

FLScalize: Federated Learning Lifecycle Management Platform

Yang Semo



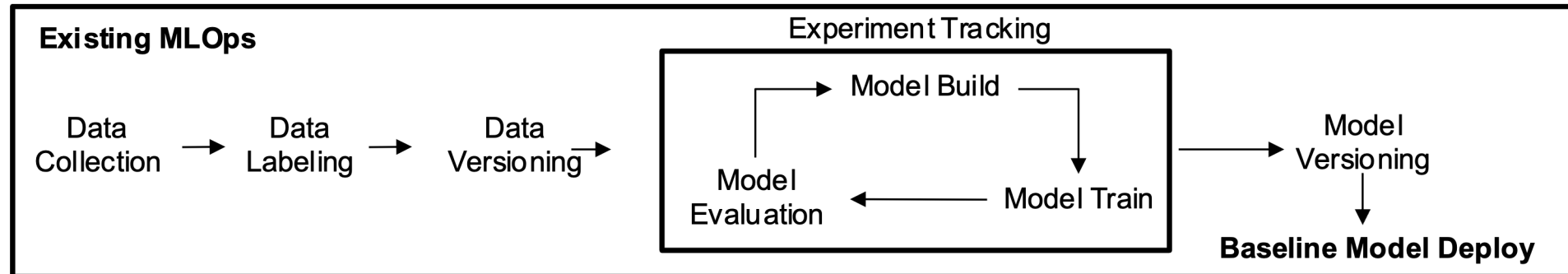
목차

- Introduction
- Related Research
- FLScalize System Design & Architecture
- Experimental Environment Setup
- Implementation and Result
- Conclusion and Future work
- FLScalize 실습

Introduction

■ 연합학습 수명주기 관리와 운영

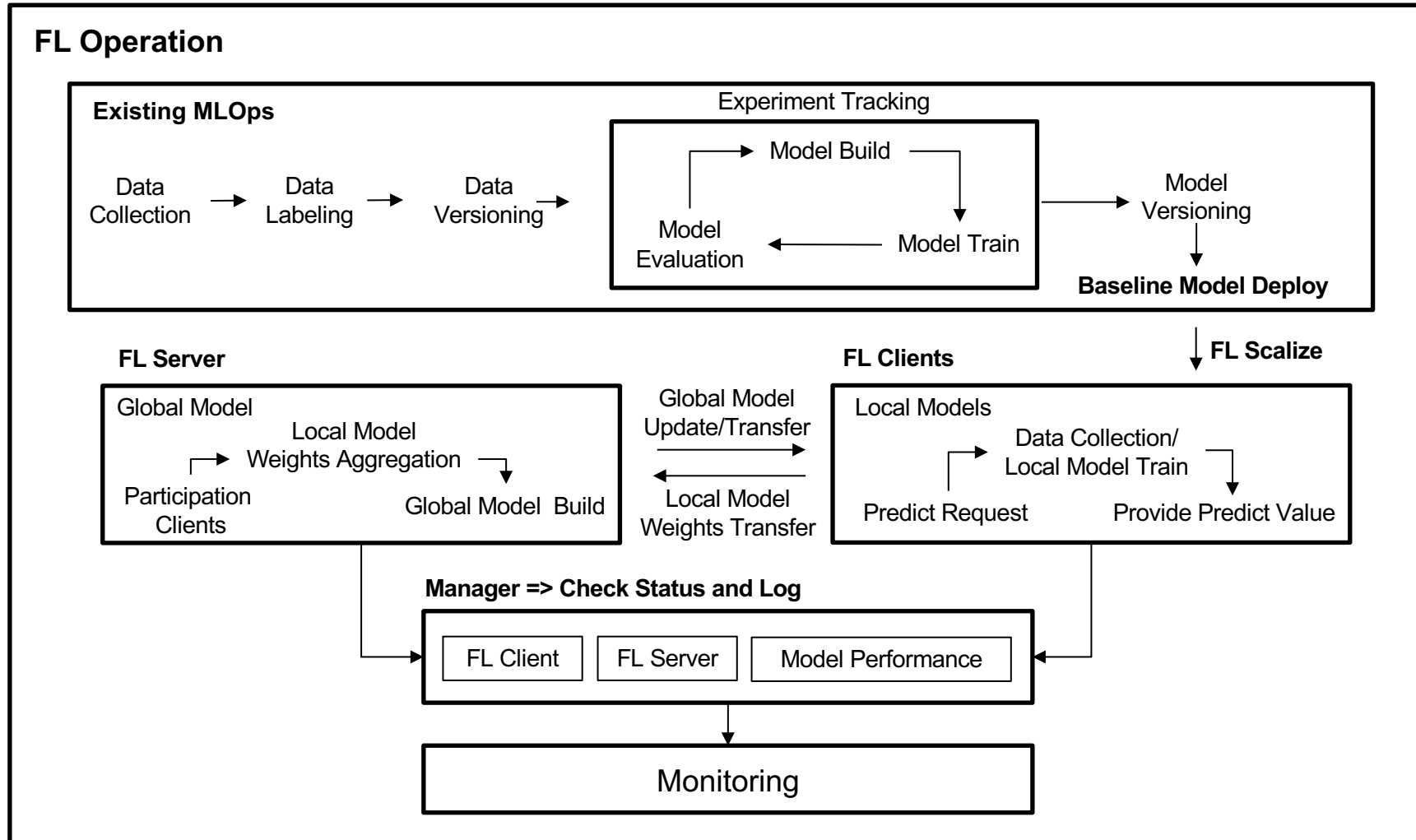
- 연합학습 구현을 위해 다양한 연합학습 프레임워크 등장
- 많은 연구에서 연합학습 성능 향상을 위한 알고리즘 연구 수행
- Real-World Project에 연합학습을 적용하여 수명주기 관리 및 운영할 수 있는 환경 필요
- 현재, AI 연구 분야에서는 MLOps를 활용하여 데이터 파이프라인, 모델 설계/학습, 배포 기능 제공



