Technical Report

This report contains every technical detail of my final project for Practical Machine Learning including the problem I want to solve, approach I choose to do, every decision I made through this project, and other technical details related to this project.

The problem

We sometimes cannot find things we need. Whether in a large space or small messy room, things always find unexpected places to hide themselves away from us. I decided to fight this unfortunate fate of humanity using machine learning! By combining advancement in computer vision and language model, I have developed a detection system in which you can input free-expressed object descriptions such as “white laptop with many stickers on the back” to the model and it understands what to keep an eye on.

You may ask why it is important in a business context. It would be nice if we can add more “understanding about the world” in object detection models, which may make it easier to implement computer vision in business. Example, a coffee shop can input “empty glass” to the model and it will keep an eye on it without the training model for this specific task.

To keep it simple, my system will receive a video and a text as inputs. The output will be the top three most possible to have the object frames, each with a bounding box around the detected object, that contain the object described in the text.

Selected dataset

As I focus on finding objects with specific description, I am looking for a dataset that not only provides a short class name but the class name that labels objects with detailed description.

I first found the dataset from InstructDET: Diversifying Referring Object Detection with Generalized Instructions which provide InDET dataset created by involving emerging vision-language model (VLM) and large language model (LLM) to generate instructions guided by text prompts and object bounding boxes. However, I found out that this dataset has an expression label that seems to be out of scope for this project because it focuses on refereeing object expressions like ‘Object in front of the guy with white hat’ which will require a more complex model to learn this task.

Finally, I found a Description Detection Dataset. Unlike traditional detection datasets, the class names of the objects are no longer simple nouns or noun phrases, but rather complex and descriptive, such as a dog not being held by a leash. For each image in the dataset, any object that matches

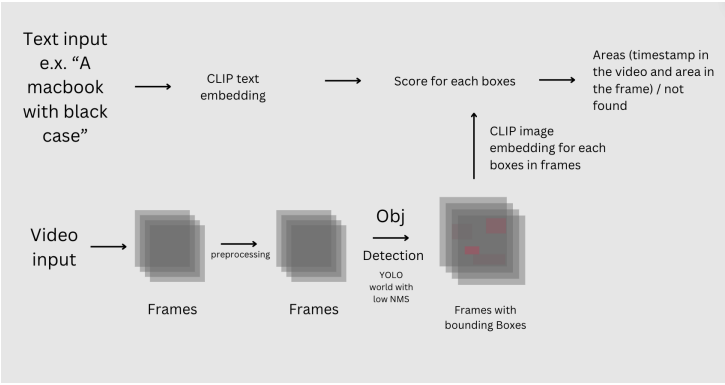
the description is annotated. At the left, you can see one of the images in the dataset which labels a toy by describing that it's a light on in the room.

Data preprocessing

Even though the selected data seems to fit well into my requirement, it has some parts containing annotated images which might not need to train the model to find described objects in video. For example, some annotations are about animals and some are about humans. As I need to process around 20 thousand for annotation in the dataset, I decide to transform those text labels in each annotation into vectors using a tokenizer from Spacy library. After reducing their dimensionalities using TSNE, I clustered them using KMeans algorithm, and then filtered each cluster out by investigating the label in each cluster manually. The results of this process is in the table on the left which shows example labels (expression column) from 7th cluster.

	expression	label	x	y
212	wild lioness	6	-0.609946	-20.425186
8	basketball untouched	6	2.373617	-15.772728
90	clothes hanger	6	-7.644678	7.093328
302	fake peacock	6	0.714620	-18.480389
48	incomplete donut	6	-3.076991	-17.983952
61	incomplete apple	6	0.346156	-18.739069
216	parrot eating	6	12.966591	-6.571068
49	chocolate donut	6	-3.482907	-17.649300
96	open package	6	-0.187884	-19.520372
95	backpack	6	-4.727466	-14.798507

Selected method



My selected method in the diagram combines CLIP (Contrastive Language-Image Pretraining) and YOLO (You Only Look Once) to locate specific objects in video frames based on the description (text input). The process starts with the text input, such as "A MacBook with black case," which is embedded using CLIP's text embedding model. At the same time, the video input is divided into individual frames, which then will be processed by YOLO with a low NMS, identifying objects and marking them with bounding boxes. Next, CLIP's image embedding model processes each bounding box within the frames, generating embeddings that represent the visual features of the detected objects. These detected image embeddings are then compared with the text embedding to generate a score for each bounding box then show the top 3 frames with the most score to the users with one detected bounding box for each frame around the predicted object.

