AI4001/CS4063 - Fundamentals of NLP/NLP
Course Project

1 Objective

The project is an important part of this course and makes up a significant portion of your final mark. The project gives you a chance to develop a small AI Agent from among the options given and rigorously test it for scalability, reliability and performance. Your work should be well documented for reproducibility.

In particular, you will be expected to become familiar with existing work on Agentic solutions and develop a small system that is containerized and can be used as an API and tested via Postman.

You may do your implementation within any freely-available infrastructures, toolkits and models (Phi, Llama, StableDiffuson etc), however, the solution should be self-contained and automatically deployable via workflows/deployment scripts. Your microservice should be detached from the front-end that you would develop to demo the functionality of your microservice.

2 Microservice Requirements

The microservice you develop is expected to:

- Have gRPC API with at least 1 core endpoint (e.g., /generate, /infer)
- Handle multiple requests concurrently (basic async/concurrency)
- Support minimal error handling (invalid input, server errors)
- Return JSON responses with status codes
- Should be thoroughly tested (testcases to be submitted)
- Should be demoable through Postman + a minimal fronted using Gradio or StreamLit.

3 Project Ideas

Projects should be done in groups of two or three and must solve one of the following problems:

- Story2Audio — Given a storyline, convert it to an engaging audio-story (like an audio book). The audio tone should match the text being read and not in monotonic voice.
- TrendStory — Write a program to extract current trends (for the past 24hrs, from youtube, tiktok or Google) and generate a story script in a particular theme of your choice (comedy, sarcasm, tragedy etc)
- Text2Image — Given a sentence or a paragraph along with context (text, context-of-text) generate an appropriate image.

- Text2Video — Given a sentence or a paragraph along with context (text, context-of-text) generate an appropriate video.
- Image2Video — Given an image and its context in text (image,context-of-image) generate an appropriate video.

As a suggestion use off-the-shelf open-weights models, GitHub Actions for CI/CD and Huggingface Spaces to deploy your model.

4 Submission Guidelines

The deliverables expected are:

1. Source code in a GitHub repo (also to be submitted on classroom)
2. Gradio/StreaLit frontend application + Postman collection with test cases
3. Dockerfile and/or deployment scripts
4. Documentation (README with setup, usage, architecture, model sources, limitations)
5. Testcases and performace evaluation (performance graphs like # of concurrent requests/response time)
6. Short video demo (optional but helpful)

Evaluation Criteria:

- Functionality and Correctness: 30%
- API Design and Usability: 10%
- Code Quality and Documentation: 20%
- Testing and Fault Tolerance: 20%
- Reproducibility and Deployment: 20%

5 Project Presentation

Near the end of the term, you will give a demo of your project in a 3-minute presentation to the class and an evaluation panel.