가천대학교
Gachon University

연합학습 시스템에서의 MLOps 구현 방안 연구

Paper Review (2022.09.23)

202240132 양세모

Cognitive Computing Lab

목차

Paper Info / Abstract

Title: 연합학습 시스템에서의 MLOps 구현 방안 연구

- Journal Name: Journal of Internet Computing and Services, KCI, 2022

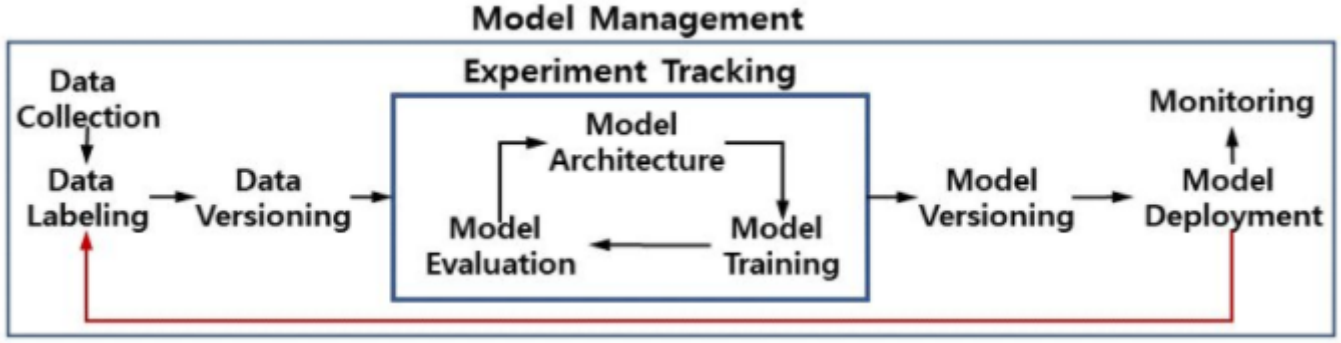
- Abstract

  - 연합학습은 학습 데이터의 전송없이 로컬(디바이스) 환경에서 모델의 학습을 수행할 수 있는 방법
  - 데이터가 이동하지 않기 때문에 개인정보 유출에 자유로운 학습 방법으로 각광 받고 있음
  - 연합학습을 사용하는 시스템의 개발과 운영을 위한 시스템 설계의 구체적인 연구가 부족
  - 본 연구에서는 연합학습을 실제 프로젝트에 적용하여 연합학습의 수명주기를 관리하는 **코드/모델 버전 관리**, **디바이스 성능/상태 모니터링**, **서버-클라이언트 학습 스케줄링**을 할 수 있는 FedMLOps 시스템 설계 제안

MLOps(Machine Learning Operations) 란?

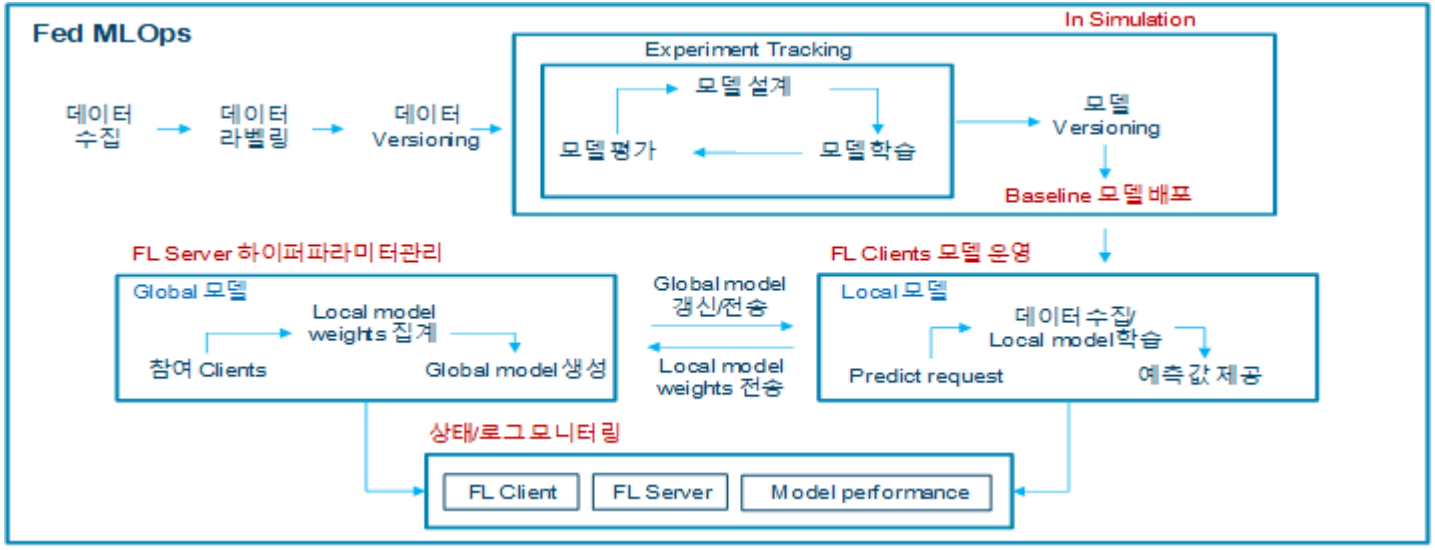
▪ 딥러닝/머신러닝 활용 분야가 증가하면서 DevOps(Development Operations)를 기반으로 지속적인 모델 설계, 학습, 응용, 통합/배포, 모니터링 절차를 자동화