- 기존 MLOps는 수많은 Client를 보유하고 있는 연합학습 환경에 적합하지 않음
=> 기존 MLOps를 확장시킨 Federated Learning Operation (FedOps) 필요

Introduction

- 연합학습 수명주기 관리와 운영



Introduction

- FLScalize Contribution

- Easily application of data and models to the FL environment
- Manager component that checks the state of FL client and server
- Continuous integration/deployment/training
- Providing simulation environments for two categories of FL that occur in the real world (System/Data Heterogeneity)
- FL lifecycle management applicable to real projects

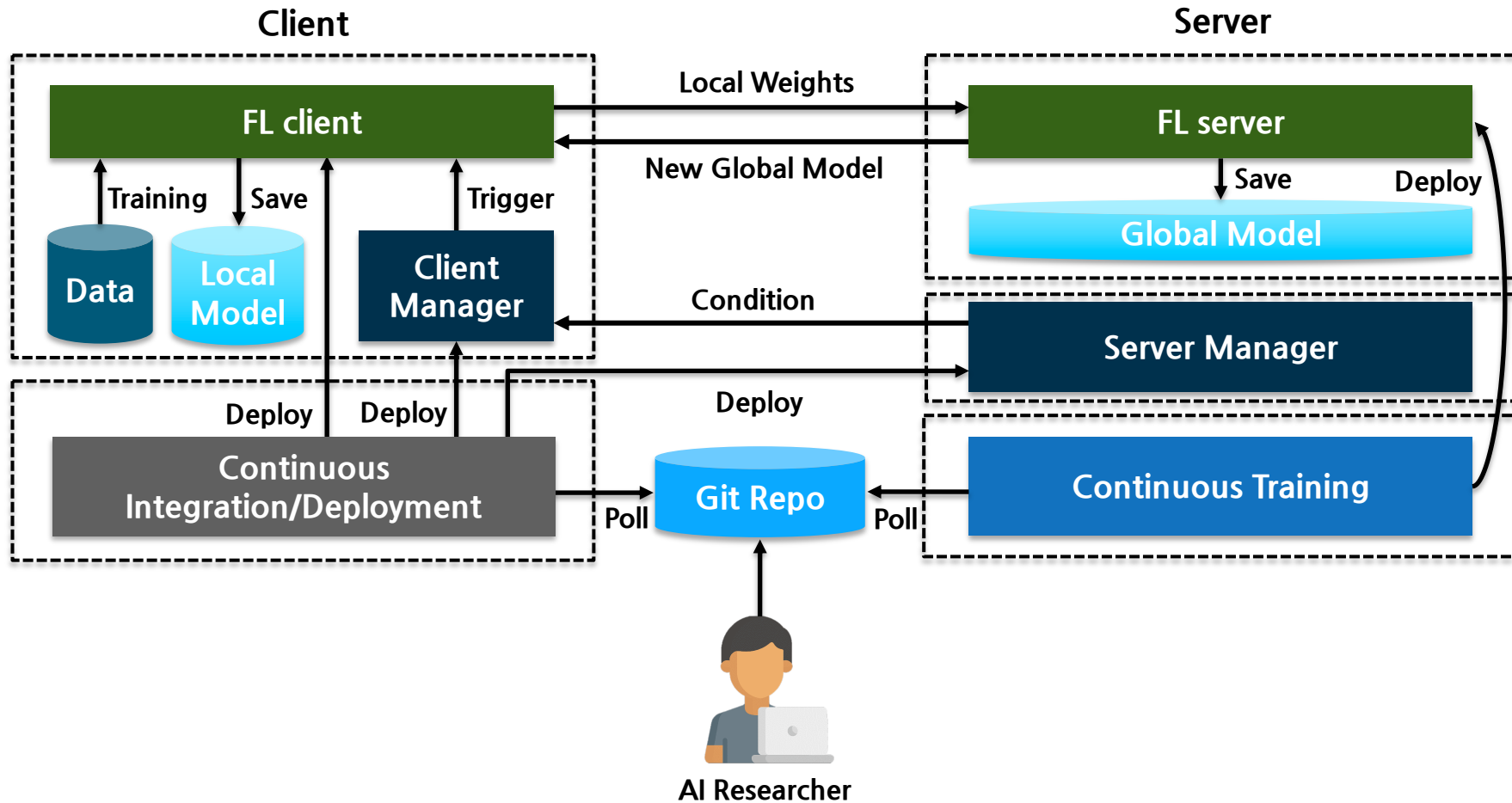
Related Research

- Comparison of FLScalize with Existing FL Frameworks

	TFF	FATE	FedScale	EasyFL	FLScalize
Simple application of data and model	○	○	○	○	✓
FL client/server management	X	X	X	X	✓
Heterogeneous system and data	○	○	○	✓	✓
Multi-client CI/CD/CT	X	○	X	X	✓
FL lifecycle management	X	○	X	○	✓

