

Kafka-ML: connecting the data stream with ML/AI frameworks

발표자: 윤요섭

Contents

01 Introduce

02 Background

03 ML/AI Pipeline in Kafka-ML

04 Kafka-ML Architecture

05 Kafka-ML Validation

01

Introduce

- 논문 선정 배경
- Abstract
- Keywords

논문 선정 배경

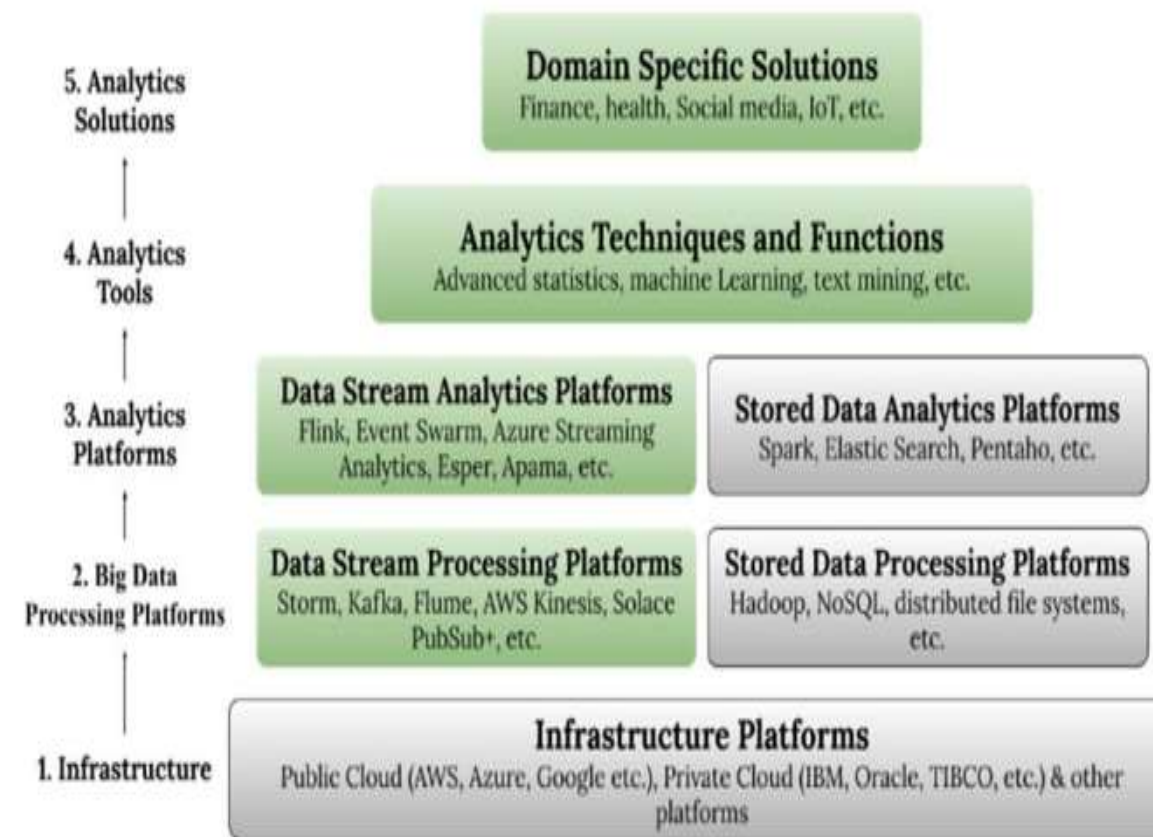
✓ Trend의 변화

1. Hadoop의 등장 이후, 빅데이터가 각광을 받기 시작
2. 기업들은 빅데이터의 가치를 인식하고 대규모 데이터 플랫폼을 구축하는데 주력



1. Kafka와 같은 실시간 데이터 처리 기술이 부상하면서 대량의 데이터를 실시간으로 처리하여 실시간 의사결정 및 개인화된 서비스 제공에 활용 Needs 증가
2. 기존 빅데이터 환경을 넘어 실시간성과 개인화에 중점을 두는 새로운 데이터 처리 패러다임의 등장

✓ The real-time analytics stack



real-time data로 수집 및 처리를 넘어서 AI의 데이터 학습 및 추론에도 real-time data를 활용하는 방법은 없을까?



“실시간 데이터 수집,처리를 넘어 학습 및 추론까지 하는 논문 탐색”

Kafka-ML:
connecting the data stream with ML/AI frameworks

Abstract

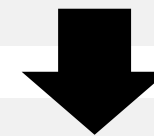
✓ From static data To continuous data streams

“데이터가 최신일수록 가치가 높아집니다. Doordash 및 Uber와 같은 데이터 중심 기업은 실시간 분석을 바탕으로 업계를 뒤흔드는 비즈니스를 구축함으로써 이를 입증했습니다. 다른 모든 기업은 이제 실시간 데이터를 활용하여 즉각적이고 개인화된 고객 서비스를 제공하고, 운영 의사 결정을 자동화하거나, ML 모델에 최신 데이터를 공급해야 한다는 압박감을 느끼고 있습니다.”

- 2022.01.10 Dhruba Borthakur(Rockset의 공동 창립자이자 CTO)

✓ Introduce Kafka-ML

However, most of the ML/AI Frameworks are not fully prepared this revolution



Kafka-ML: an open-source framework that enables the management of TensorFlow ML/AI pipelines through data streams(Apach Kafka)



Kafka-ML: fully managed through containerization technologies, ensure its portability and easy distribution and other features (HA, fault-tolerance), (no) utilization of data storage and file systems.