- 효율성: 모델을 빨리 배포, 양질의 ML 모델 제공
- 확장성: 여러 모델을 감독, 제어, 관리, 모니터링하고 지속적인 통합/배포
- 리스크 완화: DL/ML 모델 검토 및 검사 ⇒ 투명성 강화



FedMLOps(Federated MLOps) 란?

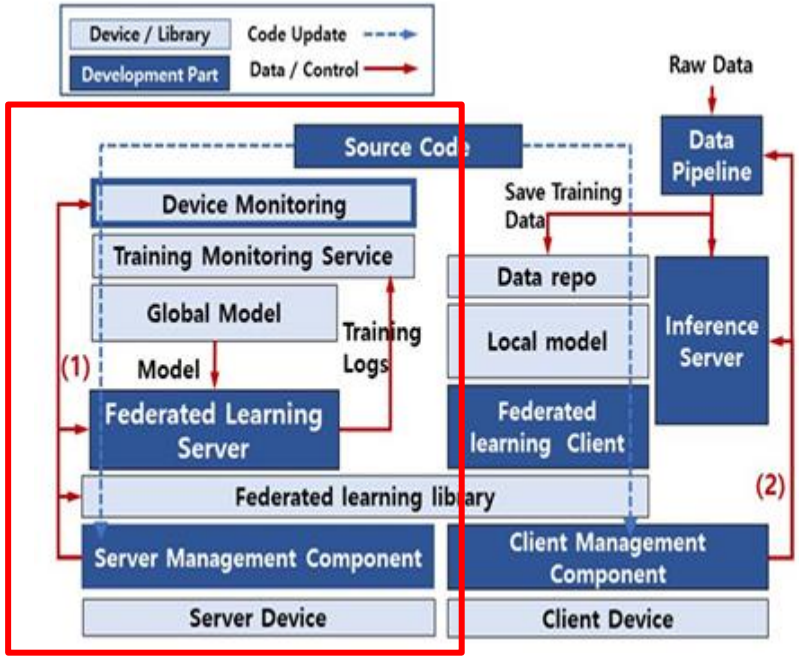
- MLOps는 중앙화 되어있는 머신러닝 시스템에 적합하여 연합학습에 적용하기 어려움
  - 기존 MLOps 개념에 연합학습을 활용하여 프로젝트를 진행할 수 있는 FedMLOps 개념 도입
  - 연합학습을 구현하기 위한 라이브러리(TFF, PySyft, Flower 등)를 활용하여 Local/Global Model의 코드 버전, 학습 스케쥴 관리, Client/Server 상태 및 성능 모니터링