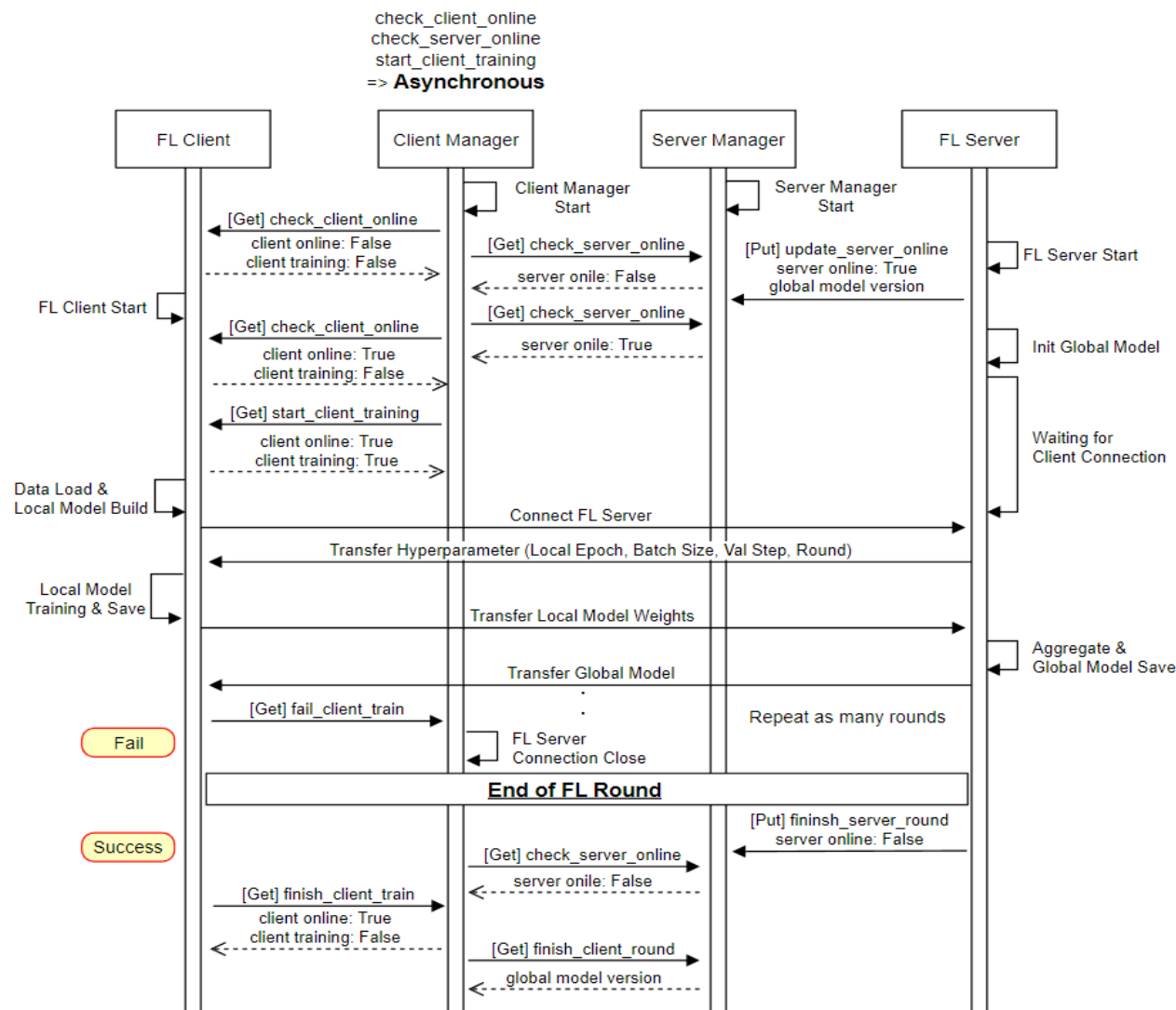
FLScalize System Design & Architecture

- FL Lifecycle Management



FLScalize System Design & Architecture

FL Lifecycle Process



Experimental Environment Setup

▪ FLScalize Environment

Server Name	CPU	GPU
D Server	Intel i9-9900KF (8core)	NVIDIA®2080Ti (2)
E Server	Intel Xeon Silver 4214R (12core)	NVIDIA®3090Ti (2)
X Server	Intel i9-10980XE (18core)	NVIDIA®3070Ti (2)

microk8s cluster

Dataset	Model	Sample
CIFAR-10	CNN	60,000
MNIST	ResNet50	70,000
FASHION MNIST	VGG16	70,000

