Yuqi Xie

912 W 22nd Street Apt 208, Austin, TX, 78705
737-341-4564 xieleo@utexas.edu xieleo5.github.io

Education

University of Texas at Austin **Sep. 2023 – Present**
Master's in Computer Science - College of Natural Science *Texas, U.S.*

University of Michigan, Ann Arbor **Sep. 2021 – Apr. 2023**
Bachelor of Computer Science - College of Engineering *Michigan, U.S.*

Shanghai Jiao Tong University **Sep. 2019 – Aug. 2023**
Bachelor of Electrical and Computer Engineering - UM-SJTU Joint Institute *Shanghai, China*

Publication

Voyager: An Open-Ended Embodied Agent with Large Language Models **In submission, 2023. Arxiv**
Guanzhi Wang, **Yuqi Xie**, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, Anima Anandkumar

Research Experience

Collaboration with NVIDIA and UT Austin **April 2022 - August 2023**
Undergraduate Research Assistant | Advised by Prof. Yuke Zhu, Dr. Linxi "Jim" Fan *Remote*

- **Voyager: An Open-Ended Embodied Agent with Large Language Models**
  Voyager is the first lifelong learning agent that plays Minecraft purely in-context. It continuously improves itself by writing, refining, committing, and retrieving code from a skill library, all **without relying on gradient descent**.
  - Developed the asynchronous backend server in JavaScript responsible for agent control and game observation.
  - Leveraged OpenAI's GPT API to generate **Code as Policy** and used **Chain-of-Thought** prompting to improve code.
  - Created the control primitives as both helper functions and examples of code writing.
  - Designed the curriculum, action, self-verification, and skill library execution loop for the agent.
  - Proposed the **warm-up schedule** and the **auto-resume mechanism** to optimize the training process.
  - Conducted experiments on downstream unseen tasks and involved human interactions for building tasks.
- **Auto Machine Learning Code Generation with Large Language Model**
  Existing AutoML tools, such as Auto-Sklearn, are based on brute force searching and Bayesian optimization. Our goal is to train a code generation model to achieve an end-to-end approach from DataFrames to code for the best model.
  - Collected and cleaned web-scaled data from solutions to Kaggle competitions.
  - Synthesized data using the optimal model searched by current AutoML algorithm.
  - Fine-tuned Codex model with **RLHF** using the accuracy of the generated code as the reward.
- **Development for MineDojo2**
  MineDojo is a framework built on the popular *Minecraft* game, featuring a simulation suite with thousands of diverse open-ended tasks and an Internet-scale knowledge base. My contributions include:
  - Dramatically modified the backend infrastructure to allow the environment to run faster in a later version of the game, and making the environment more scalable for training a large-scale open-ended agent.
  - Implemented a universal interface instead of cheating commands for an agent to complete zero-shot tasks like a human.

Projects

Visual Language Model with Multi-modal Reasoning | *Major Design Project* **Summer Semester, 2023**
- Collected and synthesized multi-modal training data from open source datasets such as MIMIC-IT, VIST, VQA, .etc
- Fine-tuned **Vicuna-7B** with an image encoder and decoder from **Stable Diffusion** 2.1 on the datasets.
- Created a user interface for chatting with the model using image and text based on Gradio.

Insta485 | *Flask, React, JavaScript, Python, SQL, AWS* **Winter Semester, 2021**
- Built the client side dynamic pages using React and JavaScript.
- Created the server side endpoints and databases using Flask, Python, and SQL.
- Implemented a search engine using tf-idf scoring and map-reduce techniques.

Probing into the Reason behind Wasserstein GAN's Success | *Machine Learning Course Project* **Fall Semester, 2021**
- Implemented Wasserstein in Generative Adversarial Network, as well as WGAN with gradient penalty.
- Stated the problem of the original GAN loss function and analysis the modifications made by WGAN mathematically.
- Compared the stability of WGAN's loss function with the widely-used DCGAN using FID scores.

Technical Skills

Skills: Propmt Engineering, Natural Language Processing, Reinforcement Learning, Computer Vision
Software and Libraries: Pytorch, Hugging Face, Gym, Unreal, Ray, Docker, React, Selenium, Unity
Programming Language: Python, Java, JavaScript, SQL, C++