제안하는 시스템은 서버-클라이언트 구조를 가지며 API로 통신하는 컴포넌트 구성

- Server Device

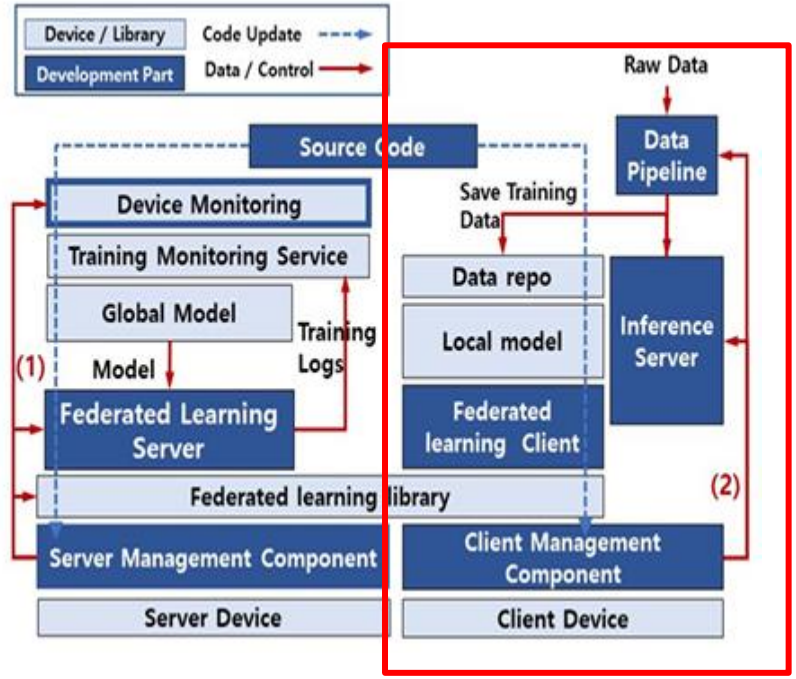
  - Server Management Component: 연합학습 서버를 실행시켜 Aggregation을 시작시키는 역할과 서버의 상태를 클라이언트에게 알림
  - Federated Learning Server: 연합학습에 참여하는 Client의 Weights를 Aggregation 하는 역할
  - Device Monitoring: 서버 error log와 리소스 사용 상황, 서버-클라이언트 관리 컴포넌트의 상태를 모니터링
  - Training Monitoring Service: 글로벌 모델의 학습 상황을 모니터링, 모델 개발자는 이를 보고 모델을 개선



제안하는 시스템은 서버-클라이언트 구조를 가지며 API로 통신하는 컴포넌트 구성

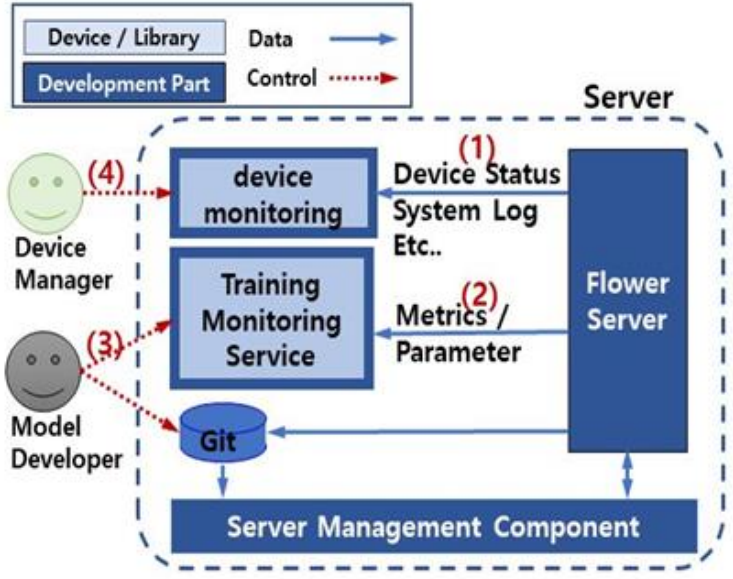
- Client Device

  - Client Management Component:
    - 코드와 모델 버전 관리하여 최신 상태를 유지
    - 지속적으로 클라이언트의 상태를 확인
    - Client가 학습에 참여 할 수 있게 실행 트리거 발생
  - Federated Learning Client:
    - FL Server 연합학습 라운드에 참여
    - 최신 Global Model 기반 Local Model 생성
  - Inference Server:
    - Client에서 발생하는 Data 관리
    - Local Model 기반의 예측 수행 및 예측값 저장



디바이스 매니저(전체 시스템 개발을 담당)와 모델 개발자 간의 역할 구분

- Device Manager
  - 연합학습 서버에서의 디바이스 상태 로그 및 시스템 로그 등을 모니터링
- Model Developer
  - 트레이닝 모니터링: 학습 상황이나 성능을 모니터링
  - 트레이닝 모니터링 서비스를 보고 모델을 조정하여 Github에 올려 재배포

