Efficient Video Analytics

YAN Yuhang *StudentID*
CSE, CUHK
yhyan2@cse.cuhk.edu.hk

Supervisor: Eric Chi Lik LO

1 Introduction

Our ultimate goal was to create an accurate and efficient lost-and-found system for Hong Kong Airport. By inputting a linguistic or pictorial description of the owner and the lost item, our system can quickly find the moment the item was lost in a large number of videos and return the video of that moment. Therefore, our task is divided into two main areas: designing a front-end user interface that is friendly to people without programming knowledge and writing the back-end code that can achieve the desired functionality.

Since this could be a long-term project, our expectations for the project before the start of Summer Research were to complete the front-end UI design, implement the basic functionality, and improve the existing programme's performance as much as possible in the limited available time.

2 Previous Works

2.1 Object Detection

Object Detection serves as a crucial research pathway in the field of computer vision. Its primary goal is to recognize and pinpoint distinct objects within images or videos. This domain is composed of two key subtasks: object classification and object localization. The former focuses on discerning whether an image encompasses a particular object, whereas the latter zeroes in on identifying the precise location of the said object within the image.

2.2 Video Analysis

Zelda Zelda[1] is a video analytics system that delivers relevant and semantically varied results for Top-K queries on extensive video datasets. It employs Vision-Language Models (VLMs, hybrid models that interpret and generate natural language descriptions of visual content) and leverages the user's natural language queries and supplementary terms to enhance precision and pinpoint low-quality frames.

Everest Everest[2] is a Top-K video analytics system that supports efficient and accurate querying. It identifies the most interesting frames/clips from videos with probabilistic guarantees. Everest uses deep learning models to rank frames and combines techniques from computer vision, uncertain databases and Top-K processing. It leverages a combination of fast but less accurate models and slow but precise models to efficiently return results.

2.3 Legacy Code

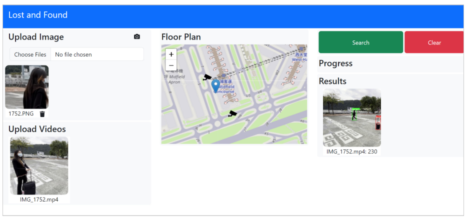
The computer vision models used by the developers before we took on this project included YOLOv5[3] and Deep SORT[4], which can simultaneously detect and track the target object, but this approach has some limitations. For example, the YOLOv5[3] has low accuracy in recognizing small objects and cannot realize the expected open vocabulary object detection function.

3 Our Approach

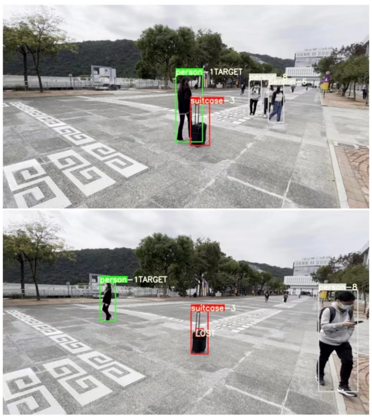
3.1 Front-end UI

Our project is bootstrapped with Create React App[5]. This design ensures the stability of the app, its adaptability to different devices, and the synergy of multiple developers working in parallel.

Our interface allows users to upload photos of individuals and on-site recorded videos from their local devices, with support for simultaneous multi-video uploads. Alternatively, users can select a specific camera on the airport map to retrieve surveillance footage. Upon clicking "Search", a video is generated, presenting surveillance clips before and after the item's loss, highlighted with bounding boxes around the person and the lost item. Furthermore, a "LOST" sign appears from the moment of item loss to notify the user.



(a) The web UI page



(b) Video test results

Fig. 1: User interface and results

3.2 Formulation of the Lost and Found Problem

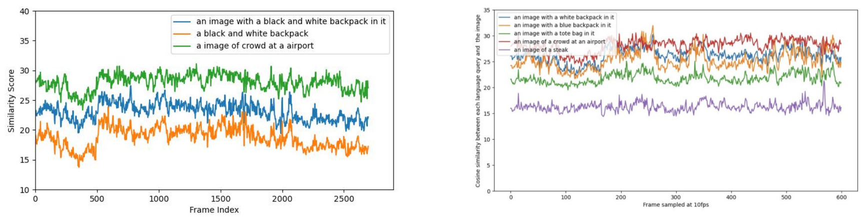
In the prior codebase, the lost-and-found issue was addressed using the pixel distance between the centers of the bounding boxes of the person and the object. However, this approach has a limitation: pixel distance may not accurately reflect the actual distance, and estimating physical distance might necessitate supplementary deep learning models. Therefore, we argue that for an airport use case, where footage is likely target-sparse (i.e., the target person and their luggage appear in a small fraction of frames), this distance-based formulation may not be optimal. We propose a “co-occurrence formulation” as an alternative: we identify all frames where the person and the luggage appear together. Instead of employing distance to pinpoint the moment the luggage was lost, we simply locate the last moment of their co-occurrence.

