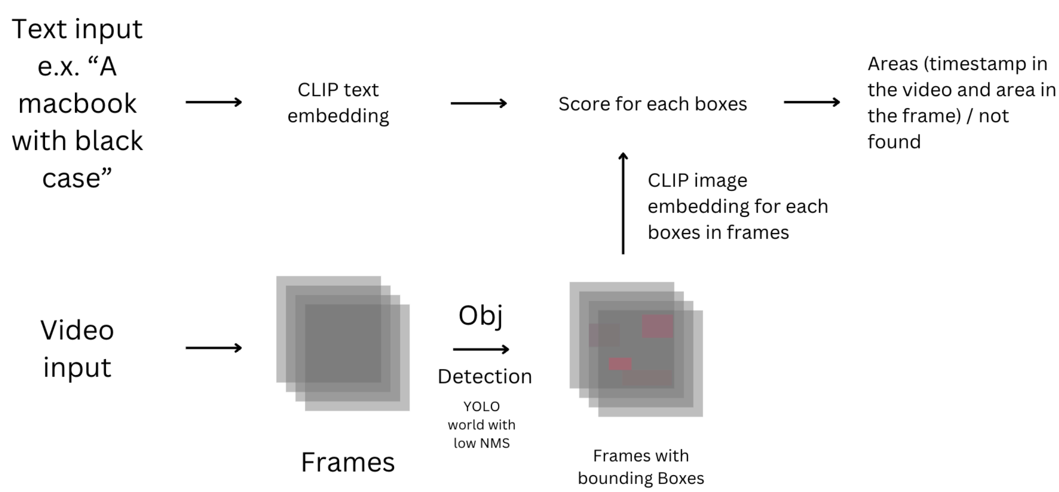
ML System Design Template

1. Problem Formulation: Find lost items in large spaces.
    - Clarifying Questions
        1. What types of items are commonly lost and need to be found?
        2. How varied are the environments where the system needs to operate? (e.g., different room types, lighting conditions)
        3. Will the system operate in real-time, or will it analyze static images?
        4. What hardware is available for capturing images? (e.g., smartphones, dedicated cameras)
        5. Images or Video
        6. What should the user give us as input? Text or Images of the item
        7. What will be the output to users? It can be segmentation, arrow points to the object?
    - Use Cases and Business Goals
        1. Finding lost objects in a large environment.
    - Requirements
        1. Input Requirement: Images or video feed of the room where the item is lost.
        2. Output Requirement: The location within the image where the item is found or a notification if the item is not detected.
        3. Performance Requirement: IOU > 30%, processing time less than 1 mins
    - Constraints
        1. Hardware Constraints: Dependence on the quality of the camera and processing power of the device used.
        2. Environmental Constraints: Performance variability based on room clutter and lighting conditions.
        3. Data Privacy Constraints: Handling of personal or sensitive images especially in environments like bedrooms or personal spaces. – Process locally not in cloud
    - Data: Sources and Availability
        1. Training Data: A dataset of images containing various items, labeled with their locations within those images. This can include public datasets or a custom dataset created by capturing images from various rooms.
        2. Video of the environment.
    - Assumptions

    i. The environment is static, meaning items aren't being moved during the search.
    ii. The item is visible at least partially in the camera feed and not completely obscured.
- ML Formulation
    i. Task: Object detection.
    ii. Model Choice: Use a pre-trained object detection model like YOLO, SSD (Single Shot Multibox Detector), or Faster R-CNN which can be fine-tuned for specific items if necessary.
    iii. Training Process: Train the model on a labeled dataset where common items are marked with bounding boxes.
    iv. Inference: The model processes the input image or video feed, applies the detection model, and outputs the coordinates of the detected items.

2. Evaluation Metrics
    - Offline: Precision, Recal, IOU
    - Online: Processing Time, Found/Not Found
3. Architectural Components
    - High level architecture



4. Data Collection and Preparation
    - Data needs
        i. Images with bounding boxes and descriptive classes for model to find e.x. ‘Black hat on the top of the middle man’
    - Data Sources
        i. I am using DDD dataset which mainly focuses on descriptive object.

    ii. For annotation, DDD dataset has provided use bbox, label,a and everythings.
  - Data storage
    i. For images I keep them on Google Drive as jpg.
    ii. For labels I keep them also Google Drive as json.
  - ML Data types
    i. Image
    ii. Text (of the target object)
  - Labeling
    i. Labels contain Text (of the target objects), list of bounding boxes

5. Feature Engineering
  - Feature representation
    i. We use nlp to embed expression then cluster them to select more appropriated clusters to use
  - Feature preprocessing
    i. Image resizing, normalization, cropping
    ii. Tokenized the label and cluster them to filter out bad label
    iii. Model Development and Offline Evaluation
  - Model selection
    i. Using only Yolo world
    ii. Pretrained yolo world with clip
    iii. Fine tune yolo world with clip
  - Dataset construction
    i. We have train.csv and test.csv for keeping annotation and file paths
    ii. We have images.zip for keeping both train and test images
  - Model Training
    i. We train YoLo-CLIP models by having them together and use 2 losses to add up together.
  - Model eval and HP tuning
    i. We evaluate the model using a test set with 1400 images.
  - Iterations
    i. Improving model and experiment with architecture]

7. Prediction Service

  - Find specific objects in video (maybe from cctv records) with natural language.

8. Online Evaluation and Deployment

  - Found/Not Found rate
  - Correct object rate, precision@K (show most k possible area detected)