Keywords

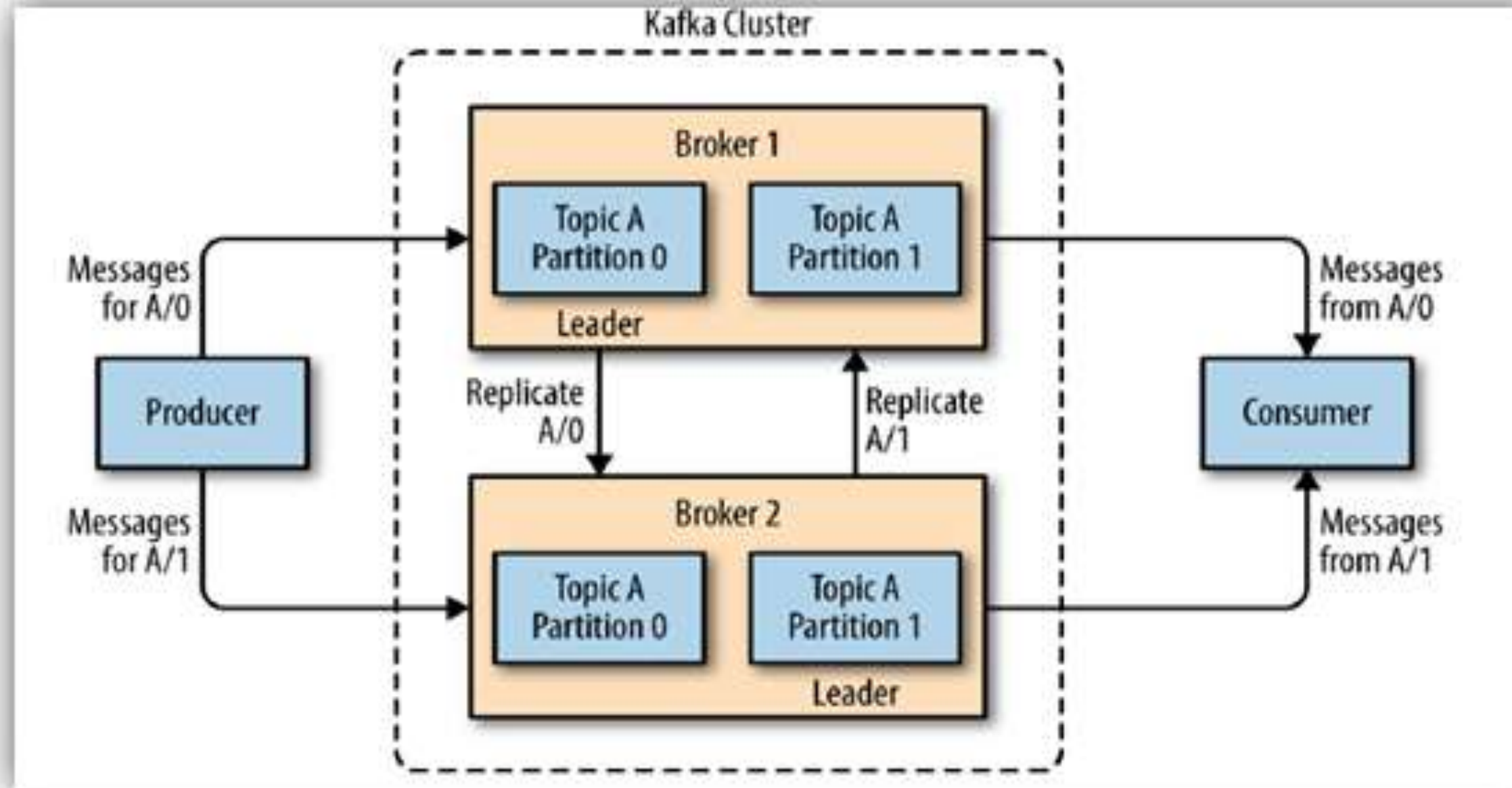
- ✓ Scalability
- ✓ High Availability
- ✓ Load Balancing
- ✓ RESTful API
- ✓ Docker
- ✓ Fault-tolerance
- ✓ Kafka
- ✓ Front-end / Back-end
- ✓ Tensorflow
- ✓ Kubernetes

02

Background

- Apache Kafka
- Tensorflow
- Docker
- Kubernetes

Apache Kafka



✓ What is Kafka?

“대규모 데이터 스트림을 처리하고 전달하기 위한 분산형 이벤트 스트리밍 플랫폼 ”

✓ Why Kafka?

- 높은 처리량과 낮은 지연시간
- 높은 확장성
- 고가용성
- 내구성

Apache Kafka Keyword

Keyword	Discription
Topic	데이터를 Pub/Sub할 대상 주제 (해당 토픽으로 데이터가 전송되고 소비됨)
Broker	Kafka가 사용할 서버. 보통 여러 대로 구성하여 Broker 클러스터로 구성
Producer	데이터를 생성하고 Kafka로 전송
Consumer	Kafka에서 데이터를 읽어옴
Partition	데이터가 저장될 장소. 토픽이 생성되고 Producer가 데이터를 전송하면 토픽 안의 파티션에 데이터가 전송되어 저장됨(이진 바이트 배열)
Zookeeper	분산 시스템의 조정 및 구성 관리를 위해 사용
Replication	고가용성을 위해 데이터를 여러 브로커에 복제하여 장애에 대비할 수 있음

Tensorflow



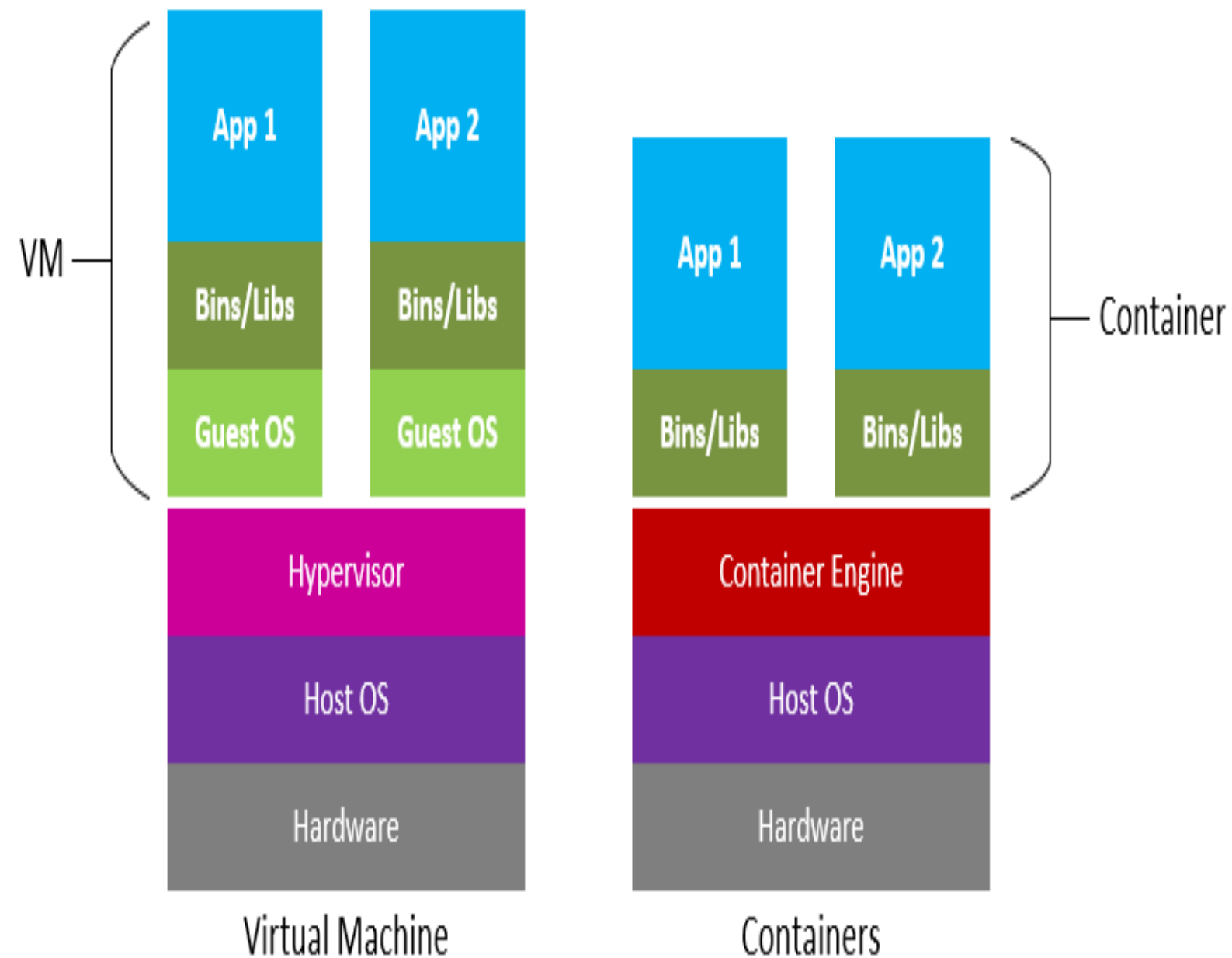
TensorFlow

✓ What is TensorFlow?