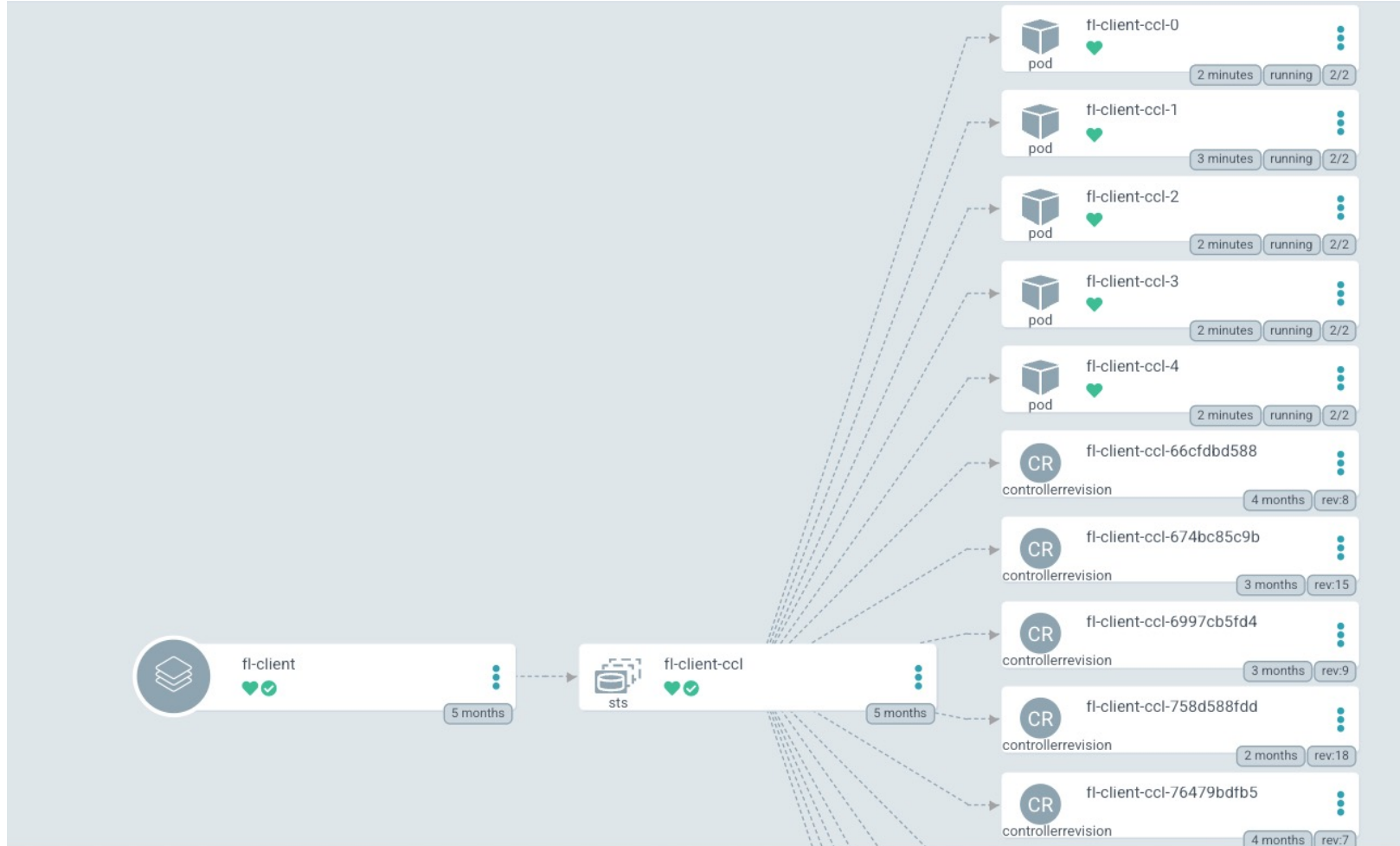
data/model

Non-IID	Description
Imbalanced Data	Each client is randomly distributed in different sizes
Skewed Data	Each client has only one class (or two/three classes)
Imbalanced /Skewed Data	Each client has only one class and is distributed in different sizes