Experiments

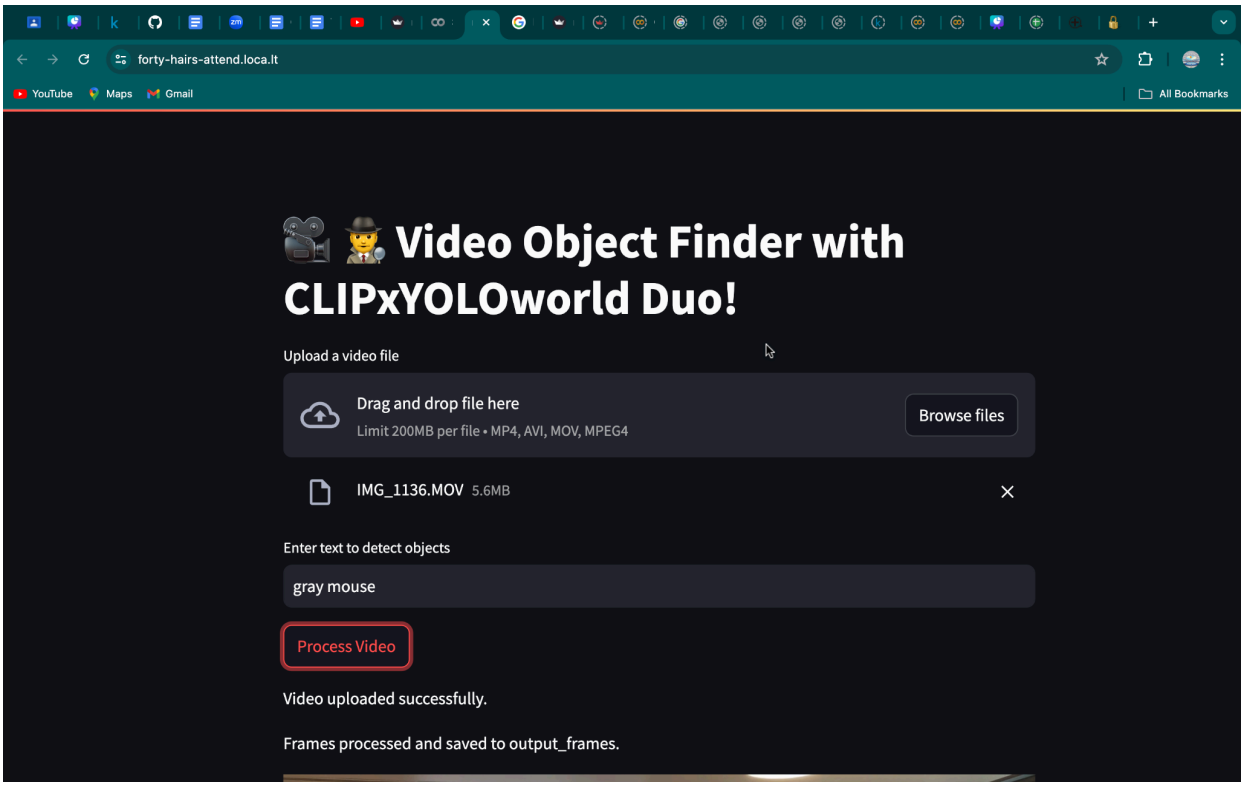
I decided to do an experiment with 3 model combinations: 1) use only pretrained Yolo world 2) use pretrained Yolo world with pre-trained CLIP 3) fine tune Yolo and CLIP with processed

Description Detection Dataset. For evaluation, I decide to process only one image for each annotation. By calculating the average IoU over all test set for each model combination (code for evaluation can be found in my kaggle notebook), I will choose the best combination to deploy in production.

Aside from model related, I will use the Streamlit library to create the UI of my demo which will be able to accept text input and video input, and use Google Colab to be the interface to start the Demo which we can utilize GPU for free.