However, due to privacy issues, we don’t have access to the real Hong Kong Airport’s CCTV footage yet, and due to the limitation of the equipment, we can’t record a large number of sample videos by ourselves for testing. Therefore, our demo version still mainly uses the “Center-Distance Method” to demonstrate the function.

3.3 Zero-Shot Object Detection

Zero-Shot Object Detection models[6] can identify and locate objects within an image based on either textual or visual input, without requiring any category-labeled training samples. However, frame-wise systems like CLIP[7], as used in Zelda[1], fall short when processing queries demanding object-level information - for instance, they cannot count the number of objects within a frame, nor locate a specific object in a densely-populated video, such as airport footage.

After reviewing the latest literature on large vision language models, we ultimately selected OWL-ViT[8] (Vision Transformer for Open-World Localization), a zero-shot object detection model. Following extensive tuning and testing, we concurred that this algorithm best aligns with our current design goals and delivers robust performance.



(a) CLIP (b) OWL-ViT

Fig. 2: Detection ability of different models

3.4 Current Person and Object Detection System

By inputting the test video and the corresponding language/image query, we can return a JSON file containing the results of the query (the number of frames where the object is located, the class it belongs to, and its score in the recognition process; the higher the score, the higher trustworthy the result is), as well as video frames labelled with the corresponding bounding box.

3.5 Dataset

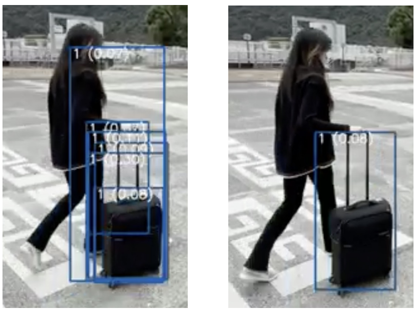
We mainly utilized a vlog[1] downloaded from YouTube recorded by a mobile phone at the airport and videos recorded by the developers ourselves on the campus of the Chinese University of Hong Kong in the early stages of the project.

3.6 Testcase design and Results

For some pre-trained models like YOLOv5[3], they can only recognise a limited number of classes, whereas OWL-ViT[8] can really do open vocabulary object detection. To test this feature, we designed multiple testcases based on the colour, size or other features of the object, which may be complicated for YOLOv5[3], for example, a black and white striped backpack, a man wearing black clothes and so on. On these queries, OWL-ViT[8] shows surprising accuracy.

3.7 Non-Maximum Suppression

We ran the OWL-ViT[8] model directly on the video and there was a large amount of overlap in the objects recognized. We then manually integrated the Non-Maximum Suppression algorithm into our model, which simplified multiple recognition bounding boxes caused by the same object into one.



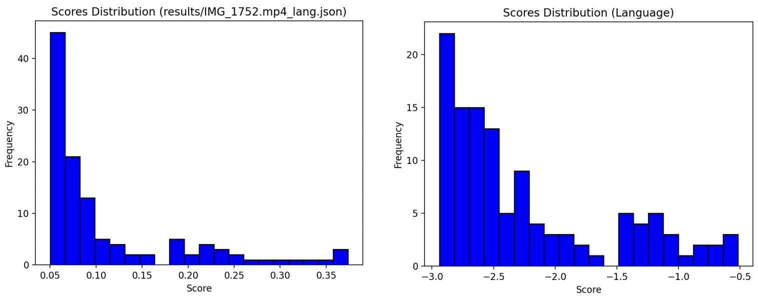
(a) Before NMS (b) After NMS

Fig. 3: Before and after Non-Maximum Suppression

[1] Hong Kong Airport 2023 Post-Covid Walkthrough (One of the Best Airports in the World!) (4K) https://www.youtube.com/watch?v=ZgxirOW9_go

3.8 Scores and Cosine Similarity

The object scores predicted by our model are relatively close to each other, which is not conducive to improving accuracy at a later stage. The original model will be based on the feature distance between the content of the language/image query and the actual content of the video, i.e. "Cosine Similarity". And the original model will further map this cosine value to the probability on the interval of 0 to 1 through the `sigmoid()` function, which reduces the gap between the scores of different objects. We then switched to using the Cosine Similarity directly as the score for the prediction result.



(a) Scores (b) Cosine similarities

Fig. 4: Distribution of scores under different calculation methods

3.9 Top-K Selection

Our model had a high False Positive rate for the results of video runs. So we manually added the Top-K Selection algorithm to the model. Based on the "Cosine Similarity" of the detected objects, we output the K frames most likely to contain the object from highest to lowest.