data partition

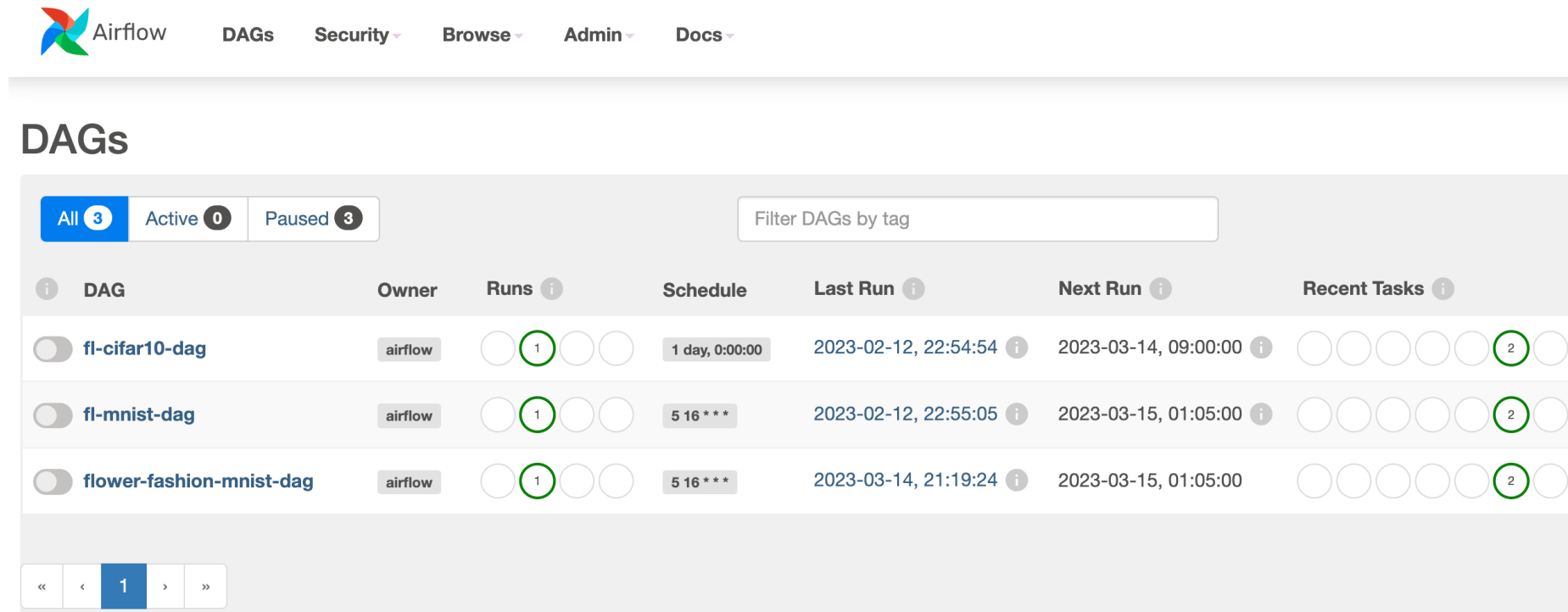
Implementation and Result

- Deploying and running five client pods using ArgoCD






























Implementation and Result

- Creating three FL tasks (CIFAR-10, MNIST, and FASHION MNIST) to support a workflow that creates an FL round



The screenshot shows the Apache Airflow web interface. At the top, there's a navigation bar with the Airflow logo and links for DAGs, Security, Browse, Admin, and Docs. Below this, the 'DAGs' section is active, showing a list of DAGs. The interface includes filters for 'All' (3), 'Active' (0), and 'Paused' (3) DAGs, along with a search bar for 'Filter DAGs by tag'. The DAGs are listed in a table with columns for DAG name, Owner, Runs, Schedule, Last Run, Next Run, and Recent Tasks. Three DAGs are visible: 'fl-cifar10-dag', 'fl-mnist-dag', and 'flower-fashion-mnist-dag'. Each DAG has a toggle switch, a status indicator (a green circle with a '1' for the first run), a schedule, the last run date and time, the next run date and time, and a series of task status indicators (circles). The first run of each DAG is highlighted with a green circle and the number '1', and the second run is highlighted with a green circle and the number '2'.

 DAG	Owner	Runs 	Schedule	Last Run 	Next Run 	Recent Tasks 
 fl-cifar10-dag	airflow	   	1 day, 0:00:00	2023-02-12, 22:54:54 	2023-03-14, 09:00:00 	              

Implementation and Result

■ System/Data Heterogeneity Result

Client Group	System	Each Round Time	Local Training Time	Total Time
<i>Group 1</i>	CPU 1, Memory 2 GB	66.7 s	48.2 s	1,346.4 s
<i>Group 2</i>	CPU 3, Memory 4 GB	60.5 s	44.3 s	1,224.1 s
<i>Group 3</i>	CPU 1, GPU 1, Memory 2 GB	61.1 s	44.7 s	1,243.4 s

Time Cost

Data Heterogeneity	CIFAR-10		FASHION MNIST		MNIST	
	Local Model	Global Model	Local Model	Global Model	Local Model	Global Model
<i>IID</i>	91.8%	91.4%	96.6%	89.8%	95.6%	91.7%
<i>Imbalanced</i>	91.3%	91.1%	84.2%	80.6%	94.9%	91.5%
<i>Skewed One Class</i>	84.2%	81.9%	14.7%	32.1%	26.8%	13.4%
<i>Skewed Two Class</i>	88.6%	88.3%	25.2%	12.7%	25.9%	25.2%
<i>Skewed Three Class</i>	90.4%	89.7%	59.3%	55.9%	38.9%	33.6%
<i>Imbalanced/ Skewed Three Class</i>	89.3%	88.5%	49.1%	47.3%	29.9%	25.7%

Accuracy

Conclusion and Future work

▪ FLScalize

- data와 model을 연합학습 환경에 간편하게 적용하고 다양한 FL Task 생성
- client/server 상태를 지속적으로 확인하고 관리하는 Manager Component와 Multi-Client의 Resource 구성 환경
- 지속적인 통합/배포 및 학습할 수 있는 Federated Learning Lifecycle Management
- 자신의 FL Task 실험을 위해 System/Data Heterogeneity Simulation 환경 제공

▪ FLScalize of the Future => FedOps

- microk8s 서버환경이 아닌 실제 여러 Client와 Server를 분리한 환경에서의 FL Lifecycle Management 지원
- 실제 디바이스(Mobile/IoT/PC)를 Client로 적용하여 Real Multi FL Client 관리/모니터링/운영 서비스 제공
=> FedOps Platform으로 확장

