Lecture 5: Data - Pipelines, Versioning, Cloud Storage

AI-5

Productionizing AI (MLOps)

Pavlos Protopapas, Shivas Jayaram

Outline

1. Motivation
2. Data Pipelines
3. Tutorial

Outline

1. **Motivation**
2. Data Pipelines
3. Tutorial

Motivation

The 3 components for better Deep Learning


More Data


Better/Faster Models


Faster Hardware

Motivation

The 3 components for better Deep Learning

More Data

- Extraction
- Transformation
- Labeling
- Versioning
- Storage

- Processing
- Input to Training

Better/Faster Models

- SOTA Models
- Transfer Learning
- Distillation
- Compression

Faster Hardware

- Scaling data processing
- GPU, TPU
- Multi GPU Server Training

Motivation: Data Management Challenges

Extraction

- **Varied Sources/Formats**: Data comes in different shapes, sizes, and formats.
- **Timelines of Updates**: Data can change over time, affecting model performance.

Transformation

- **Labeling**: Manual annotation is often labor-intensive.
- **Versions**: Multiple versions can cause inconsistency.
- **Quality**: Poorly processed data can lead to poor models.

Management

- **Labeling**: Consistency and quality are paramount.
- **Versions**: Ensuring data traceability and reproducibility.
- **Quality**: Ensuring the data is clean, relevant, and well-documented.

Motivation: Data Management Solution

Containerize Data Tasks

- **Benefits**: Consistent environment, easy to scale, and improves reproducibility.

Using Prebuilt Containers for Data Tasks

- **Benefits**: Saves time, ensures quality, and utilizes community-verified methods.

Manage Tasks Using Pipeline Management Tools

- **Examples**: Apache Airflow, Kubeflow Pipelines.
- **Benefits**: Streamlines data workflows, manages dependencies, and allows for easy monitoring.

Motivation: Data Management Tools

Pipeline Management

- **Kubeflow** End-to-end orchestration of machine learning pipelines

Data Labeling

- **Label Studio**
  - Annotation of text, images, audio, and more.
  - Customizable templates, multi-format support.
  - Teams needing flexibility in data labeling tasks.

Data Versioning

- **DVC (Data Version Control)**
  - Version control for datasets and machine learning models.
  - Git-like commands, storage optimization.
  - Teams that want to maintain version history of data and models.

Outline

1. Motivation
2. **Data Pipelines**
3. Tutorial

Components of an AI Application

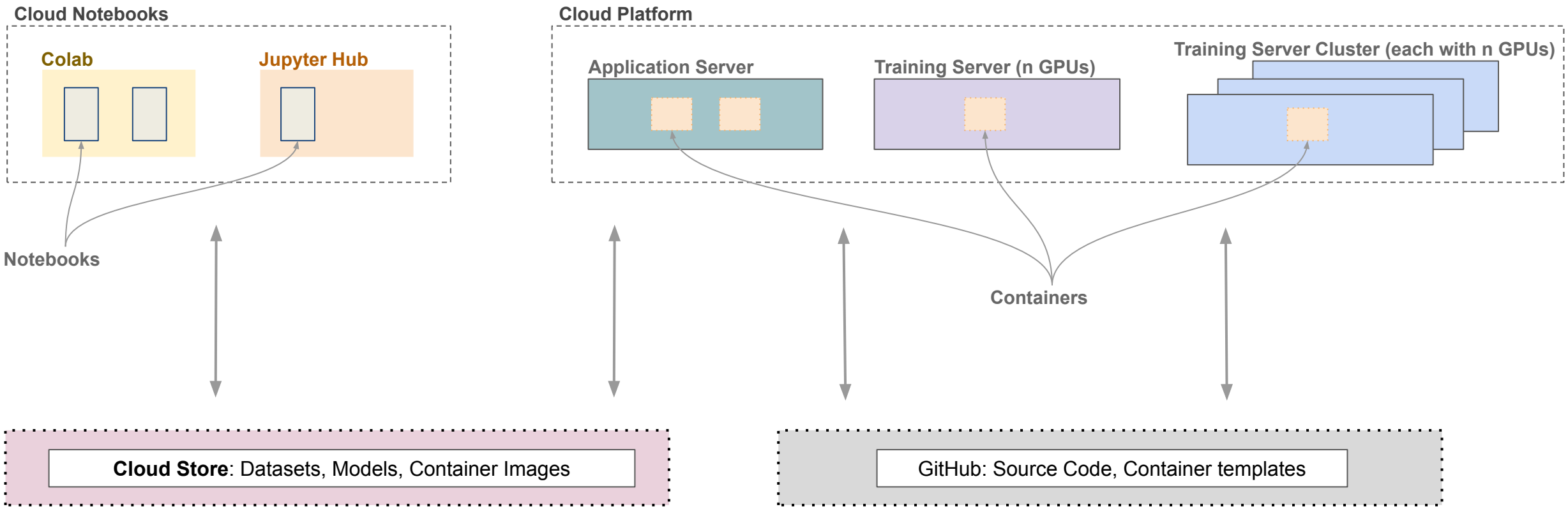
What are the components of an AI App?

