API(s) Requirements Document **By Austin Whitesky**

API for image depositing

API for checking for new images

API for checking for new weights

Introduction

This document outlines the requirements for designing an API to deposit data (JPEG image packets) into a server running YOLOv5 with ML Flow. The API will facilitate storing image data in a structured manner to enable future interaction such as automated training based on specific criteria.

API(s) Overview

The API(s) will allow users to deposit JPEG image packets into the server for further processing and analysis using the YOLOv5 model with ML Flow. It will provide endpoints for uploading new image data and querying existing images for automation purposes.

Functional Requirements

1. Upload Image Data
    - **Endpoint:** `/upload`
    - **Method:** POST
    - **Request Payload:**
        - JPEG packets
    - **Response:**
        - HTTP Status Code: 200 (OK)

2. Check for New Images (to be implemented at a later date)
    - **Endpoint:** `/check_new_images`
    - **Method:** GET
    - **Response Payload:**
        - New image data (if present)
    - **Response:**
        - HTTP Status Code: 200 (OK)
        - Message: “New images found” or “No new images found”
            - If new images found; query for them

3. Check for New Weights (to be implemented at a later date)
    - **Endpoint:** `/check_new_weights`
    - **Method:** GET
    - **Response Payload:**
        - TBD (if present)
    - **Response:**
        - HTTP Status Code: 200 (OK)
        - Message: “New images found” or “No new images found”
            - If new weights found; query for them

Non-Functional Requirements

1. Scalability
    - The API(s) should be able to handle a large number of image uploads simultaneously, as well as handle acquisition of weights & images simultaneously.
2. Security
    - Ensure secure transmission of data using HTTPS.
    - Implement authentication mechanisms to restrict access to authorized users.
3. Reliability
    - The API(s) should have minimal downtime to ensure continuous availability for users.

4. Performance
    - Optimize API(s) endpoints for efficient image data and weights processing and storage.

Data Storage Requirements

- Store image data in a structured format (e.g. file system with naming conventions or a database) for easy retrieval and management.
- Implement versioning or timestamping mechanism to track the status of images (i.e. whether they have had training performed or not).

Conclusion

The proposed API(s) will provide team members a means to deposit JPEG image packets into the server for YOLOv5 training. They will support functionalities such as uploading image data, checking for new images, and checking for new weights. Additionally, the API(s) will adhere to non-functional requirements such as scalability, security, reliability, and performance.