FLScalize 실습

- docker 설치 (<https://www.docker.com/>)
- anaconda prompt 실행

- FedOps Git clone

-mac: `git clone https://github.com/Kwangkee/Gachon.git && mv Gachon/pratice/FedOps . && rm -rf Gachon`

-window: `git clone https://github.com/Kwangkee/Gachon.git && move Gachon\\practice\\FedOps . && del Gachon`

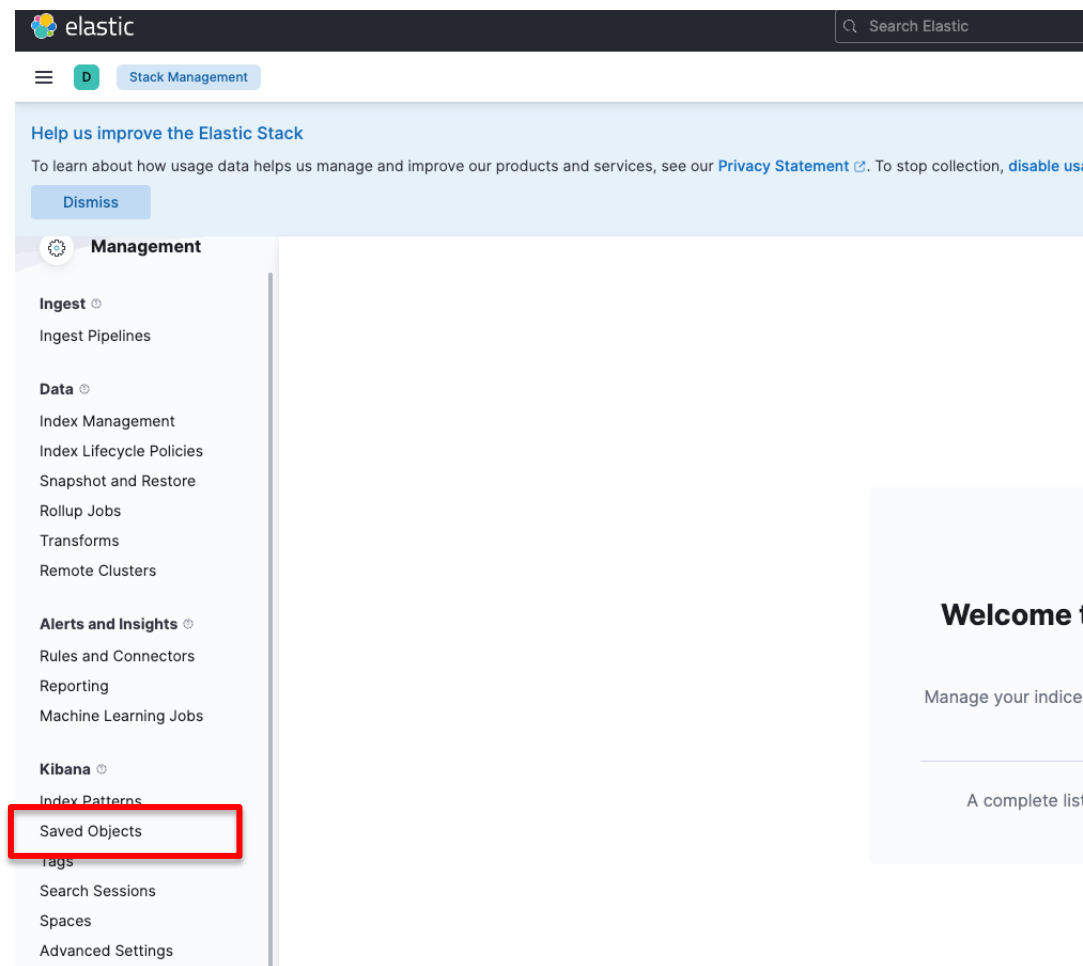
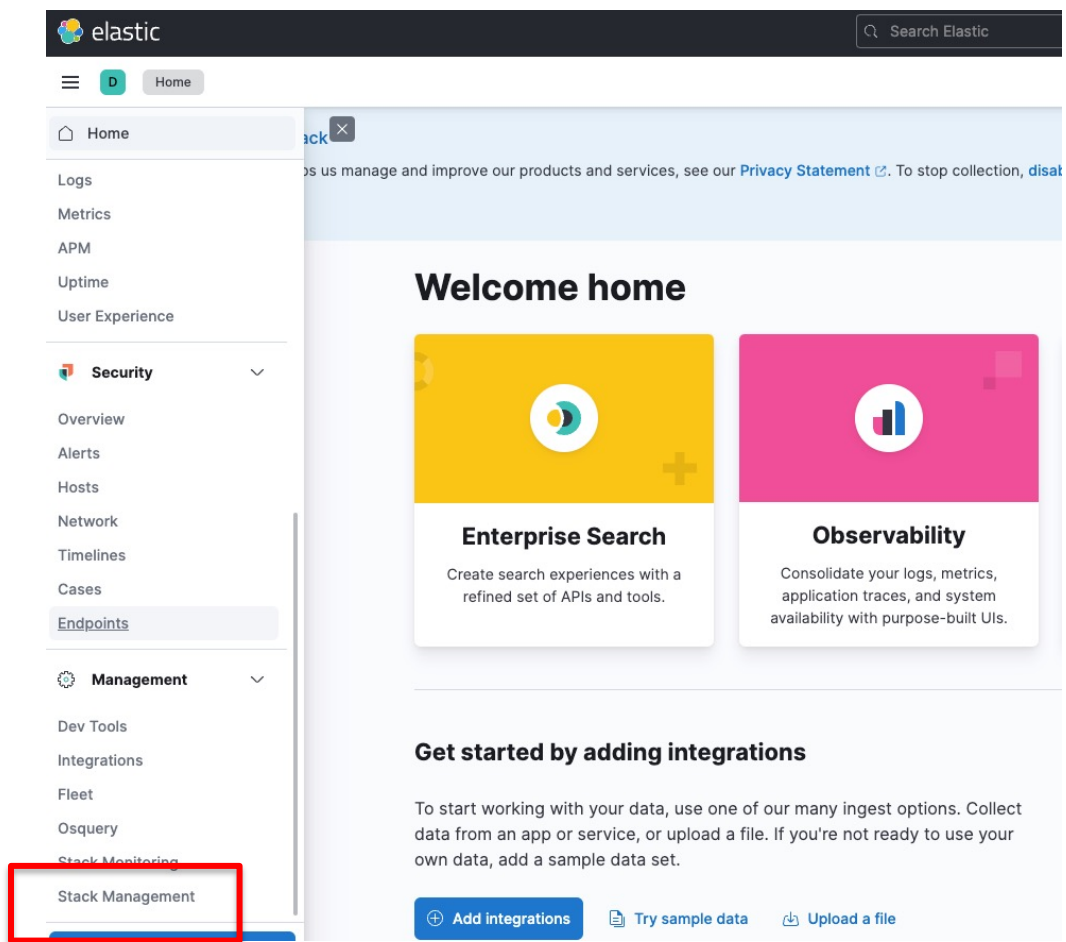
- `conda create -n fedops python=3.9`
- `conda activate fedops`
- `git clone`해서 받은 폴더로 이동 => `cd FedOps`
- `pip install -r requirements.txt`
- `conda install tensorflow==2.10.0 keras==2.10.0`
- `pip install numpy --upgrade`