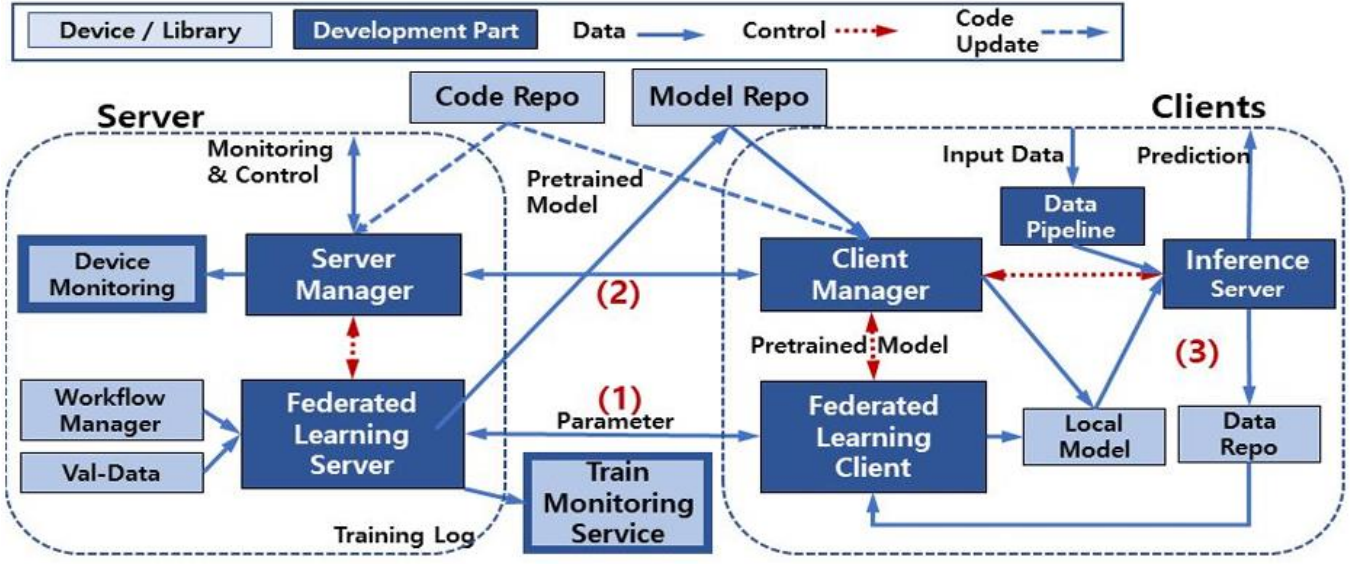


시스템 컴포넌트 간의 관계

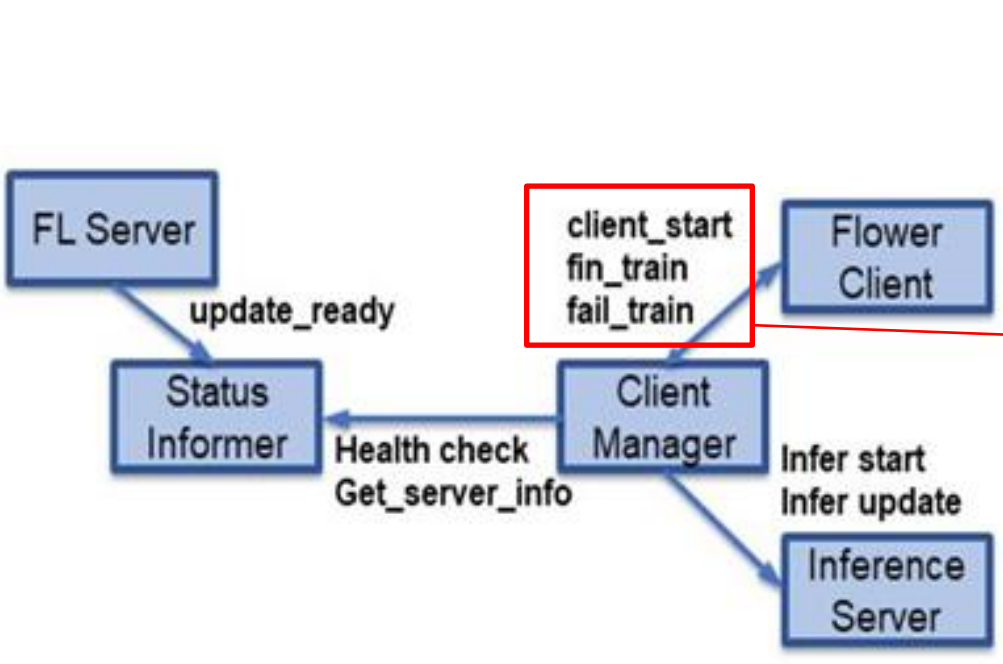
(1) 연합학습 클라이언트와 서버에서는 gRPC (google's Remote Procedure Calls: 구글의 원격 프로시저 호출)를 통해 통신