TensorFlow는 저수준 기능과 고수준 작업을 갖춘 딥러닝 도구상자로, 데이터플로우 프로그래밍 모델을 위해 설계됨

인공 신경망 같은 기계 학습 응용프로그램 및 딥러닝에도 사용되며, 머신러닝 및 딥러닝 프로그램을 쉽게 구현할 수 있도록 기능을 제공

Docker



✓ What is Docker?

애플리케이션의 컨테이너 이미지를 쉽게 생성할 수 있는 도구 세트를 제공

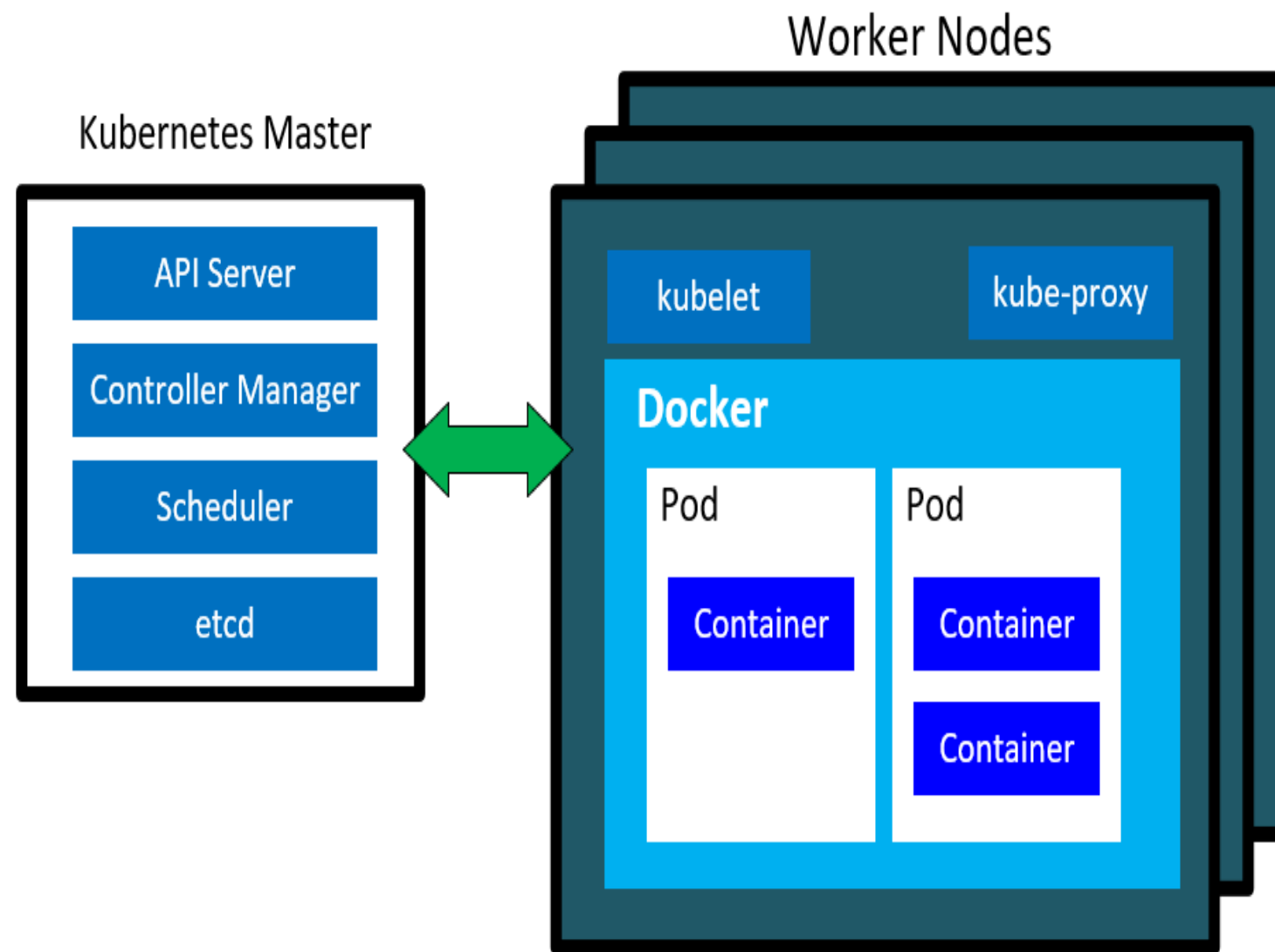
✓ Docker images vs Docker Containers

Docker images: 템플릿

Docker Container: Docker Image의 실행중인 인스턴스

Dockerfile이라는 파일에 명령어 목록 지정 후, Docker builder가 지정된 파일을 가져와 이미지로 패키징하여 컨테이너 레지스트리에 push 할 수 있음

Kubernetes



✓ What is Kubernetes

오픈 소스 컨테이너 관리 플랫폼

Kubernetes를 사용하면 Docker 엔진을 활용하여 컨테이너 그룹으로 애플리케이션을 구성하고, 원하는 대로 애플리케이션을 실행하도록 관리

➡ 애플리케이션이 요청한대로 지속적으로 실행되도록 관리해줌

ML/AI Pipeline in Kafka-ML

- Designing and defining ML models
- Creating a configuration
- Deploying the configuration for training
- Ingesting the deployment with stream data
- Deploying trained models for inference
- Ingesting the trained model to make predictions