FLScalize 실습

- 1. Server Manager 실행
 - cd FedOps/server_manager
 - conda activate fedops
 - python app.py
- 2. FL Server 실행
 - 또 다른 shell 창 실행
 - cd FedOps/server
 - conda activate fedops
 - python app.py
- 3. Client Docker Compose 실행
 - docker 실행
 - 또 다른 shell 창 실행
 - cd FedOps/server
 - cd git_clone경로/FedOps/cross_silo
 - sh monitoring.sh

FLScalize 실습

■ Client Monitoring Dashboard 설정



FLScalize 실습

■ Client Monitoring Dashboard 설정

The screenshot shows the FLScalize interface with a modal dialog titled "Import saved objects". The dialog has a close button (X) in the top right corner. It contains a section "Select a file to import" with a file input field and a red rectangular box highlighting the "Import" button. Below this is the "Import options" section, which includes three radio button options: "Check for existing objects" (selected), "Automatically overwrite conflicts", and "Request action on conflict". At the bottom of the dialog are "Cancel" and "Import" buttons. The background shows the "Saved Objects" page with a table listing objects like "Advanced Settings [7.17.3]", "FedOps Client Dashboard", and "fl-*".

- Fedops/cross_silo 디렉토리에 있는 파일 import
“FedOps_Client_Dashboard_V3.ndjson”

FLScalize 실습

■ Client Monitoring Dashboard 설정