- **Data**: The backbone of any AI application, needed for training and validation.
- **Model**: The trained AI algorithm
- **Source Code:**
- **Container Images:** Encapsulated environments that ensure the application runs consistently across different systems.

How do we manage all of these?

What are Pipelines

Example components of an AI App:



Wish List

We want a system with these features:

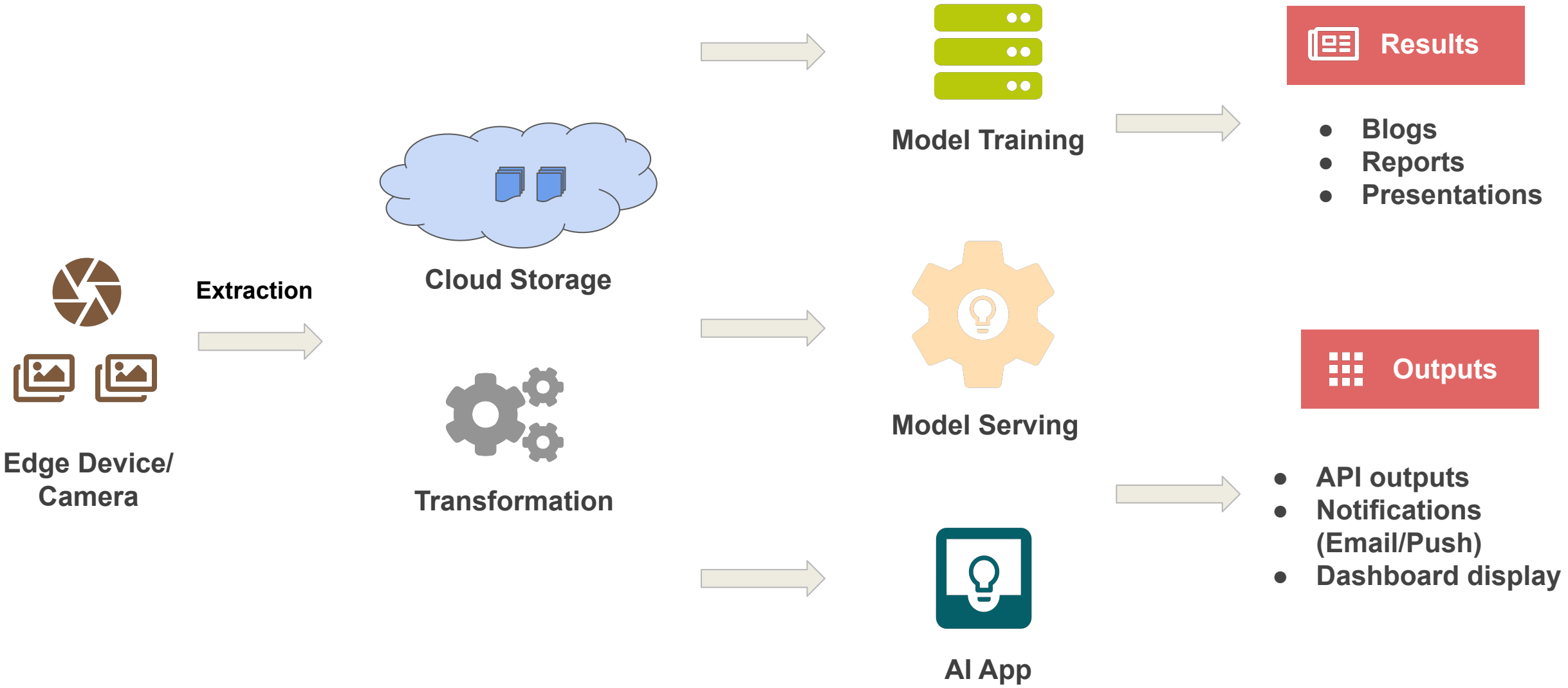
- Version control code, data, and models
- Easy access of data and models from external tools
- Automate data and model tasks

