Allan Zhang

31 E Funderburg Blvd, Mountain House, 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, Computer Science (Math 32A, Math 32B, Math 33A, Math 33B, Math 61A, CS 31)

Self Study *July 2024 - Present*
Machine Learning and Data Science

- **Textbooks:** *Reinforcement Learning: An Introduction*, *An Introduction to Statistical Learning with Applications in Python*, *Dive into Deep Learning*
- **Courses:** Andrew Ng's Machine Learning Specialization

Experience

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

- Set up experiments involving vision transformers, 2D projection layer (MLP), and LLMs to study why VLMs perform poorly on spatial reasoning tasks
- Used mechanistic interpretability techniques to extract tensors from every layer, analyzed them to understand how each layer processes input data. Created heatmaps using Matplotlib and Seaborn, visualizing model's predictions for every input token

Projects

Doodle Guesser *MyOwnDoogleGuesser*

- Using vision transformer (OpenAI's CLIP-Vit-Large-Patch-14), downstreamed model to predict animal drawings created by users. Collected and processed 1,000,000+ images from 6 classes to create dataset to fine-tune model on. Improved model's accuracy from 54% to 87% after fine-tuning
- Also created CLIP model from scratch using PyTorch. Wrote custom tokenzier and encoders to embed input labels and images into multi-dimensional vectors. Achieved 70% accuracy after being trained on subset of previous dataset, only using 200,000 images.
- Both models used to create game similar to Google Doodle where users draw animals from 6 specified classes and model guesses what animal was drawn. Users' drawings converted into 28x28 tensor consisting of 0s and 1s, fed into model for prediction
- Tools Used: Python, PyTorch, Pygame, Hugging Face, NumPy

Efficient Finetuning Pipeline *Fine-Tuning Functions*

- Created 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

Hand-Controlled Drone *Drone Control*

- Created and labeled dataset consisting of 400+ images of hands pointing up, down, left, right, backwards, and forwards. Trained YOLOv8 model with the data and utilized OpenCV to detect directions real time.

Fed real-time collected information to FPV drone's flight computer to control movement

- Tools Used: Python, OpenCV, YOLOv8

Flappy Bird AI *Work in Progress*

- Created own version of Flappy Bird in Python using Pygame. Currently creating neural network that utilizes Q-learning to learn to play the game. Game state is collected at every frame is collected and fed into model, including bird's vertical position, vertical velocity, and distance from both pipes
- Tools used: Python, PyTorch, Pygame

Self Published Novel *2020 - 2021*

- Wrote a 45,000 word novel, published it online. Amassed 272,000 reads, 7,000 comments, and 6,200 reviews. Reviewed other authors' works, providing grammatical advice

Skills

Programming Languages and Frameworks: Python, PyTorch, NumPy, Matplotlib, TensorFlow, scikit-learn, OpenCV, Hugging Face, LaTeX, MATLAB, HTML (basic), CSS (basic), Bash, Git

Languages: English, Korean