

Deep Learning Optimisé - Jean Zay

Visualization Tools

cnrs IDRIS

CC BY NC SA

Why use a visualization tool ?

Can’t fix what you can’t see ◄

Training -> metrics ◄

Experiment -> comparaison ◄

Can’t fix what you can’t see





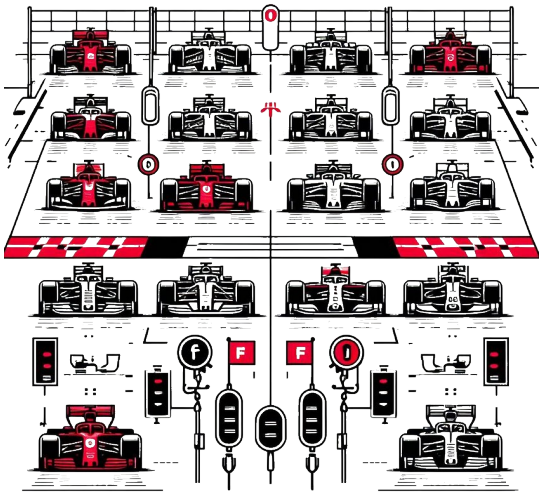
Training : metrics



Specific to learning:

- train/val loss/acc
- prototyping
- profiling & debugging

Experiment : comparaison



Not only training specific :

- hyperparameters
- dataset & source code
- hardware tracking
- multi-user / collaboration