Pipelines

And a few more things like

Real-Time Monitoring of Models
Auto-Scaling Resources
Automated Testing Frameworks
Easy Rollback and Rollforward Mechanisms
Built-in Security Measures

Example AI App Pipeline

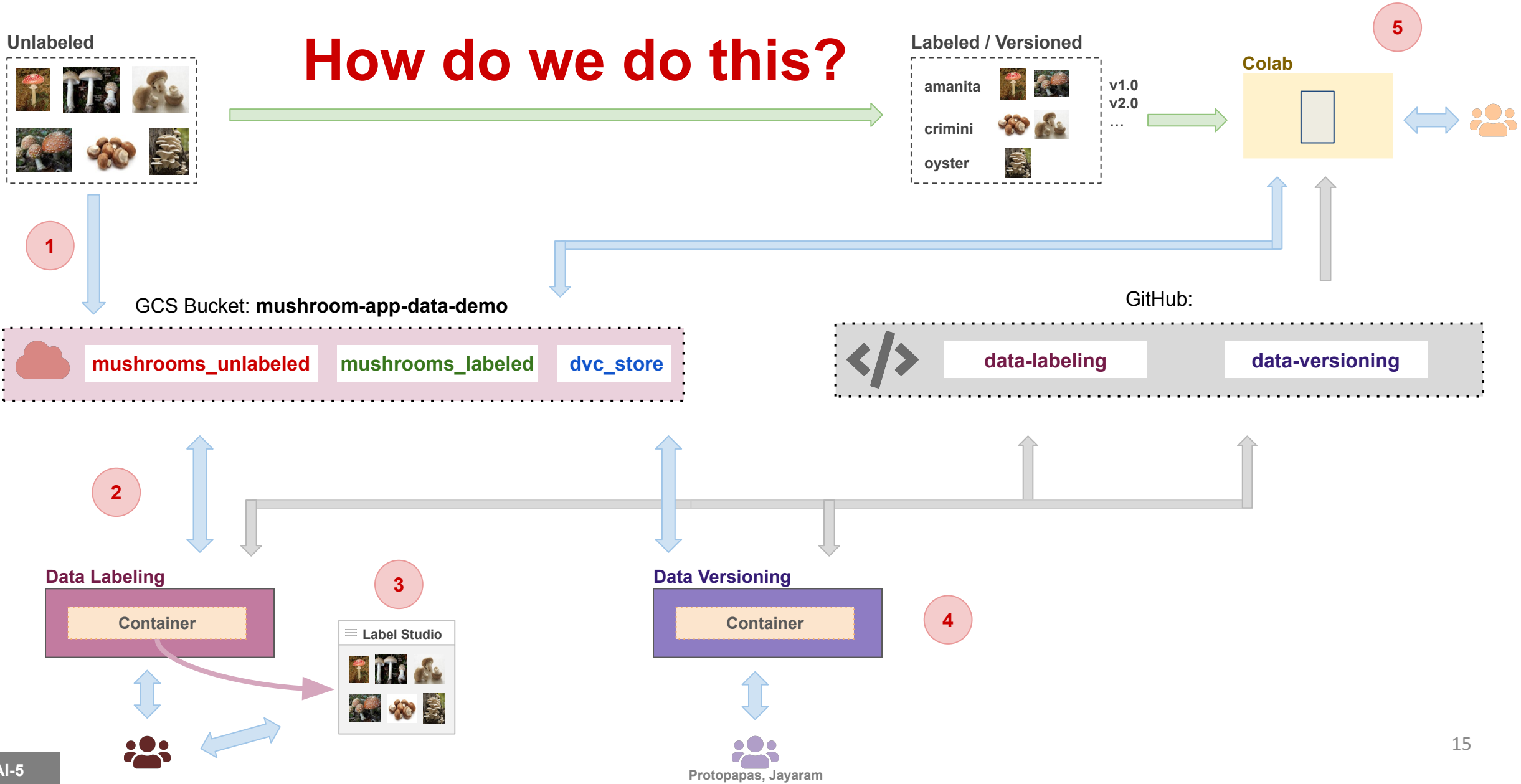


What are Data Pipelines

Various data tasks in a Machine/Deep Learning project:

