Assignment 2: Learning Rate Schedulers

Overview

This assignment explores different learning rate scheduling strategies in PyTorch by implementing and visualizing at least three learning rate schedulers.

Score: 95/100 ⭐

Objectives

- Understand learning rate scheduling in deep learning optimization
- Implement multiple learning rate schedulers from PyTorch
- Visualize learning rate curves across training epochs
- Compare different scheduling strategies

Task Requirements

Step 1: Choose Learning Rate Schedulers

Select at least three learning rate schedulers from PyTorch's torch.optim.lr_scheduler.

Note: Using more schedulers will improve your score.

Step 2: Implement Schedulers

Learning rate scheduling must be applied **after** the optimizer's update step.

Code Structure Example:

```python
for epoch in range(num_epochs):
    for batch in range(num_batches):
        optimizer.step()
        # Record learning rate
    scheduler.step()  # Step scheduler after all batches
```

Step 3: Visualize Learning Rate Curves

Plot learning rate changes over epochs for each scheduler using matplotlib.

Example Implementation

```python
```

```
import torch
import matplotlib.pyplot as plt
import matplotlib as mpl
mpl.rcParams['figure.dpi'] = 200

# Setup
model = torch.nn.Linear(2, 1)
optimizer = torch.optim.SGD(model.parameters(), lr=0.9)

# Define scheduler
lambda1 = lambda epoch: 0.65 ** epoch
scheduler = torch.optim.lr_scheduler.LambdaLR(optimizer, lr_lambda=lambda1)

# Training loop (10 epochs, 100 batches)
lrs = []
for epoch in range(10):
    for batch in range(100):
        optimizer.step()
        lrs.append(scheduler.get_last_lr()[0])
    scheduler.step()

# Plot
fig, ax = plt.subplots(1, 1, figsize=(3, 2))
ax.plot(range(10), lrs)
ax.set_xlabel("Epoch #", fontsize=8)
ax.set_ylabel("learning rate", fontsize=8)
plt.show()
```

Deliverables

- Jupyter notebook containing:
  - Implementation of at least 3 learning rate schedulers
  - Visualization plots for each scheduler
  - Comparison and analysis

Important:

- ✅ Include result images in the notebook
- ❌ Do NOT compress the notebook file

Requirements

```
python
```

```
torch
matplotlib
```

Installation

```bash
pip install torch matplotlib
```

Suggested Schedulers to Explore

- `LambdaLR` - Custom lambda function
- `StepLR` - Decay at fixed intervals
- `ExponentialLR` - Exponential decay
- `CosineAnnealingLR` - Cosine annealing
- `ReduceLROnPlateau` - Reduce on metric plateau
- `CyclicLR` - Cyclical learning rates
- `OneCycleLR` - One cycle policy

Results

Successfully implemented and visualized multiple learning rate scheduling strategies, demonstrating their different decay patterns and behaviors across training epochs.

Assignment completed as part of Deep Learning coursework