Allan Zhang

Last updated in January 2025

Los Angeles, CA allanzhang440@gmail.com allanzhang@g.ucla.edu
609-943-8429 https://giyushino.github.io giyushino

Education

University of California, Los Angeles *Sept 2024 – June 2028*
BS in Applied Mathematics

- GPA: 4.00
- **Relevant Coursework:** Multivariable Calculus, Linear Algebra, Discrete Structures, Intro to C++ (Math 32A, Math 32B, Math 33A, Math 61A, CS 31)

Experience

Undergraduate Research Assistant | BigML *Los Angeles, CA*
UCLA *Nov 2024 – Present*

- Aided PhD student in developing pipelines to train and evaulate custom, lightweight VLMs on spatial reasoning tasks, attemping to understand where and why model fails. Findings allow us to work on creating optimal, high-quality datasets to improve performance
- Wrote functions to create highly modular datasets, testing the effects of modality mismatch, spurious correlations, alignment between visual and textual context, and number of unique images per class on model's performance
- Utilizing mechanistic interpretability techniques, used hooks to extract each layer's activation patterns, as well as the model's predictions during each layer. Created heatmaps using Seaborn to visualize how data is processed throughout the model
- Used custom linear probes to determine if tensors passed through model still retained spatial information, specifically testing the tensors passed through the multimodal projector

Projects

TinyMathLLM *TinyMathLLM*

- Tested different methods of training and fine-tuning small, opensource models to create lightweight math chabots that can run locally. Attempted to see if model could learn to do math from question + answer dataset. Also tested if model could learn to extract important values from questions and classify problem types, allowing it to be implemented into a math agent
- Experimented if TinyLlama (1.1B parameters) could truly learn how to do math, specifically testing model on addition, multiplication, derivatives, integrals, matrix multiplication, and determinant questions
- Both methods were trained using 4bit-quanitzation + low-rank adaption, as well as built-in Transformers training loop. Wrote custom LORA implementation from scratch to speed up inference
- Created custom datasets for both use cases, including 100k+ example user inputs + model responses. Used Deepkseek-chat as judge to guage model performance

Doodle Guesser *MyOwnDoogleGuesser*

- Fine-tuned vision transformer (CLIP-Vit-Large-Patch-14) on 6 different animal drawings. Collected and processed 1,000,000+ images from 6 classes to create custom dataset. After training, accuracy increased from 54% to 87%. Worst group accuracy increased from 39% to 76%
- Created CLIP model from scratch using PyTorch. Wrote custom tokenizer and encoders to embed input labels and images into multi-dimensional vectors. Acheived 70% accuracy after only being trained of a subset of the previous dataset (200,000 images)
- Created GUI using Pygame to allow users to draw on a canvas. After drawing, screenshot was taken, processed, and fed into model, softmax taken of the outputted logits to predict the most likely class

UCLA Dining Assistant *What2Eat@UCLA*

- Using OpenAI API and Deepseek-chat, created chatbot that recommends UCLA dining halls based on student preferences. Real time data about dining halls menus and hours was scraped from UCLA Dining website and processed using BeautifulSoup
- Deepeek was fed with data about each dining hall, including menu + nutrional information for each item. Taking

into account user's flavor/cuisines preferences, dietary restrictions, and other criteria, chatbot would recommend dining halls

- Currently fine-tuning small, opensource model on example chat data to improve performance and remove reliance on OpenAI. Model can be run locally using 4-bit quanitzation

Efficient Finetuning Pipeline *Fine-Tuning Functions*

- Wrote functions to efficiently fine-tune and evaluate models on weak GPUs or CPU. Functions focused on batching images fed into model to reduce VRAM requirements and preventing Google Colab from crashing
- Tested on OpenAI's CLIP-Vit-Large-Patch-14 model using CIFAR-10 dataset. Using built-in training functions from Hugging Face caused Google Colab to crash, custom functions did not. Saw 5% improvement in accuracy (91% $\rightarrow$ 96%) with limited training data, computational power, and time. To prevent overfitting, wrote new function to shuffle training dataset for every epoch
- Tools Used: Python, PyTorch, Google Colab, Hugging Face

Skills

Programming Languages and Frameworks: Python, PyTorch, NumPy, Matplotlib, TensorFlow, scikit-learn, OpenCV, Hugging Face, LaTeX, C++ (basic)

Languages: English, Korean