- Extraction
- Transformation
- Pre-processing
- Train, validate, test split
- Pre-process step during model inference

Mushroom App Data Pipeline



Outline

1. Motivation
2. Data Pipelines
3. **Tutorial**

Tutorial: Docker Compose

For this tutorial we will use Docker Compose.

- A docker compose file is for defining and running multi-container Docker applications.
- With Compose, you use a YAML file to configure your containers.
- Then, with a single command, you create and start all the containers.

docker-compose.yml

```yaml
version: "3.8"

# Define network that the various docker containers will share
networks:
    default:
        name: data-labeling-network
        external: true

services:
    data-label-cli:
        image: data-label-cli
        container_name: data-label-cli
        volumes:
            - ../secrets:/secrets
            - ../data-labeling:/app
        environment:
            GOOGLE_APPLICATION_CREDENTIALS: /secrets/data-service-account.json
            GCP_PROJECT: "ac215-project"
            GCP_ZONE: "us-central1-a"
            GCS_BUCKET_NAME: "mushroom-app-data-demo"
            LABEL_STUDIO_URL: "http://data-label-studio:8080"
        depends_on:
            - data-label-studio
    data-label-studio:
        ...
```

List of containers to run

Volumes to mount to the container

Environment variables to set inside container

Does this container depend on another container to be up

docker-compose.yml continued

```yaml
data-label-studio:
    image: heartexlabs/label-studio:latest
    container_name: data-label-studio
    ports:
        - 8080:8080
    volumes:
        - ./docker-volumes/label-studio:/label-studio/data
        - ../secrets:/secrets
    environment:
        LABEL_STUDIO_DISABLE_SIGNUP_WITHOUT_LINK: "true"
        LABEL_STUDIO_USERNAME: "pavlos@seas.harvard.edu"
        LABEL_STUDIO_PASSWORD: "awesome"
        GOOGLE_APPLICATION_CREDENTIALS: /secrets/data-service-account.json
        GCP_PROJECT: "ac215-project"
        GCP_ZONE: "us-central1-a"
```

Port to expose from inside container to the host outside

Volumes to mount to the container

Environment variables to set inside container

Tutorial: Docker Advanced Options

```
export BASE_DIR=$(pwd)
export SECRETS_DIR=$(pwd)/../secrets/
export GCS_BUCKET_NAME="mushroom-app-data-demo"
export GCP_PROJECT="ac215-project"
export GCP_ZONE="us-central1-a"

# Run Container
docker run --rm --name data-version-cli -ti \
-v "$BASE_DIR":/app \
-v "$SECRETS_DIR":/secrets \
-v ~/.gitconfig:/etc/gitconfig \
-e GOOGLE_APPLICATION_CREDENTIALS=/secrets/data-service-account.json \
-e GCP_PROJECT=$GCP_PROJECT \
-e GCP_ZONE=$GCP_ZONE \
-e GCS_BUCKET_NAME=$GCS_BUCKET_NAME \
--network data-versioning-network data-version-cli
```

Volumes to mount to the container

Environment variables to set inside container

Network to connect the container to

Tutorial: Docker Network

```
docker network inspect data-versioning-network >/dev/null 2>&1 || docker
network create data-versioning-network
```

docker network

inspect — display info about network

data-versioning-network — Name of the network

|| — Run next command only if previous command fails

> /dev/null — **Suppresses any output**

2>&1 — **1-stdout, 2-stderr**
Redirect the stderr to the same place we are redirecting the stdout

