Assignment 1: Neural Network Foundations with JAX

Due: February 13, 2026
Submission: Brightspace Assignment 1

Objective

Understand optimization and architecture design using JAX. Learn how JAX's functional programming paradigm differs from PyTorch/TensorFlow and explore the role of `jit` and `grad` in performance and automatic differentiation.

Tasks

1. **Implement a simple feedforward neural network from scratch using JAX primitives**
    - No high-level libraries like Flax or Haiku for the core logic.
    - Include parameter initialization, forward pass, loss computation, and gradient-based updates using `jax.grad`.
2. **Train your network on the MNIST dataset**
    - Use an appropriate training/validation/testing split.
    - Normalize inputs and one-hot encode labels.
3. **Compare performance with a PyTorch or TensorFlow implementation**
    - Implement the same architecture in PyTorch or TensorFlow.
    - Measure:
        - First epoch time (including JAX compilation overhead)
        - Steady-state epoch time
        - Final test accuracy

        - **Batch size comparison:** Train with batch sizes of 64, 256, and 1024. Record timing and accuracy for each.

4. **Explain in a short report (maximum 2 pages): Use results from your experiments to support the discussion below**

    - How does JAX's functional programming paradigm differ from PyTorch/TensorFlow?
    - Discuss the role of `jit` and `grad` in performance and automatic differentiation.
    - **Focus on why performance differs rather than expecting big speedups:**
        - Compilation overhead in JAX (`jit` warm-up cost)
        - Functional style and immutability
        - Graph optimizations and kernel fusion
        - Impact of batch size on amortizing compilation cost
    - Reflect on your experience: What was easier/harder in JAX vs PyTorch/TensorFlow?

GPU Requirement

- **You must run your experiments on a GPU** to ensure realistic performance comparisons.
- Options:
    - **Morningstar cluster** (available through the university)
    - Your **personal GPU**
    - **Google Colab** or other cloud-based GPU services (e.g., Kaggle, AWS, Azure)
- Document which platform you used and any setup steps in your report.

Deliverables

- **Code:**
    - JAX implementation (pure JAX, no Flax)
    - PyTorch/TensorFlow implementation
    - Reproducibility readme file

- **Report (PDF):**
  - Architecture details
  - Performance comparison (include timing and accuracy for different batch sizes)
  - Conceptual discussion
  - Reflection on AI tool usage (if applicable): *What did AI suggest? What did you accept/reject and why?*

Grading Rubric (100 points)

Component	Points
JAX Implementation	30
- Correct forward pass & loss	10
- Gradient updates using `grad`	10
- Use of `jit` for optimization	10
PyTorch/TensorFlow Implementation	20
- Correct architecture & training	10
- Performance comparison included	10
Report (Conceptual Discussion)	30
- JAX vs PyTorch/TensorFlow paradigm	10
- Role of `jit` and `grad` explained	10
- Analysis of performance differences	10
Code Quality & Documentation	10
- Clear, readable, commented code	10
Bonus	10
- Visualization of training curves	5
- Insightful AI tool reflection	5