AMD AI SPRINT

Team : Star Pioneers

Track 1 - AAIPL

Task Overview

- Create 2 Agents
  - Q Agent: Creates questions for the given topics
  - A Agent: Answers questions generated by the opposing team
- Teams are pitched against each other to come up with scores for the Q and A agents
- Access to M3150 GPU for 24 hours was given for this task

Dataset curation

- We used Gemini-2.5-pro batch API calls to curate questions from all the 3 topics for training.
- We validated the questions in the curated dataset using Gemini-2.5-pro to check for the correctness
  - We ensure the questions have a solution
  - One and only one of the options is correct
  - Other options are incorrect
- We also validated the token limits of the generated questions
- Only if all the validations passed the set was added to the gold dataset.

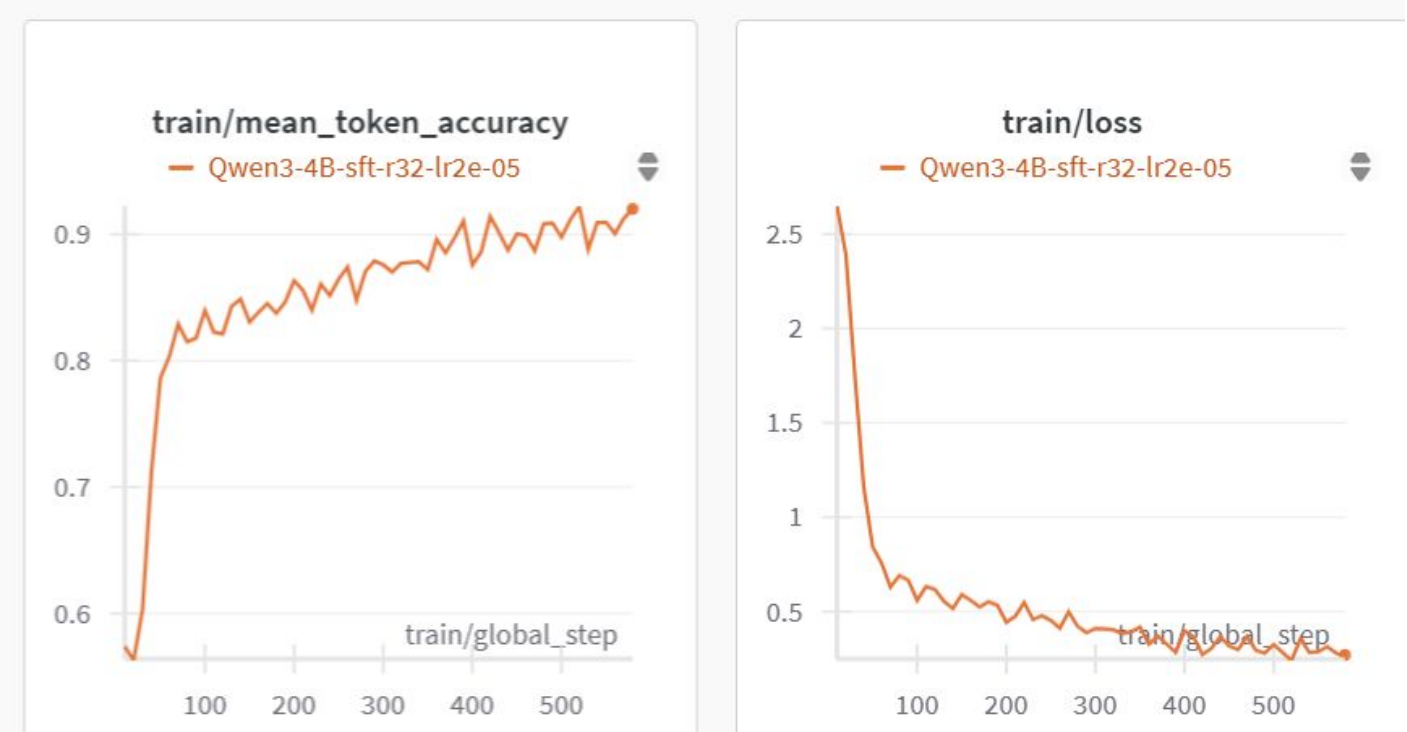
Q-Agent

- Fine-tuned it on curated dataset. We used Parameter-Efficient Fine-Tuning-LoRA technique.
- **Implemented Dynamic Prompt Engineering for Q-Agent:** To ensure variety in the generated questions, we created a dynamic prompt system. For each batch, the script randomly selects from a pool of:
  - **Personas:** (e.g., "master logician," "psychic," "wordsmith") to change the question's tone and style.
  - **Instructions:** To vary the type of difficulty (e.g., asking for hidden traps vs. testing deep concepts).
  - **Structure requests:** The purpose of this was to add variety not just to the question itself, but also to the style of the explanation.

A-Agent

- Fine-tuned the A agent on curated dataset
- Have used tree-of-thought (ToT) prompting for answer generation
  - Specifically used for problem solving in a lot of areas (such as crossword puzzles)
- A-agent first detects the topic a question is from the question and calls the ToT prompt specifically created for this topic
- Fine tuning was done with ToT prompts as well since that is what the model would receive during inference

A-Agent Finetuning Curves



Ideas Explored but not Pursued

To evaluate the questions generated by our Q model, we considered using a constraint solver, if they were logically correct with the answer we keep them otherwise discard and re-generate the rest

Extending this idea we wanted to try to solve the questions using LLM to understand the problem and then use a constraint solver

This would ensure a hallucination-free, consistent output

Thank you