**17 May 2025**

Started the day by searching for a suitable dataset, but couldn’t find one that met all the criteria. Decided to move forward with web scraping instead. Initially attempted to scrape 100 questions and answers from five Stack Overflow pages tagged with pytorch. However, I soon realized that some questions had no answers, so I updated the script to skip those and store only the highest-voted answer per question.

Later, I revised the goal to extract 1,000 top-rated questions and answers based on vote count. Eventually, I finalized a script that scrapes top-voted Q&A pairs across various machine learning and deep learning frameworks.

Upon running the script, I encountered a limitation: Stack Overflow's API enforces strict rate limits on user calls. As a workaround, I plan to run the script incrementally over time. In the meantime, I moved on to the next task in the pipeline.

Created a preprocessing script that concatenates all the JSON files and formats them into the structure:

{

"instruction": "How do I fix 'command not found' in Ubuntu?",

"input": "",

"output": "Ensure the package is installed. Use `sudo apt install <package>`."

}

This part went smoothly and worked well on the data I had collected so far.

Next, I began working on the fine-tuning script. Attempted to use Unsloth with LLaMA 3–8B in 4-bit precision, but it exceeded my laptop’s GPU capacity. That’s enough progress for today. Tomorrow, I’ll continue scraping and try using Hugging Face’s transformers and peft libraries for fine-tuning instead.

**18 May 2025**

Today, I finalized the web scraping process for the dataset, utilizing a Stack Overflow API key to increase the request limit from 300 to 10,000. This allowed for significantly broader data collection across various machine learning and deep learning topics. As part of the data cleaning effort, I revisited the preprocessing script to ensure that any sensitive information—such as Hugging Face and OpenAI API keys inadvertently captured in code snippets—was identified and removed. I then attempted to upload the full dataset along with the code to GitHub, but encountered issues due to large file sizes and the presence of sensitive keys in some of the raw files. Given these constraints, I decided it was best to upload only the cleaned codebase, excluding the dataset for now. Additionally, I submitted requests for access to gated model repositories on Hugging Face, which are necessary for the upcoming fine-tuning phase. I plan to resume work on the fine-tuning pipeline tomorrow, likely switching from Unsloth to the transformers and peft libraries due to hardware limitations on my local machine.

**19 May 2025**

I moved on to setting up the fine-tuning pipeline. Chose to work with TinyLlama/TinyLlama-1.1B-Chat-v1.0 as the base model to keep VRAM usage low. Installed necessary libraries like transformers, datasets, peft, bitsandbytes, and accelerate. Wrote a training script using Hugging Face's Trainer class and integrated LoRA via PEFT. Made sure to wrap the base model with LoRA adapters for q\_proj and v\_proj attention layers. Used 4-bit quantization to reduce memory usage.

Initially tried to use bfloat16, but encountered a compatibility issue since my RTX 3060 GPU doesn’t support it. Switched over to fp16=True and bf16=False, which resolved the issue. Also disabled TensorBoard logging since it caused runtime errors, and replaced it with simple progress logging using tqdm. Added a few enhancements like dynamic input formatting and printing a tokenized sample before training to verify correctness.

Trained the model successfully with 4-bit LoRA adapters and mixed precision. The setup ran efficiently on my hardware. Tomorrow, I plan to test the fine-tuned model in a chat loop, explore PEFT further by inspecting trainable parameters, and possibly start working on merging the adapters into the base model or converting it to GGUF for llama.cpp inference.