-cloud environment. Additionally, we leverage a deep learning-based model to verify the outcomes. In the future, we plan to conduct research aimed at extending our capabilities to an automated MLOps platform that continuously monitor and analyze the status of the edge-cloud environment in real time, promptly responding to any anomalies. Furthermore, we aim to apply this platform across a range of deep learning models for diverse service.

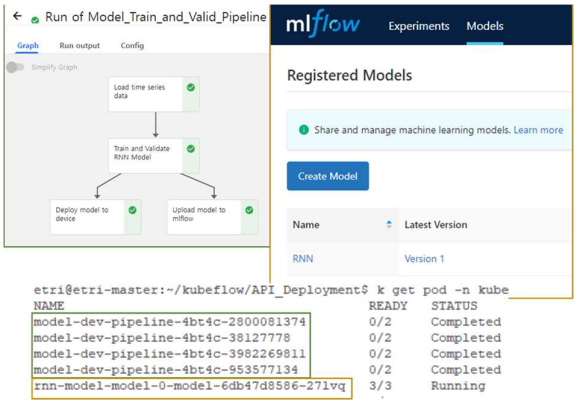

Figure 3. The results obtained from the implementation and execution of a pipeline encompassing model development, validation, storage, and deployment.

Acknowledgment

This work was supported by the Korea Institute of Energy Technology Evaluation and Planning (KETEP) and the Ministry of Trade, Industry and Energy (MOTIE) of the Republic of Korea (No.2021202090053B, Development and Demonstration of Cloud Energy Management System for Distributed Factories).

References

[1] J. Chen and X. Ran, "Deep Learning With Edge Computing: A Revice," in *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1655-1674, Aug. 2019.

[2] A. Banijamali, O.-P. Pakanen, P. Kuvaja, and M. Oivo, "Software architectures of the convergence of cloud computing and the Internet of Things: A systematic literature review," in *Information and Software Technology*, vol. 122, Jan. 2020.

[3] Raghubir Singh, and Sukhpal Singh Gill, "Edge AI: A survey," in *Internet of Things and Cyper-Physical Systems*, vol. 3, pp. 71-92, Mar. 2023.

[4] "EdgeX Foundry", Accessed on: July 28, 2021. [Online]. Available: https://www.edgexfoundry.org/

[5] Christof Ebert, Gorka Gallardo, Josune Hernantes, and Nicolas Serrano, "DevOps," in *IEEE Software*, vol. 33, no. 3, pp. 94-100, Apr. 2016.

[6] D. Kreuzberger, N. Kühl and S. Hirschl, "Machine Learning Operations (MLOps): Overview, Definition, and Architecture," in *IEEE Access*, vol. 11, pp. 31866-31879, Mar. 2023.

[7] "Practitioners guide to MLOps: A framework for continuous delivery and automation of machine learning", Accessed on: August 15, 2023. [Online]. Available: https://services.google.com/fh/files/misc/practitioners_guide_to_mlops_whitepaper.pdf

[8] Giha Yoon, Geun-Yong Kim, Hark Yoo, Sung Chang Kim, and Ryangsoo Kim, "Implementing Practical DNN-based Object Detection Offloading Decision for Maximizing Detection Performance of Mobile Edge Devices," in *IEEE Access*, vol. 9, pp. 140199-140211, Oct. 2021.

[9] Changsik Lee, Seungwoo Hong, Sungback Hong, and Taeyeon Kim, "Performance analysis of local exit for distributed deep neural networks over cloud and edge computing," in *ETRI Journal,* vol. 42, pp. 658-668, Oct. 2020.

[10] Chorwon Kim, Geun-Yong Kim, *et al*, "An opensource based software framework for analysis on high-speed generation industrail data," in *Summer Annual Conference of KIEE*, 2022, pp. 1786-1787

[11] "Kubeflow", Accessed on: August 15, 2023. [Online]. Available: https://www.kubeflow.org/

[12] "Kubernetes", Accessed on: August 15, 2023. [Online]. Available: https://kubernetes.io/