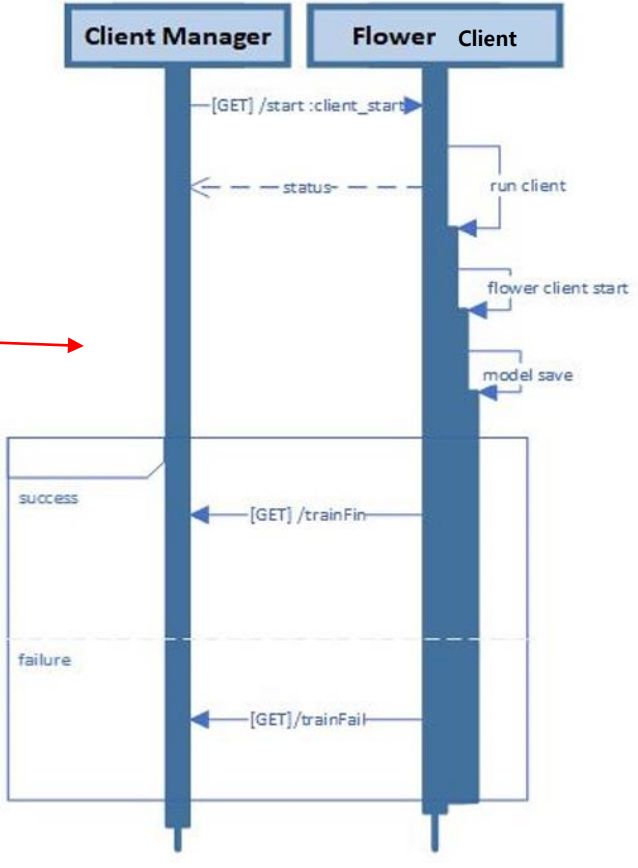
(2) 서버/클라이언트의 관리 컴포넌트 사이의 통신은 RESTful API를 사용

(3) 인퍼런스 서버는 클라이언트에서 외부의 데이터를 입력받아 로컬 모델을 통해 예측값을 제공하고 해당 데이터와 정답을 저장



각 컴포넌트는 API 통신으로 서로의 상태 확인



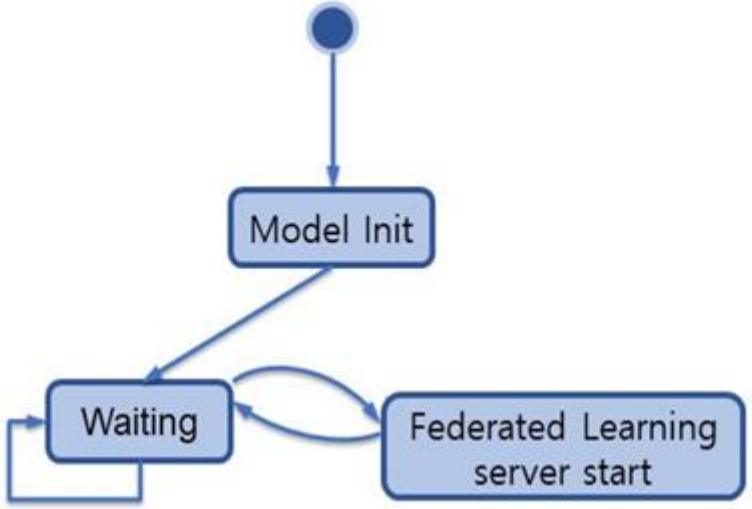


서버 상태 전이도

- Model Init
  - 모델 저장소에 저장된 모델이 없다면 서버에서 초기 글로벌 모델 생성

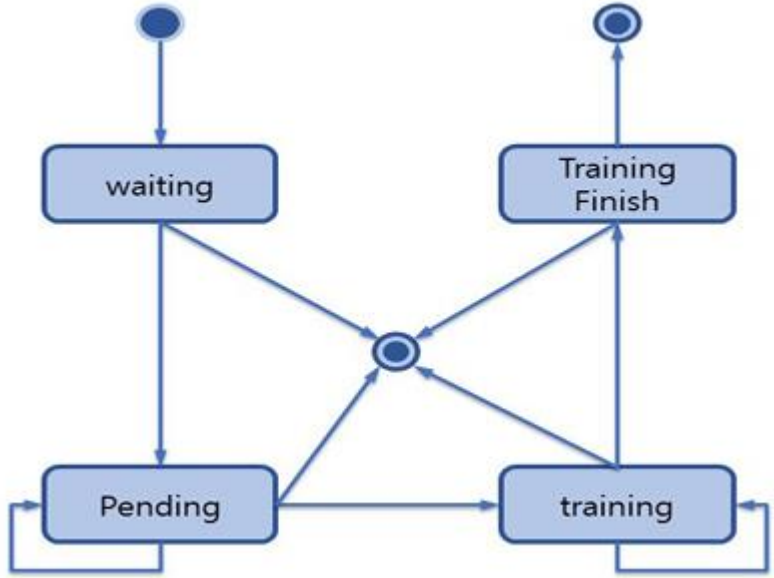
- Waiting
  - 워크플로우 매니저에서 서버를 수행하기 전까지 대기