ML/AI pipeline in Kafka-ML

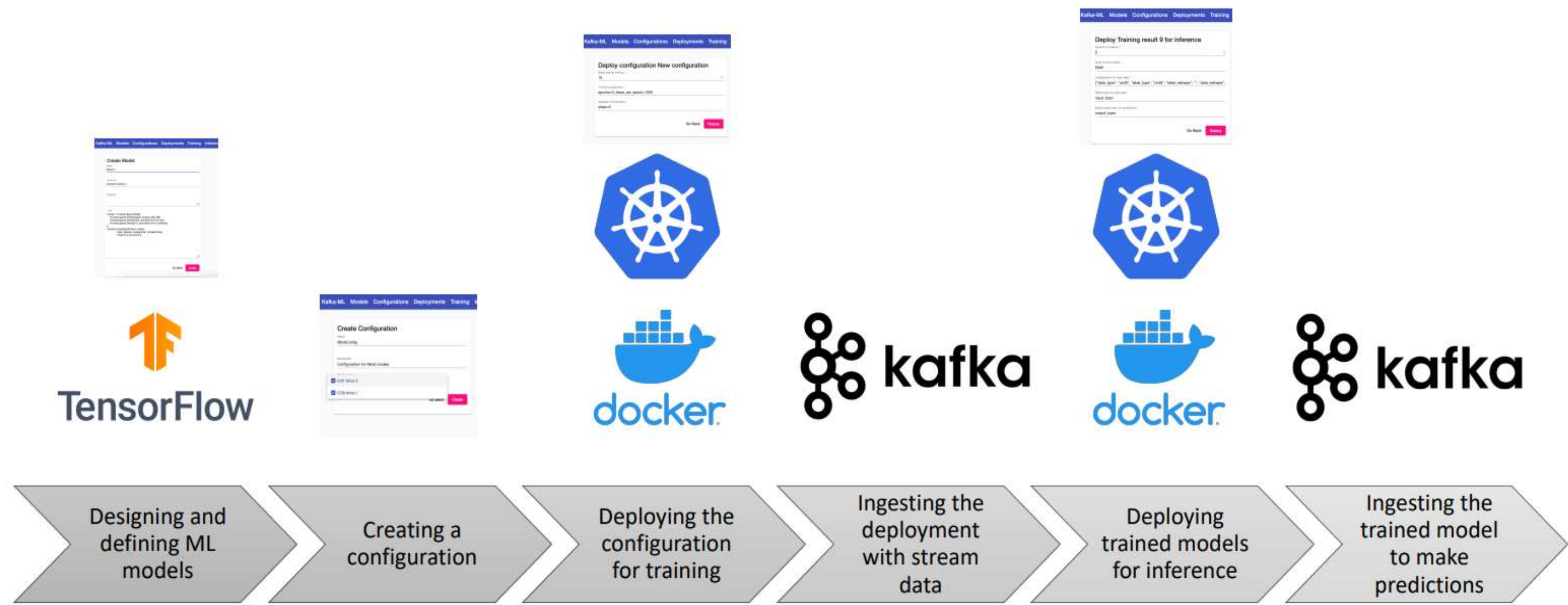


Fig. 1. ML/AI pipeline in Kafka-ML

ML/AI pipeline in Kafka-ML

Step	Description
Designing and defining ML models	Kafka-ML에 활용할 ML 모델 정의
Creating a configuration	Kakfa-ML을 훈련을 위한 설정(훈련에 사용할 모델 개수 등)
Deploying the configuration for training	학습을 위한 Batch Size, training and validation parameter 등 설정 배포
Ingesting the deployment with stream data	훈련을 위해 Stream Data를 파이프라인을 통해 전송
Deploying trained models for inference	훈련 및 평가가 끝나면 훈련 모델과 손실 및 정확도 등의 측정항목을 Kafka-ML 아키텍처에 제출하며, 추론을 위해 모델을 배포할 수 있음
Ingesting the deployed trained models with stream data for inference	학습된 모델을 배포하면 ML/AI 파이프라인이 종료되며, 향후 들어오는 Stream Data 전송시 배포된 모델이 예측값을 반환

“대부분의 이전 단계에서 RESTFul API를 사용하며 파이프라인은 자동화되고, ML 모델 공급과 관련된 모든 단계는 Stream data를 통해 수행”

Designing and defining ML models

- TensorFlow/IO에서 Apache Kafka를 지원해주기 때문에 Kafka-ML에서 Tensorflow를 지원 할 수 있음
- Jupyter와 같은 ML editor를 통해서도 모델을 정의할 수 있고, Kafka-ML Web UI를 통해서도 모델을 정의할 수 있음

Fig. 2. Definition of an ML model in Kafka-ML

Kafka-ML Models Configurations Deployments Training

Create Model

Name *

My model

Description

Single model with 1 hidden layer and 1 output

Imports

Code *

```
model = tf.keras.Sequential([
    tf.keras.layers.Dense(32,input_dim=100,
        activation=tf.nn.relu),
    tf.keras.layers.Dense(1,
        activation=tf.nn.sigmoid)])
model.compile(optimizer='adam',
    loss='sparse_categorical_crossentropy'
    metrics=[accuracy])
```

Go Back Create

Creating a configuration

- Kafka-ML model은 훈련을 위해 그룹화하여 사용할 수 있음
- 그룹화된 각각의 모델들의 지표들을 비교하고 평가 할 수 있음
- 이 때, 학습시 사용되는 데이터 스트림은 같아야 하고 유니크해야함

Fig. 3. Creation of a configuration in Kafka-ML

Kafka-ML Models Configurations Deployments Training

Create Configuration

Name *
New configuration

Description
Configuration to evalute models

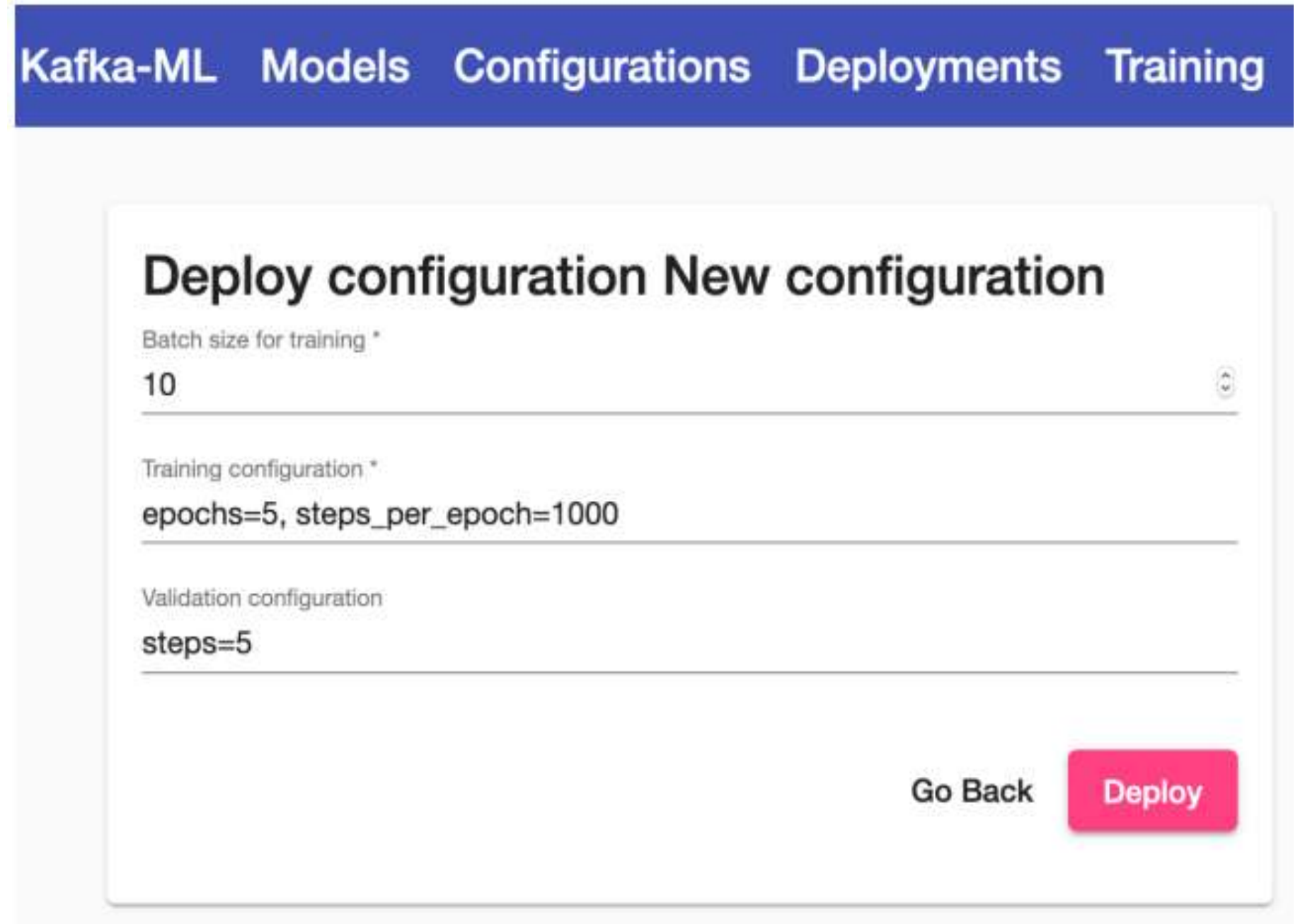
- ☒ ID37 Minst
- ☐ ID39 HCOPD
- ☒ ID40 My model