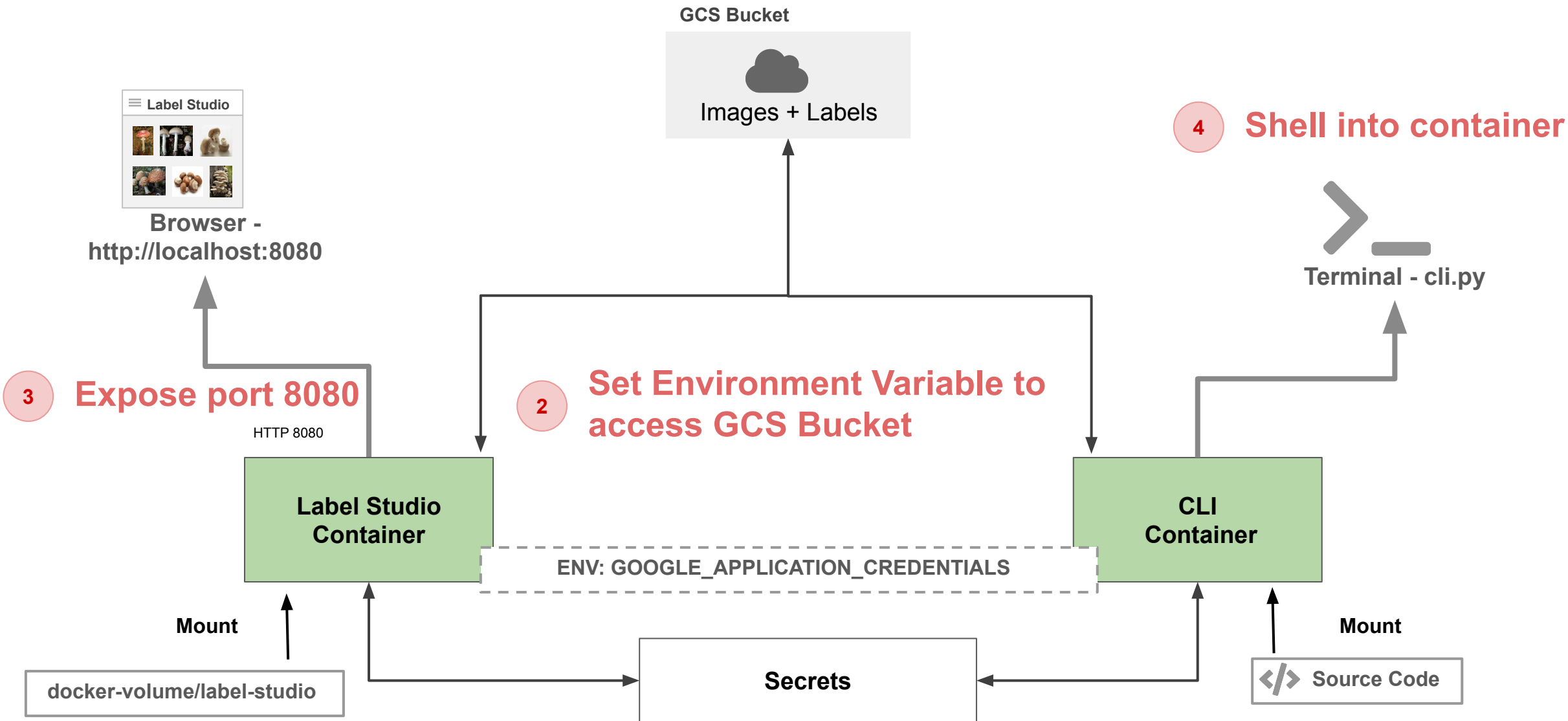
`docker network inspect data-versioning-network >/dev/null 2>&1` - Displays nothing, no error, no output

Tutorial: Service Account

A service account is a special type of Google Cloud account that represents a non-human user.

It is used by applications and virtual machines (VMs) to interact with Google Cloud services programmatically. Unlike a regular user account, which is linked to an individual end-user, a service account belongs to an application or a service running on Google Cloud Platform (GCP)

Tutorial: Label Studio + CLI



Tutorial: Mushroom App Data Pipeline

Steps to create a **Data Pipeline** to use unlabeled images and create a processes to label and version a dataset:

- Create a GCS bucket to store all data.
- Run Data Labeling Container.
- Run Data Versioning Container.
- Test data versions from Colab.
- For detailed instructions, please refer to the following link
  - Data Labeling & Versioning. (https://github.com/dlops-io/data-labeling)
  - Test Data Version Notebook. (https://colab.research.google.com/drive/1UXfp9IDnzczGYTQ_tMLGsrrKH5F397_S?usp=sharing)