AI Engineer assignment

Overview

The overall objective of this assignment is to design and build a containerised, asynchronous backend service to detect and track a given logo across a video. An optional frontend application can be built to interact with the service.

The purpose of the assignment is to assess your ability to design and implement a robust end-to-end AI solution. We will be looking at your problem-solving skills, your approach to the computer vision task, and your code quality in terms of software engineering best practices.

Context

The problem is framed within a neuromarketing context, where a user (e.g., a Creative Designer or Insights Manager) uploads their creative asset (e.g., a video) to our platform. To provide granular insights related to brand attention throughout the video, it is essential to identify when and where the brand's logo appears. Given that videos consist of a large number of frames, this process must be automated.

For this task, a user would provide their brand assets (like a logo) and the video to be analysed. The final output should be a version of the input video with bounding boxes overlaid on each frame where the logo is detected.

In practice, the logo may be extracted from a brand kit uploaded by the user. A brand kit is provided for context, as it would be in a real-world scenario. For this assignment, use the provided “neurons_logo.jpg” as the reference image for detection and tracking.

Instructions

- In terms of preferred tech stack, the backend part should be implemented using Python and FastAPI, whilst the optional UI can be implemented using Streamlit or Gradio.
- You are free to employ any libraries you deem suitable (e.g. OpenCV, Pillow).
- You do not need to train a deep learning model from scratch. Pre-trained models or classical computer vision techniques (e.g. template matching) are perfectly valid approaches.
- Prior to the technical meeting, please send us the link to a repository (e.g. GitHub).
- Any optional enhancements or features are marked as [**optional**] below.

Technical interview

- Code walkthrough of the core components and the results obtained with the 4 short videos provided. A demo of the application using Gradio/Streamlit would be much appreciated.
- A brief explanation of your technical approach, algorithm choice, and any assumptions made.
- A brief discussion of limitations and potential future improvements.

Detailed requirements

Available material

- Neurons brand kit (PDF) provided only for context.
- Neurons logo (**neurons_logo.jpg**) provided as the reference image for your detection algorithm.
- 4 short sample videos (**video_1.mp4 - video_4.mp4**) from Neurons Marketing team for testing your solution.

Proposed workflow

- The backend should be designed as an asynchronous service to handle time-consuming video processing without blocking or timeouts.
- The main steps encompass the following:
    - Upload a logo file, along with a video asset (MP4 format).
    - Detect the logo on video frames using the provided logo and label them using bounding boxes.
    - The FastAPI service returns the video with overlaid bounding boxes.
    - [**optional**] Render the video with overlaid bounding boxes in the frontend app (see section below).

Backend service

- Develop a RESTful API using FastAPI that implements the asynchronous workflow described above.
- Implement the video processing logic asynchronously.
- Include proper error handling (e.g., for invalid file types or failed processing) and status reporting through the API.
- Include proper logging for debugging and monitoring.
- [**optional**] Write unit tests for key functionality (e.g., API endpoint logic, a core utility function). We value quality and testing, but full coverage is not expected.

Frontend app [optional]

- Design an intuitive user interface with drag-and-drop file upload capabilities using Streamlit or Gradio.
- Use a docker-compose framework to deploy both the frontend and backend services.
- If you build the frontend, it should demonstrate the asynchronous nature of the backend (e.g., show a loading/processing state while waiting for the result).

Deliverables

- Code repository with the following:
    - Source code with clear documentation.
    - Setup instructions for local development and deployment of backend service.

Thank you very much in advance for investing your time and efforts in working through this assignment. Please schedule the technical meeting at your own convenience, and do not hesitate to let us know if you have any questions.