Create

Deploying the configuration for training

- batch_size, epochs 등 모델 훈련에 사용될 파라미터들을 설정 할 수 있음
- 파라미터들의 설정이 다 끝나면, 각 Kafka-ML model로 해당 task를 배포할 수 있음
- 배포하게 되면, Kafka-ML 아키텍처에서 해당 ML 모델을 가져와서 로드한 뒤, 훈련을 시작할 수 있음

Fig. 4. Deploying a configuration for training in Kafka-ML



The screenshot shows the Kafka-ML web interface with a navigation bar at the top containing links for Kafka-ML, Models, Configurations, Deployments, and Training. The 'Configurations' link is highlighted. Below the navigation bar, there is a form titled 'Deploy configuration New configuration'. The form contains three input fields: 'Batch size for training *' with the value '10', 'Training configuration *' with the value 'epochs=5, steps_per_epoch=1000', and 'Validation configuration' with the value 'steps=5'. At the bottom right of the form, there are two buttons: 'Go Back' and a red 'Deploy' button.

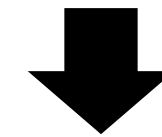
Field	Value
Batch size for training *	10
Training configuration *	epochs=5, steps_per_epoch=1000
Validation configuration	steps=5

Ingesting the deployment with stream data

- 모델이 배포되면, 훈련을 위해 data stream이 구성되고 data가 보내져야함 (모델 배포전에 선행되어야 함)
- Kafka에 데이터가 들어오지 않으면 훈련을 시작할 수 없음
- 이를 극복하기 위해 Kafka에서 최소 2개의 토픽을 사용
 - 1) data topic: 훈련과 평가하기위한 데이터 스트림만 포함하는 토픽
 - 2) control topic: 훈련과 평가가 언제, 어디서 가능한지와 같은 control message를 배포된 ML models에게 알려주는 토픽

control topic의 control message 정보

- **deployment_id**: ID of the deployed configuration where the data stream goes.
- **topic**: Kafka topic where the training and evaluation data streams are.
- **input_format**: Format of the data stream (e.g, RAW, AVRO).
- **input_config**: Configuration required by the data format chosen (e.g., the scheme in Avro).
- **validation_rate**: Percentage of stream data that will be used for evaluation. If validation_rate is equal to zero, only training will be performed.
- **total_msg**: Number of messages dispatched in the data stream. The number of messages sent are calculated by the Kafka-ML libraries.