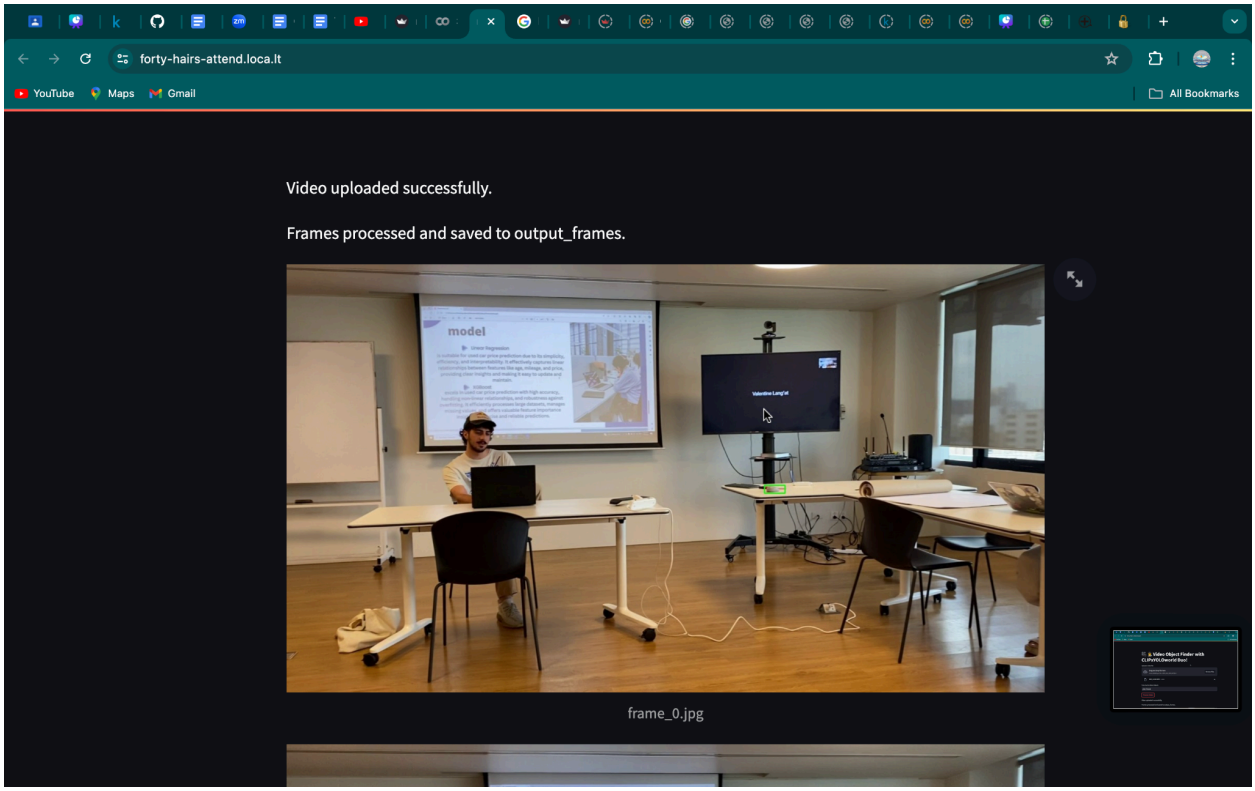
Results

Comparing 3 combinations, the best one is to use both pretrained models without any fine tuning with average IoU at around 41 (40 for fine tune and 30 for only YoLo world). You can see images from my Demo below.



In the image, I used video captured on the final day of Practical Machine Learning class during Nicolus's presentation. My text input was 'gray mouse', so the model should keep its eyes for

the gray mouse which was on the table next to the one that Nocolus was sitting on during his presentation.



We can see that the model found the mouse! It’s really on the table next to the one Nocolus was sitting on. However, my system takes more time that I expected it to use even though I process video with 3 FPS (from original around 30 FPS).

Finally, It works well as I set the requirements. However, there is a lot of room to improve such as reducing processing time, more context aware (things around the object) detection, and many more! I really want to improve this more as I think it might have potential to be commercialized. Hope I can find some specific domain that needs something like this and I can find a way to make the system work on a larger scale !

END OF THE REPORT