- Federated Learning server start
  - 연합학습 서버를 실행
  - 기존 글로벌 모델이 있다면 최신 글로벌 모델 Load
  - Client가 연합학습 라운드에 참여할 때까지 대기



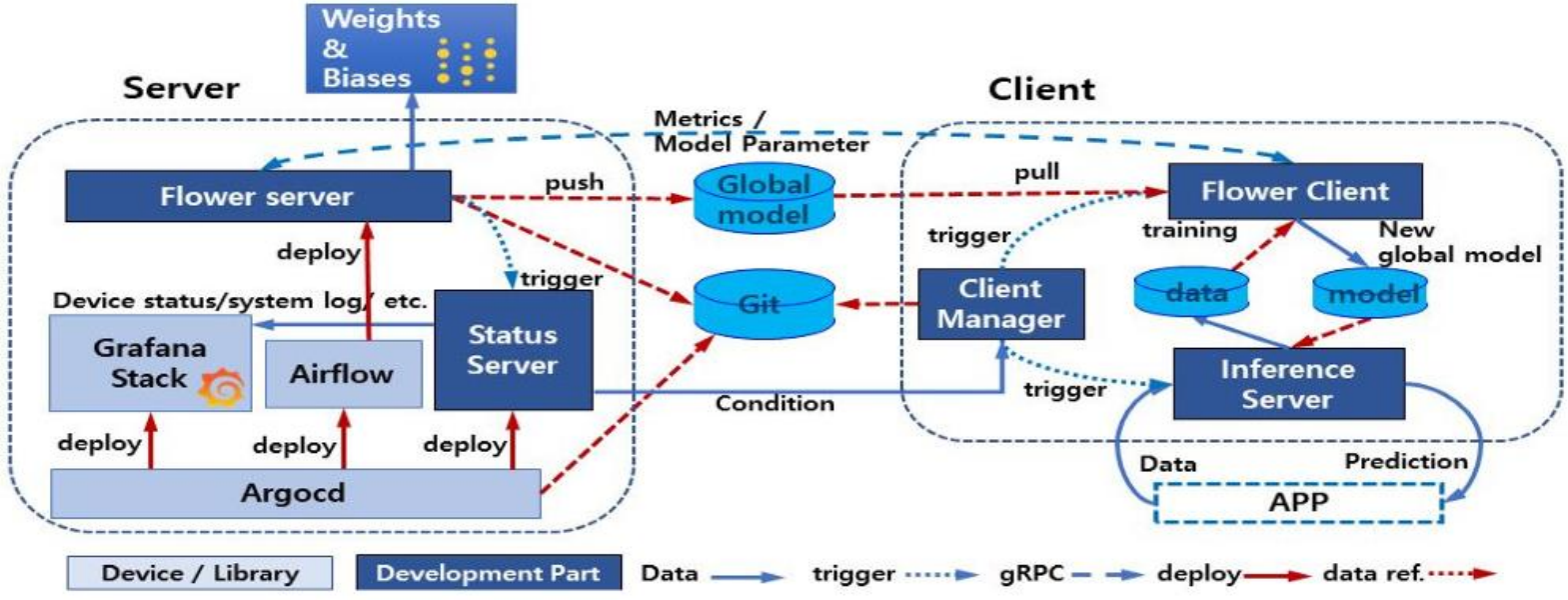
서버 상태 전이도

- watitng
  - 클라이언트 매니저의 train start 명령 대기
- pending
  - 연합학습 라운드에 접속하고 다른 클라이언트의 접속을 대기
- training
  - 연합학습 라운드에 따른 학습 수행
- training finish
  - 연합학습을 완료하고 로컬 모델 저장