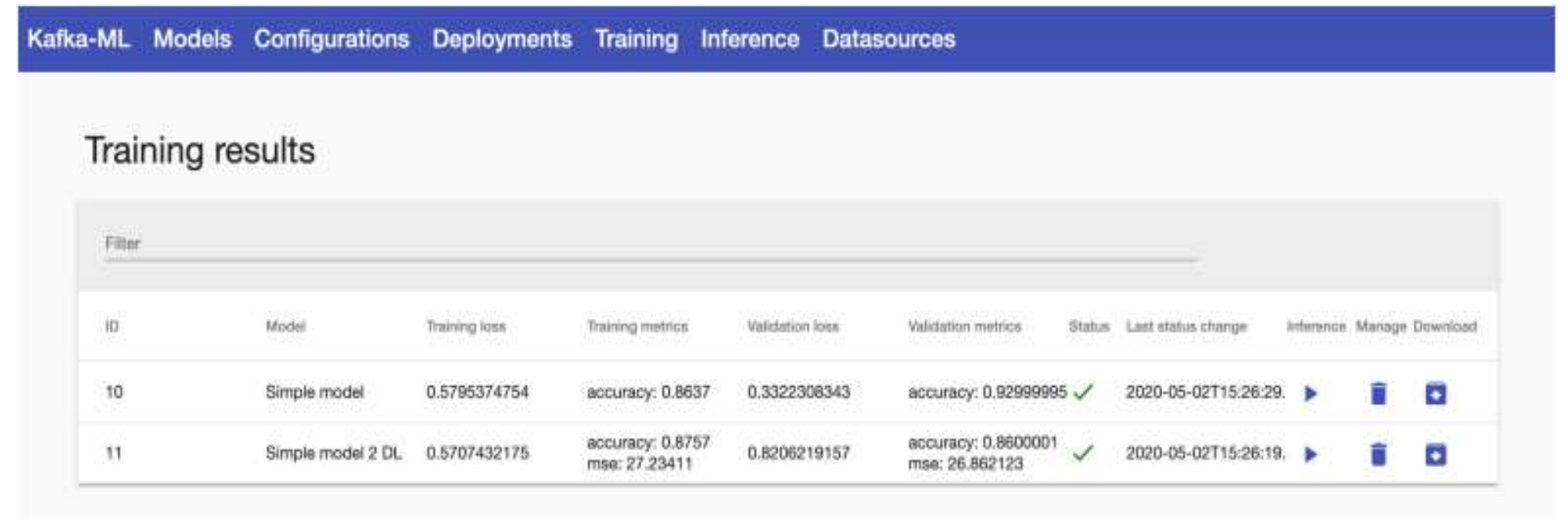


해당 deployment_id로 데이터 세트 또는 모든 정보가 전송되면
구성된 모든 ML model이 훈련을 시작하고 평가

Deploying trained models for inference

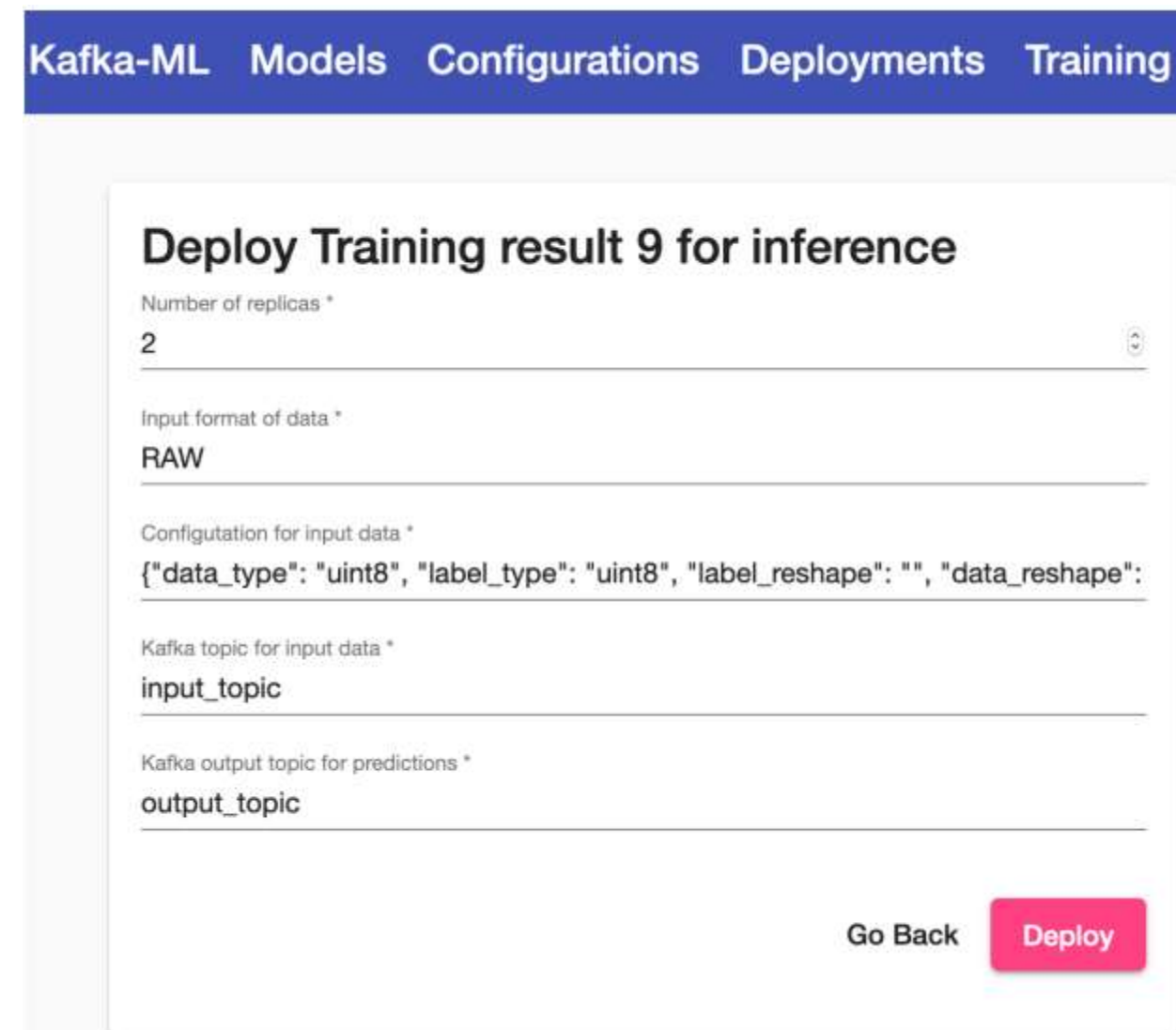
- 훈련 및 평가가 끝나면 loss와 accuracy와 같은 정의된 metrics를 Kafka-ML 아키텍처에 모델과 함께 제출
- Kafka-ML Web UI에서 해당 결과를 볼 수 있고, 편집할 수 있으며 훈련 모델을 다운받거나 추론을 위해 배포 할 수 있음
- 모든 상호작용은 Apach Kafka를 통해 이루어지며, 사용자는 예측할 값에 대한 input topic과 예측값에 대한 output topic을 구성해야 함.

Fig. 5. Training management and visualization in Kafka-ML



Kafka-ML Models Configurations Deployments Training Inference Datasources										
Training results										
Filter										
ID	Model	Training loss	Training metrics	Validation loss	Validation metrics	Status	Last status change	Inference	Manage	Download
10	Simple model	0.5795374754	accuracy: 0.8637	0.3322308343	accuracy: 0.92999995	✓	2020-05-02T15:26:29.	▶	🗑️	⬇️
11	Simple model 2 DL	0.5707432175	accuracy: 0.8757 mse: 27.23411	0.8206219157	accuracy: 0.8600001 mse: 26.862123	✓	2020-05-02T15:26:19.	▶	🗑️	⬇️

Fig. 6. Deploying a trained ML model for inference in Kafka-ML



Kafka-ML Models Configurations Deployments Training

Deploy Training result 9 for inference

Number of replicas *

2

Input format of data *

RAW

Configuration for input data *

{"data_type": "uint8", "label_type": "uint8", "label_reshape": "", "data_reshape":

Kafka topic for input data *

input_topic

Kafka output topic for predictions *

output_topic

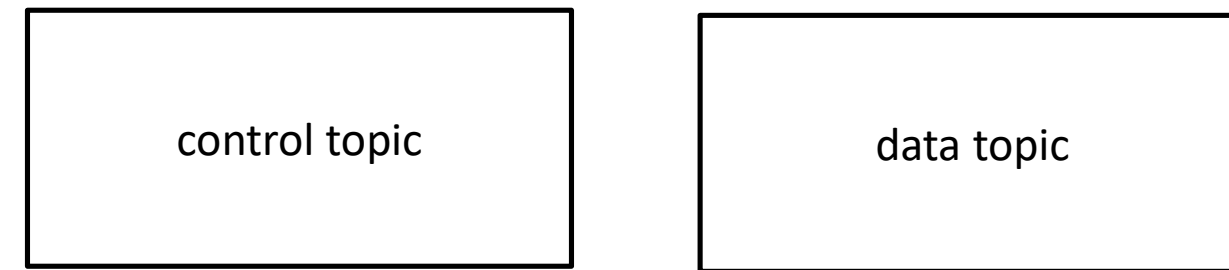
Go Back Deploy

Ingesting the deployed trained models with stream data for inference

- 학습된 모델이 준비되고 배포되어 데이터 스트림을 통해 예측 및 추천을 수행하면 ML/AI 파이프라인이 종료
- input topic과 output topic에는 더이상 control topic에 대한 메시지를 보낼 필요가 없음
- 사용자와 시스템은 input topic에 정의된 데이터 형식으로 인코딩된 데이터만 전송하면, 모델 예측이 구성된 output topic에 대한 추론 결과가 즉시 전송됨

