CIML Summer Institute 2025
Deep Learning - Experiment Tracking

SDSC SAN DIEGO SUPERCOMPUTER CENTER

UC San Diego

Experiment Tracking

Mai H. Nguyen, Ph.D.

MLOps

- **Machine Learning Operations - MLOps**
  - ML + DevOps
  - Practices & tools to streamline and automate deployment, monitoring, and management of ML models
- **Key components**
  - Data management - data versioning, …
  - Model training - experiment tracking, …
  - Model deployment - model serving, …
  - Model monitoring - accuracy drift, …
  - Model governance - access control, …

Experiment Tracking

- **Logging metadata about experiments**
  - hyperparameters
  - data version
  - model/code version
  - training metrics
    - loss, accuracy, …
  - artifacts
    - model weights, …
  - system
    - PyTorch Lightning version
    - GPU memory usage
    - …

Experiment Tracking Demo

- **Code**
  - logger_extraction_ptl.ipynb
- **Description**
  - transfer learning - feature extraction
- **Loggers**
  - CSVLogger
  - TensorBoardLogger
  - MLflowLogger
  - WandBLogger
  - All supported by PyTorch Lightning

Loggers

Logger	Storage	Dashboard	Setup
CSVLogger	Local CSV files	None	No setup needed
TensorBoardLogger	Local logs	Local UI	Run `tensorboard --logdir`
MLflowLogger	Local (default)	Local / Remote UI	Host MLflow server (optional)
WandBLogger	Remote (cloud)	Remote UI	Set API key

Setup for Loggers

- **PyTorch Lightning**
  - Initialize and append desired loggers into a list
  - Pass list into `pl.Trainer(logger=loggers)`
- **Loggers Setup**
  - `CSVLogger` and `TensorBoardLogger` for local storage
  - `MLFlowLogger` with `tracking_uri='http://localhost:5050'`
    - Connects to local MLflow server
  - `WandBLogger` with project and run name
    - Logs to cloud dashboard after `wandb login`)

MLflow Local Setup

- **Launch MLflow server locally**

  -
    ```
    mlflow ui --backend-store-uri file:///path/to/mlflow
    ```
- **Specify port if needed**

  -
    ```
    mlflow ui --backend-store-uri file:///path/to/mlflow
    --port <port#>
    ```
- **In Python code**

  -
    ```
    mlflow.set_tracking_uri(’http://localhost:5000’)
    ```
  - Default port is 5000. Can change to different port number if needed.

MLflow Remote Setup

- **On remote machine**
  - Run `hostname -I` to get the IP address of the remote machine
  - Run below command to launch server:

```
mlflow server \
    --backend-store-uri file:///path/to/mlflow \
    --default-artifact-root file:///path/to/mlflow \
    --host 0.0.0.0 \
    --port 5050
```

- **On local machine**
  - `ssh -L 5050:<remote-ip>:5050 <username>@login.expanse.sdsc.edu`
- **In Python code**
  - `mlflow.set_tracking_uri(’http://localhost:5050)`
  - Default port is 5000. Can change to different port number if needed.

WandB Setup

- **Set up an account with Weights & Biases**
  - https://wandb.ai/site/
- **Run command in log in:**
  - `wandb login`
- **Paste your API key**
  - Login command will prompt yout to paste your WandB API key, which can be retrieved from https://wandb.ai/authorize
  - After first login, your credentials will be saved to your system. Future runs will be auto-authenticated without having to re-enter your API key.
  -

MLflow Vs. Weights&Biases

MLflow

- Manual server setup
- Local or remote storage
- Free and open-source software
- Good for private environments

Weights & Biases

- API-based
- Hosted cloud dashboard
- Free tier + paid plans
- Easy setup

Considerations

- Use **CSVLogger** for simplicity (no dashboard, local only)
- Use **TensorBoardLogger** for local training with basic visuals
- Use **MLflow** for controlled, private environments where self-hosting is preferred
- Use **WandB** for polished dashboards, team collaboration, and ease of setup
- Consider tradeoffs:
  - Privacy vs. Ease of Use
  - Local vs. Remote
  - Free (MLflow, CSV, TB) vs. Paid (WandB for teams)