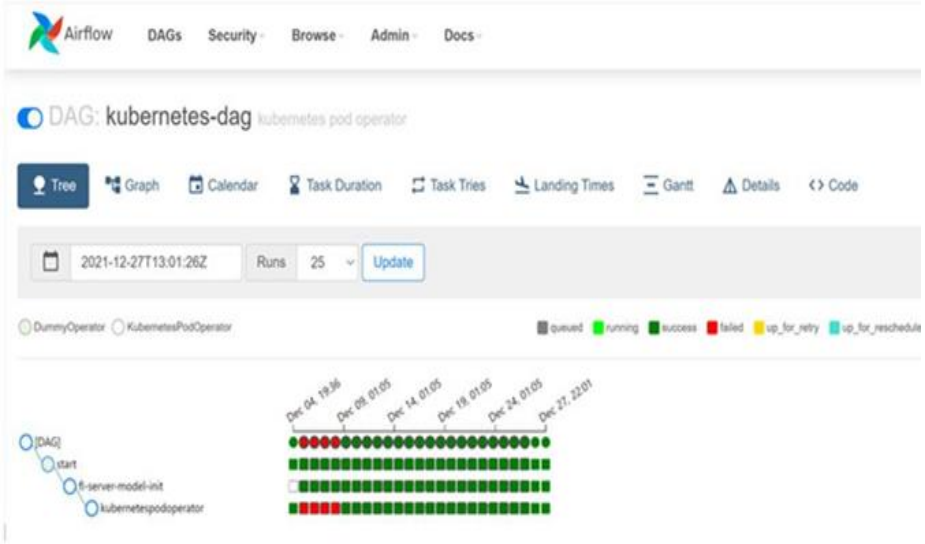
Kubernetes 환경에서의 시스템 구현

- 전체 연구 절차
  1) 코드 저장소 준비, 모델 저장소 준비
  2) Workflow 관리 프로그램 배포
  3) Server 배포
  4) Client 배포
  5) 학습 진행
  6) 학습결과 확인 및 디바이스 상태 확인



Result

Airflow를 통해 학습이 정상적으로 수행되고 있는 것을 확인



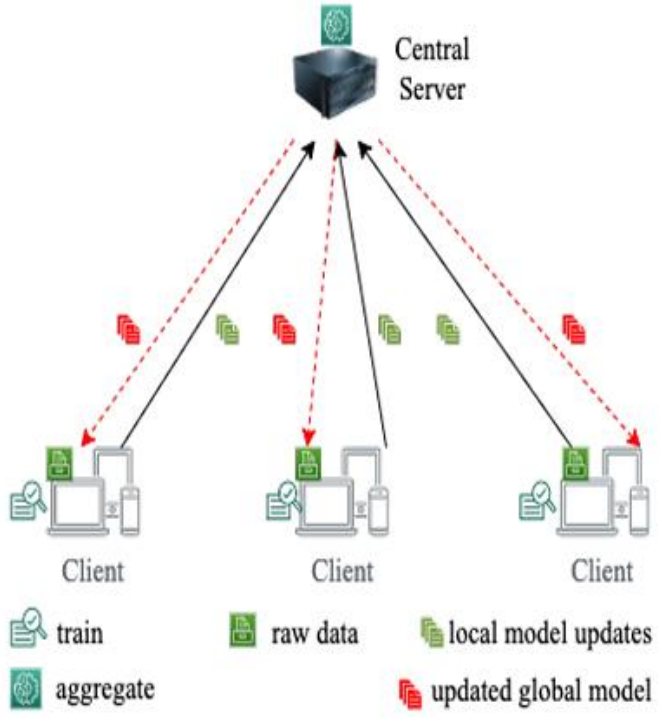
Mnist 데이터셋을 라운드 5로 학습한 결과 모니터링 (Weights & Biases)

Conclusion

- 제안한 시스템은 연합학습 구현체와 독립적으로 설계
- 다양한 하드웨어에서 원활히 실행 가능하도록 하기 위해 컨테이너 기반 구현
- 서버와 클라이언트에 관리를 위한 컴포넌트를 추가하여 연합학습 응용프로그램 외적으로 필요한 서비스를 제공
- 해당 시스템의 컴포넌트들은 서로 API를 통한 통신을 이용하고 마이크로 서비스로 분할하여 각 컴포넌트를 자유롭게 개발 가능
- 추후 많은 개발자가 연합학습을 응용하여 프로젝트를 진행하고자 할 때 시스템 구성의 참조자료로써 사용할 수 있을 것으로 기대

연합학습과 Blockchain 기술 융합 필요성

- Single point of failure: 중앙서버 의존형 연합학습의 문제
  - 기대효과: 정보보호, 데이터 무결성, 추적성 개선
- Malicious clients and false data: 악의적 참여자 및 가짜 데이터 문제
  - 기대효과: 학습/인식 정확도, 수렴 성능 향상
- The lack of incentives:
  - 극복 방안: 데이터 생산자/제공자 (원격임상시험 대상자)에게 블록체인 융합 보상/인센티브 제공
  - 기대효과: 더 많은/성실한 참여를 위한 동기부여. 보상을 통한 자기주도적 학습환경 제공