✓ Train/Inference과정에서 사용되는 토픽

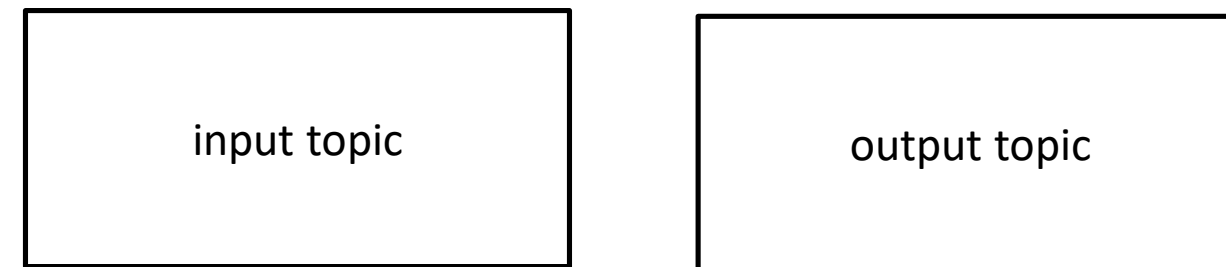
훈련과 평가를 위한 정보제공 데이터 훈련과 평가시 사용되는 데이터



✓ 추론을 위해 배포되고 나서 사용되는 토픽

예측하기 위해 전송되는 데이터

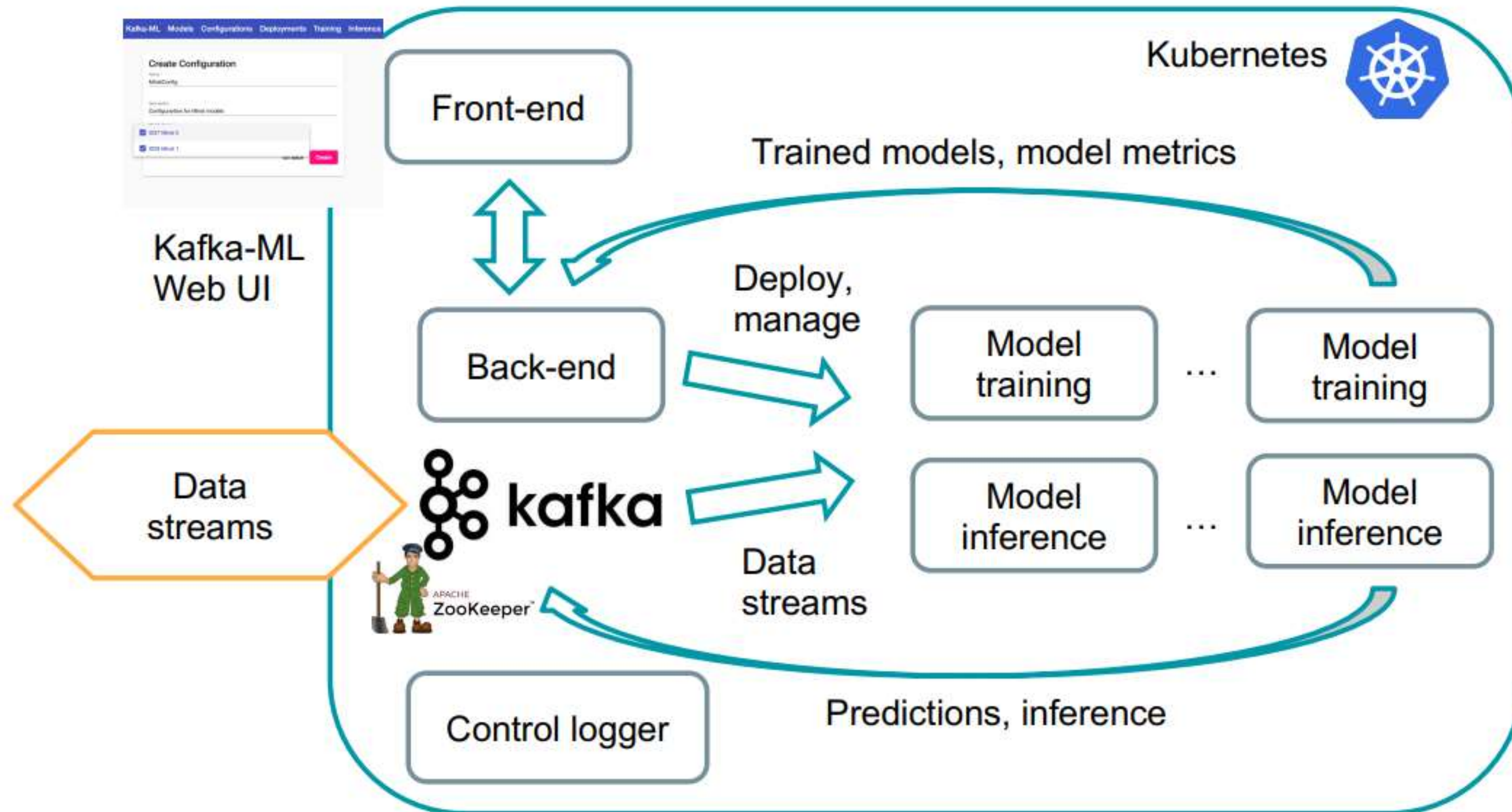
예측하고 난뒤 결과값 데이터



Kafka-ML Architecture

- Front-end
- Back-end
- Model Training
- Model Inference
- Control logger
- Apache Kafka and ZooKeeper

