Cloud-based MLOps service

Overview

Machine Learning (ML) is evolving beyond analysis and model development and now can be used for operation. Creating a real ML model and applying it to the service requires more time for data collection, analysis, model tuning than for just model development. Production-level ML Systems require the ability to automate and manage complex ML workflows in the MLOps lifecycle, from model development/training, model tuning, model building/deployment, and model management.

Kubeflow is an open source Kubernetes-based machine learning platform supporting ML workflows. The platform was first developed and released by Google, Cisco, IBM, Red Hat, and other businesses as an open source project in 2018, followed by the official launch of its 1.0 version in March 2020. **Kubeflow** provides scalable ML workflows based on appropriate combinations of several Kubernetes-based open source software (Istio, Knative, Argo, etc.).

This document explains how to configure and utilize **Kubeflow** to support MLOps in SDS Cloud.

Architecture Diagram

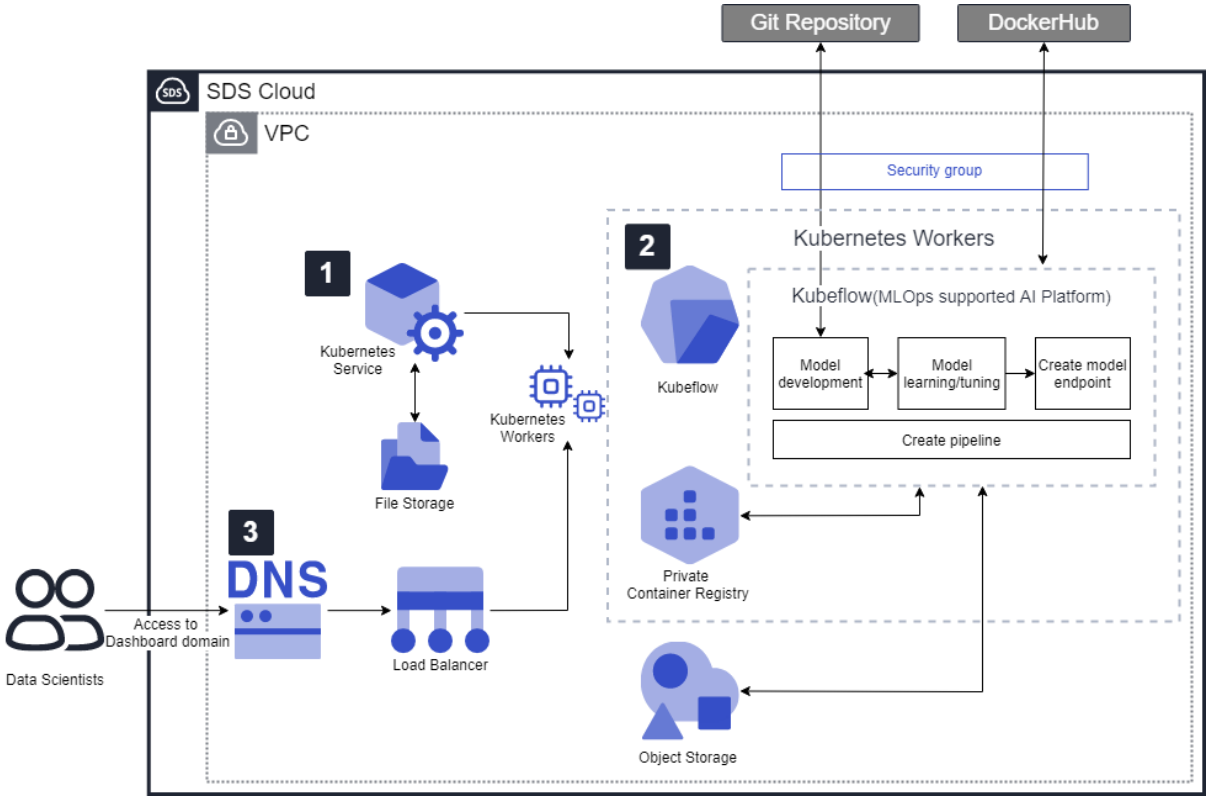

Figure 1. Kubeflow configuration that supports Kubernetes-based MLOps

1. A Kubernetes cluster is required to install **Kubeflow**. Users create a **Kubernetes Engine** first and the Persistent Volume (PV) of the Kubernetes Cluster is created from the **File Storage**.
2. You can deploy **Kubeflow** by selecting a Kubernetes cluster in your **VPC**. When the installation is complete, users can access the dashboard URL and use the ML workflow function provided by **Kubeflow**.
   A. You can use **Object Storage** to save analysis data sets and model files, and connect using S3 SDK in Jupyter notebook.
   B. Using Container Registry for container image build/deployment, you can deploy Private Container Registry in Kubernetes cluster or use external DockerHub..

   ※ When using services outside the user's **VPC** such as DockerHub and GitHub, firewall rules need to be added to the **Security Group**.
3. If you want to use the Kubeflow dashboard with a domain name, you can create a domain from **DNS.** Domain-based access is possible through the connection with the domain and Kubernetes workers in the **Load Balancer**.

Use Cases

A. Error determination system in production process using **Kubeflow**

Kubeflow provides excellent reusability, scalability, and stability based on Kubernetes. **Kubeflow** provides a Jupyter notebook-based model development environment and provides hyperparameter tuning and result validation, allowing performance comparison between multiple models.

In addition, it supports distributed learning using GPUs of ML Framework (Tensorflow, Pytorch, etc.) to shorten the model training time, and provides an Endpoint API that can be called by applications and service extensions when deploying models.

Using GPU-based high-performance infrastructure (GPU Direct RDMA) improves distributed learning performance by 1.5 times on average.

B. CI/CD for ML (model development and operation distribution system development)

Creating a recyclable workflow through the **Kubeflow** pipeline helps to configure CI/CD from model development to deployment, enabling automation within the same pipeline for relearning of the model with additional data.

Pre-requisites

A Kubernetes cluster and **File Storage** with higher than the minimum specifications are required to install **Kubeflow**.

Limitations

Kubeflow model deployment is currently only available within the same Kubernetes cluster.

The service is provided in a single cluster without separating the model development environment and the operating environment for inference service.

Considerations

You may consider using **Object Storage** for data store and model storage.

Configuring private container registry on Kubernetes cluster as container registry for container image storing or using external DockerHub can also be considered.

Kubeflow functions can be expanded by distributing Pipeline configuration open source tools such as Kale, Elyra, and Feast.

Related Products

- Kubernetes Engine
- File Storage
- Object Storage
- Load Balancer
- DNS