기존 연합학습 구조

BCFL 구조

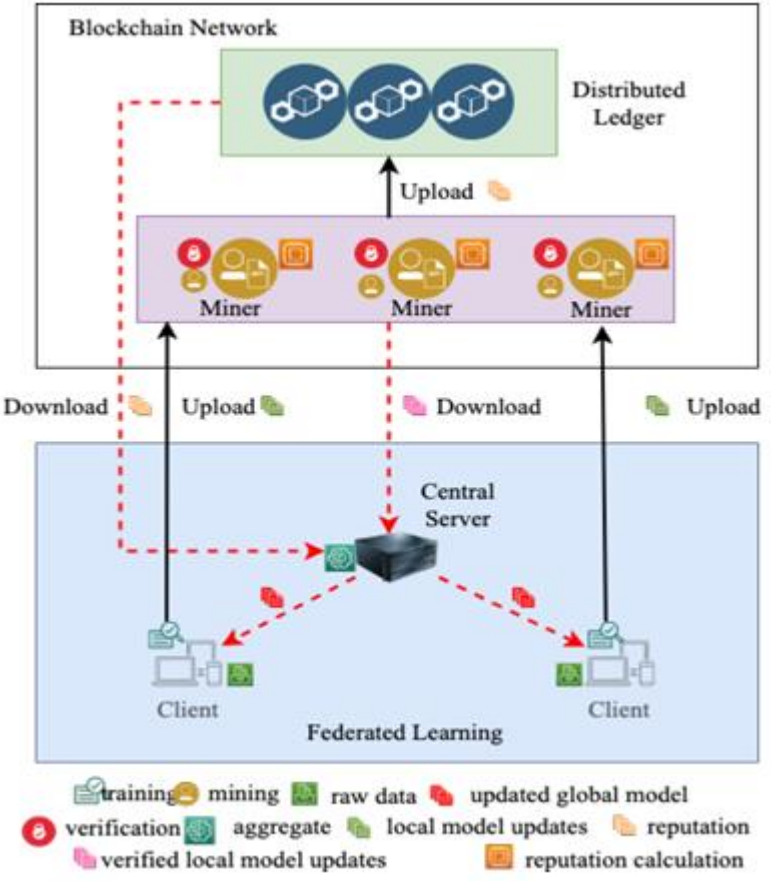
1. Clients train models locally and upload the local model updates to blockchain.

2. Miners verify the local model updates, then generate reputation opinions for the clients.

3. Miners compete to generate new block which contains the reputation related data, and the new block will be added into the distributed ledger.

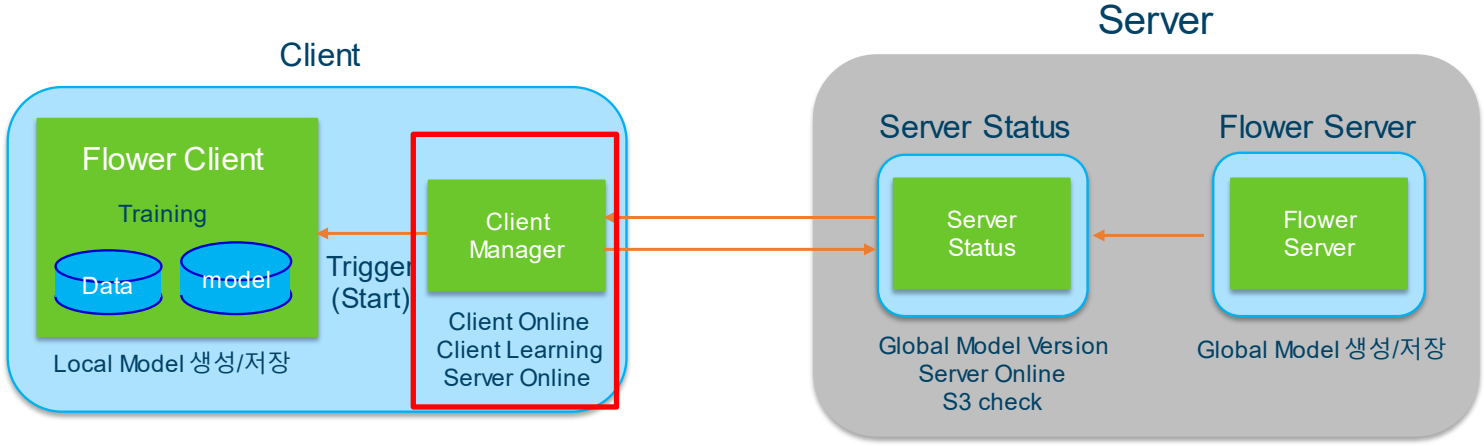
4. Aggregator collects the verified updates and then execute the global model aggregation algorithm.

5. Rewards and penalties are depended on the reputation opinions of clients.



BCFL 적용을 위한 추가/보완 사항

- Blockchain 기반의 Reward or Penalty 기준 **Client Selection**
  - Client Manager의 기능 확장을 통한 연합학습 참여 가능 Client Selection
    => Client의 데이터 상태, Loss 수렴 정도, 통신 상태 등 고려
    => Global Model의 성능을 높이기 위해 우수한 Client의 Local Model만을 Aggregation



감사합니다