Kafka-ML Architecture

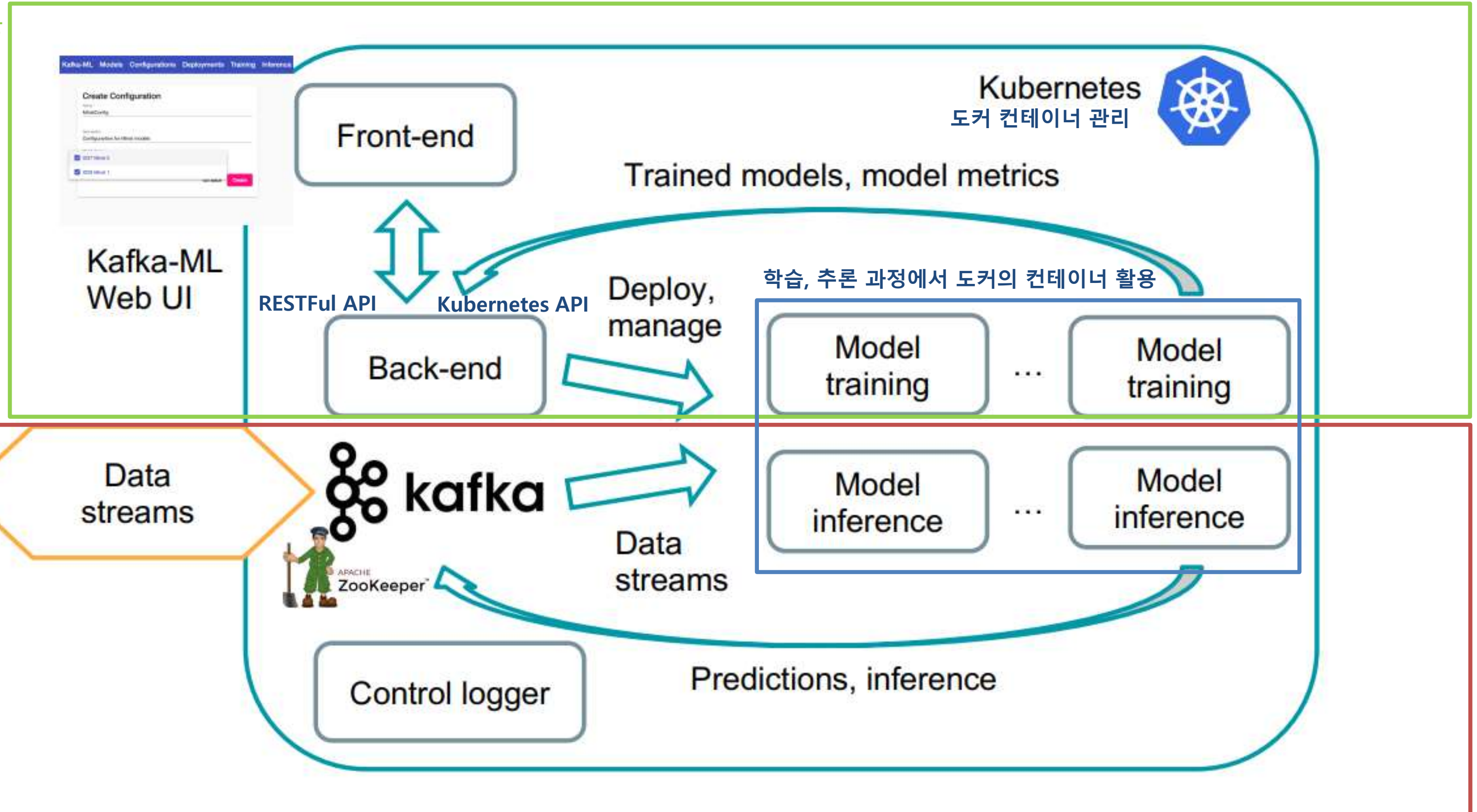


Kafka-ML Architecture

Step	Discription
Front-end	ML models를 만들고, 설정하고 학습 및 추론을 위해 배포하는 과정에서 유저친화적으로 사용 가능하도록 Web UI 제공 (Angular를 사용하여 구현)
Back-end	Kafka-ML에 포함된 모든 정보를 관리하기 위해 RESTful API를 제공, Kubernetes API와 연결되어 사용자 별로 ML 모델의 훈련 및 추론을 배포하고 관리하며, 훈련이 끝난 ML 모델 및 지표도 수신 (Kubernetes를 위한 Python Client Library와 함께 Django를 통해 구현)
Model Training	Back-end에서 구성을 배포하면 Kubernetes의 배포 가능한 단위인 Job이 Docker 컨테이너를 훈련하고 컨테이너화 하기 위해 Kafka-ML 모델별로 실행
Model Inference	ML 모델을 훈련하고 백엔드를 통해 추론을 위해 배포한 뒤, Kubernetes의 Replication Controller가 해당 모델과 함께 Kubernetes에서 실행되며, 이 때, Docker 컨테이너도 함께 실행
Control logger	Control message를 consume하고 produce하는 역할만 담당
Apache Kafka and ZooKeeper	Kubernetes의 Docker 컨테이너를 사용하여 Apache Kafka, Zookeeper를 작업으로 배포

Kafka-ML Architecture

ML모델에 대한 구성을
마치고 학습하는 과정
에서의 로직



추론을 위해 배포
된 ML모델 운영을
위한 로직

05

Kafka-ML Validation

- COPD and HC 분류 실험을 통한 Validation

만성 폐쇄성 폐질환(COPD) 환자, 건강 대조군(HC)의 타액 샘플 분류

- 타액 샘플에 대한 정보가 데이터 세트에 포함되어 있음
- Train: 80%, Test: 20% 비율로 구분하여 진행하였고, 5-fold-cross-validation method를 사용 (과적합 방지)
- Kafka-ML의 데이터 스트림 및 컨테이너화에 대한 응답시간을 측정하고 평가
- 지연 응답 시간은 훈련과 추론의 영향을 연구하기 위해 1) Kafka 통합 없이, 2) 데이터 스트림 통합 포함, 3) 데이터 스트림 통합 및 컨테이너화 모두 포함

✓ Fig.9. Architecture of a neural network model for the classification of COPD and HC samples

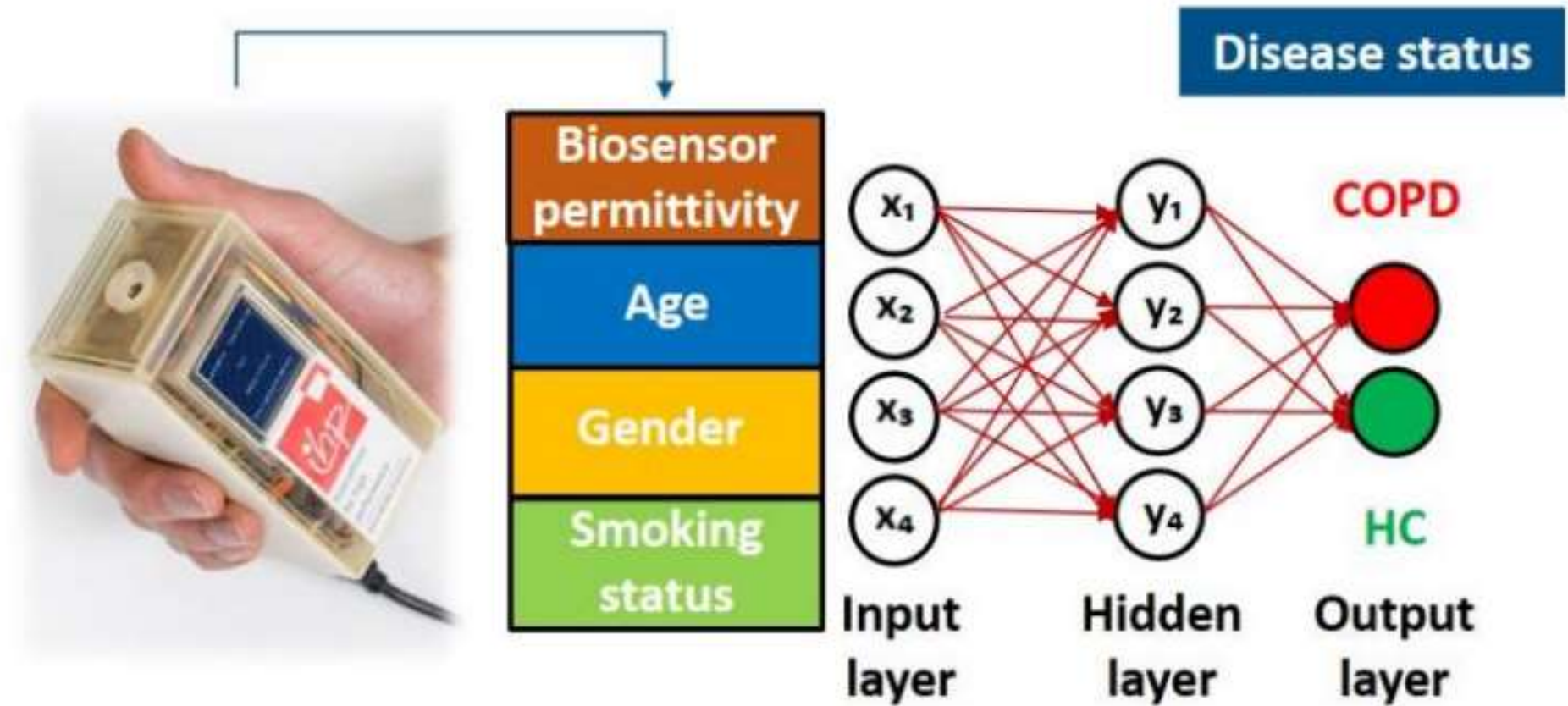


그림 9. IHP의 유전율 바이오센서를 사용하여 Exasens 데이터세트에서 사용할 수 있는 COPD 및 HC 샘플 분류에 사용되는 신경망 모델의 아키텍처

Kafka-ML Validation

✓ Training Latency Response(s)

TABLE I
TRAINING LATENCY RESPONSE (S)

Normal	Data streams	Data streams & containerization
27,37	29,61	31,44

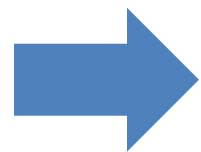
- Data streams & Containerization의 경우, 훈련하는데 있어서 지연 시간이 높은 것을 확인 할 수 있음

✓ Inference Latency Response (s)

TABLE II
INFERENCE LATENCY RESPONSE (S)

Normal	Data streams	Data streams & containerization
0,079	0,374	0,335

- Data streams & Containerization의 경우, 추론하는데 있어서 지연 시간이 낮은 것을 확인 할 수 있음



시간에 대한 실험만 있을 뿐, Accuracy나 loss에 대한 언급이 없음

감사합니다!

질문이 있으신가요?