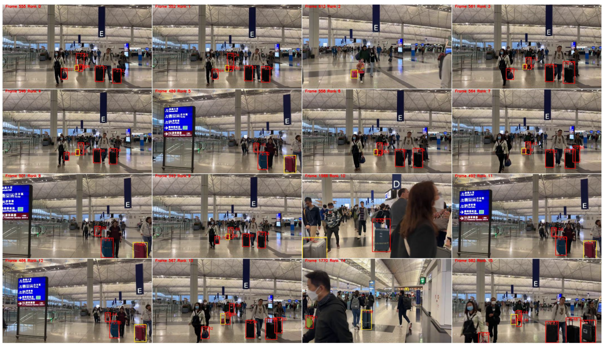


Fig. 5: The yellow boxes are the Top-16 of the image detecting result of a pink box; all the red and yellow boxes are part of the full test results

3.10 Chunk Extraction

Our current model uses OWL-ViT[8] to score each bounding box, selecting the highest score per frame as the frame's score, and ranking these to filter the frame with the highest score. We are developing new algorithms that rank and output chunks instead of frames, by using the average of the scores of all the frames within a chunk as the basis for sorting.

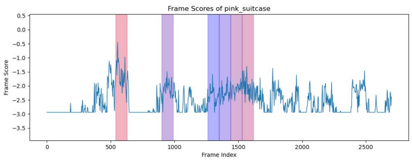
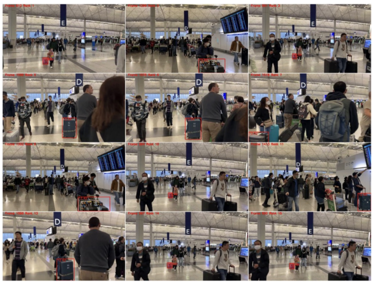


Fig. 6: Top-5 scoring chunks
(Each chunk is 3s long; The redder the colour, the higher the score)

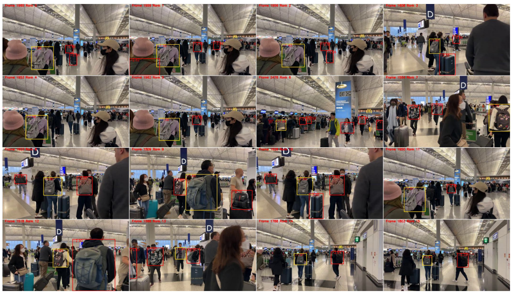
4 Conclusion

Now, we can run either an image or language query on a video and pick out frames or chunks containing objects with Top-K scores from the result.

(a) An image testcase



(b) Image query



(c) Language query: black and white striped backpack

Fig. 7: Examples of image/language queries

5 Future Work

We aim to utilize the “detecting co-occurrence” method. We believe its implementation could significantly enhance our model’s ability to identify the “LOST” status of the items. Additionally, we plan to incorporate Everest into our model. This integration is expected to boost our model’s precision and processing speed, potentially enabling real-time data analysis. These improvements will hopefully lead to more accurate and faster data analysis.

References

1. Romero, F., Winston, C., Hauswald, J., Zaharia, M., Kozyrakis, C.: Zelda: Video analytics using vision-language models (2023)
2. Lai, Z., Han, C., Liu, C., Zhang, P., Lo, E., Kao, B.: Top-k deep video analytics: A probabilistic approach. In: Proceedings of the 2021 International Conference on Management of Data. SIGMOD ’21, New York, NY, USA, Association for Computing Machinery (2021) 1037–1050
3. Jocher, G., contributors: Yolov5. https://github.com/ultralytics/yolov5 (2020)
4. Wojke, N., Bewley, A., Paulus, D.: Simple online and realtime tracking with a deep association metric. In: 2017 IEEE International Conference on Image Processing (ICIP), IEEE (2017) 3645–3649
5. Facebook Inc.: Create react app. https://github.com/facebook/create-react-app Accessed: 2023-06-15.
6. Bansal, A., Sikka, K., Sharma, G., Chellappa, R., Divakaran, A.: Zero-shot object detection. In: Proceedings of the European conference on computer vision (ECCV). (2018) 384–400
7. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I.: Learning transferable visual models from natural language supervision. CVPR **abs/2103.00020** (2021)
8. Minderer, M., Gritsenko, A., Stone, A., Neumann, M., Weissenborn, D., Dosovitskiy, A., Mahendran, A., Arnab, A., Dehghani, M., Shen, Z., Wang, X., Zhai, X., Kipf, T., Houlsby, N.: Simple open-vocabulary object detection with vision transformers (2022)