Workshop

Building Large Language Models from First Principles

Overview

A 6-day intensive, hands-on in-person course conducted on campus for undergraduate students focused on developing a deep understanding of large language models—covering conceptual intuition, mathematical foundations, and practical implementation. The workshop is interactive and laptop-based, with dedicated time for hands-on coding and implementation of real research papers.

Learning Objectives

Students will:

- Build and understand automatic differentiation and backpropagation from scratch
- Develop proficiency in PyTorch and modern deep learning workflows
- Implement language models progressing from n-gram to transformer architectures
- Understand training stability mechanisms (normalization, variance scaling)
- Construct a complete decoder-only transformer and generate text
- Survey contemporary architectural innovations and research directions

Target Audience

Second and third-year undergraduate students. Prerequisites:

- Basic linear algebra (matrix multiplication, vector dot product)
- Calculus (partial differentiation).
- Basic Python (classes, functions, control flow)

Itinerary

Pre-workshop setup

Environment configuration (PyTorch, CUDA verification)
Verify setup with hello-world script

- dependencies: `uv`, `graphviz`, `vscode`, `vscode jupyter extension`

Day 1: Automatic Differentiation

Build scalar autodiff engine from scratch
Implement backpropagation manually
Train simple networks (AND gate, XOR failure demonstration)
Takeaway: Understanding gradient flow at the lowest level

Day 2: Language Modeling Basics

Transition to PyTorch
Build bigram and trigram models (Makemore-style)
Softmax intuition, negative log-likelihood
Data loading, batching, sampling
Takeaway: Understanding probabilistic language modeling

Day 3: Multi-Layer Perceptrons

Build MLP-based language model from scratch
Introduce nn.Module and modern PyTorch patterns
Hyperparameter tuning, sampling strategies
Non-linearities and their importance
Takeaway: Transition from lookup tables to learned representations

Day 4: Normalization & Training Stability

Batch Normalization: paper reading + critical analysis of "internal covariate shift"
Layer Normalization: why it works better for sequences
Instability with depth demonstrations
Understanding variance scaling and gradient flow
Takeaway: Why normalization is critical for deep networks

Day 5: Transformer Architecture

Self-attention mechanism (mathematical derivation)
Multi-head attention
Positional encodings (sinusoidal)
Feed-forward networks, residuals, LayerNorm
Complete transformer block implementation
Takeaway: Understanding the architecture that powers modern LLMs

Day 6: Full Implementation & Future Directions

Train decoder-only transformer on Tiny Shakespeare
Sampling and generation

Survey of advanced topics: MHA variants (GQA), positional encodings (RoPE, PoPE), tokenization (BPE)

Pathways for continued learning

Takeaway: Complete working LLM and roadmap for further study