a refined selection of tools

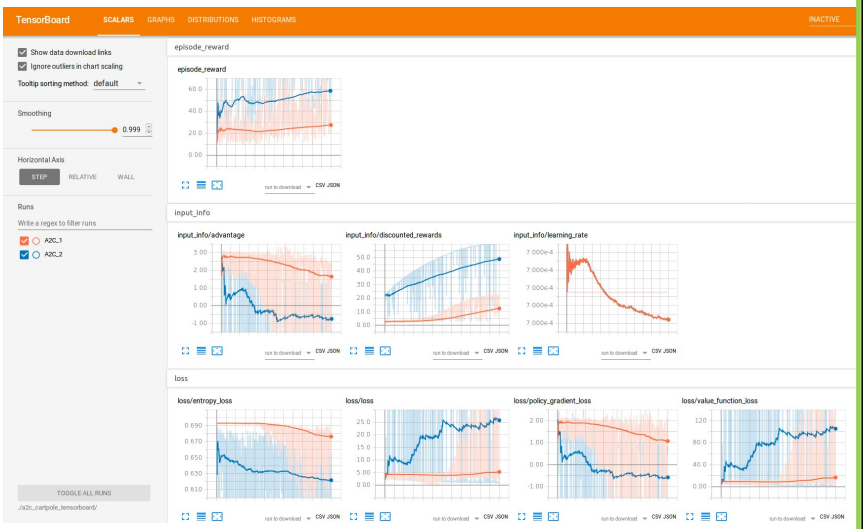
💯 TensorBoard ◂

❤ MLFlow ◂

W&B ◂

Neptune ◂

Tensorboard




```python
import torch
import torch.nn as nn
import torch.optim as optim
from torch.utils.tensorboard import SummaryWriter

# Generate some dummy data
x = torch.randn(100, 1)
y = 2 * x + 1

# Define a simple linear model
class LinearModel(nn.Module):
    def __init__(self):
        super(LinearModel, self).__init__()
        self.linear = nn.Linear(1, 1)

    def forward(self, x):
        return self.linear(x)

model = LinearModel()

# Define the loss function
criterion = nn.MSELoss()

# Define the optimizer
optimizer = optim.SGD(model.parameters(), lr=0.01)

# Create a summary writer for TensorBoard
summary_writer = SummaryWriter('./logs')

# Training loop
for epoch in range(1000):
    optimizer.zero_grad()

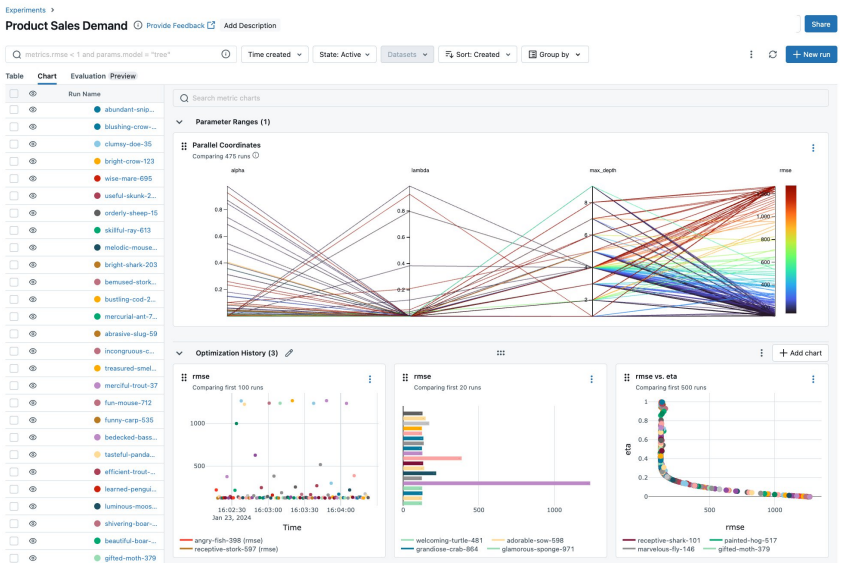
    # Forward pass
    outputs = model(x)
    loss = criterion(outputs, y)

    # Backward pass and optimization
    loss.backward()
    optimizer.step()

    # Write summary to TensorBoard
    summary_writer.add_scalar('loss', loss.item(), epoch)

# Close the summary writer
summary_writer.close()
```

MLFlow




```python
import mlflow

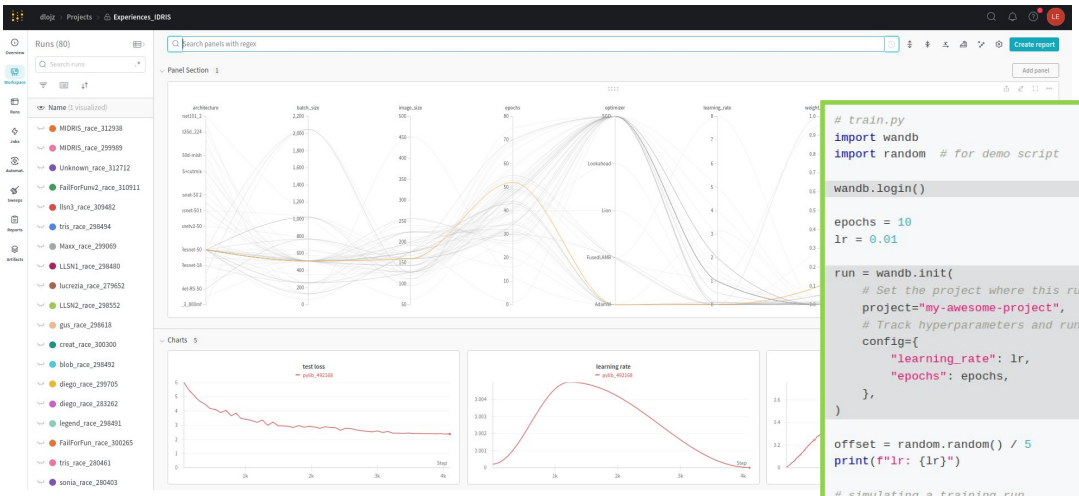
# Start MLflow run
mlflow.start_run()

# Your code to log metrics, parameters, and artifacts
# For example:
mlflow.log_param("param1", 0.001)
mlflow.log_metric("metric1", 0.987)

# Log an artifact (file)
with open("example.txt", "w") as f:
    f.write("This is an example artifact.")
mlflow.log_artifact("example.txt")

# End MLflow run
mlflow.end_run()
```

W&B




```python
# train.py
import wandb
import random  # for demo script

wandb.login()

epochs = 10
lr = 0.01

run = wandb.init(
    # Set the project where this run will be logged
    project="my-awesome-project",
    # Track hyperparameters and run metadata
    config={
        "learning_rate": lr,
        "epochs": epochs,
    },
)

offset = random.random() / 5
print(f"lr: {lr}")

# simulating a training run
for epoch in range(2, epochs):
    acc = 1 - 2**-epoch - random.random() / epoch - offset
    loss = 2**-epoch + random.random() / epoch + offset
    print(f"epoch={epoch}, accuracy={acc}, loss={loss}")
    wandb.log({"accuracy": acc, "loss": loss})

# run.log_code()
```