NANYANG THE INNOVATIVE POLYTECHNIC

2024S2

ITI107 Assignment

Introduction

In this assignment, you will apply the knowledge and skills that you learned about Object Detection to train your own object detector for custom objects.

Tasks

This is an **INDIVIDUAL** assignment.

For this assignment, you are expected to fine-tune a pretrained Object Detection model to detect **2 classes** of objects you selected. Both object classes **SHOULD NOT** be from COCO object class (refers to https://github.com/amikelive/coco-labels/blob/master/coco-labels-2014_2017.txt).

The following are the tasks you will need to complete:

- Collect and annotate the images using either Roboflow or LabelStudio.
- Train the model and evaluate the model performance based on mean average precision.
- Log your experiments using tools like Weights & Biases (wandb).
- Improve your model through fine-tuning your dataset, hyper-parameter, data augmentation, and any other techniques and compare the model performances.
- Export your model to a most performant format for a chosen platform (e.g. CPU or GPU) capable of performing fast inference on ONE selected test image and one test video (around 1 minute duration).
- Deploy the model to HuggingFace space (refers to Appendix for instructions)

Submission

You are to submit the following, in a single zip file, to POLITEmall:

1. A report (maximum 5 pages, font size 12, single column) per team that contains the following:
   - discussion of your data collection and annotation process and include the link to your dataset/annotation.
   - The finetuning process to improve your model performance and comparison of model performance.
   - screenshots of your experimental log on weights and biases
   - link to HuggingFace space
2. ONE test image and ONE test video file (.mp4) with bounding boxes (produced by the best model)

Grading Rubric

Total Marks: 45 marks

Criteria	Below Expectation	Meeting Expectation	Exceeding Expectation
Data Collection	0 - < 7.5 marks	7.5 – < 12 marks	12 – 15 marks
	• Minimum data collection effort (e.g., less than 20 images per class or only a single class), • Use existing public annotated dataset, or irrelevant dataset. • Poor annotations effort with many of bounding boxes misaligned.	• Good data collection effort (more than 20 images per class). Dataset is somewhat relevant to final detection task. • Data is sufficiently annotated with mostly correct bounding boxes.	• Good data collection effort (e.g., more than 60 images per object class). • Dataset is relevant to and representative of final detection task. • Data is well-annotated with accurate bounding boxes
Model Training	0 – <10 marks	10 – <16 marks	16 – 20 marks
	• Minimum effort in model training using a pretrained model and with minimum training steps. The trained model does not detect most of the time or no detection. No evidence of efforts to improve model performance.	• Some effort in model training (e.g., train one model with enough training steps). • The trained model shows average detection precision (measured by IOU@0.5 mAP). • Some effort in improving the model performance.	• Good effort in experimenting with fine-tuning ONE pretrained model with enough training steps • Good effort in improving the model performance. • The trained model shows higher than average detection precision (measured by IOU@0.5 mAP).

Criteria	Below Expectation	Meeting Expectation	Exceeding Expectation
Documentation	0 – <2.5 marks	2.5 – <4 marks	4 – 5 marks
	• Writing is unclear or incoherent. • Lack of content or minimum or no discussion of results.	• Writing is reasonably clear and coherent. • The content is general and contains some discussion of experimental results.	• Writing is clear and coherent. • The content is specific with high-quality discussion of experimental results.
Deployment	0 – <2.5 marks	0 – <2.5 marks	4 – 5 marks
	• Model is not deployed or deployment with error	• Model is deployed successfully • Application is mostly functioning	• Model is deployed successful • Application is functioning with good UX

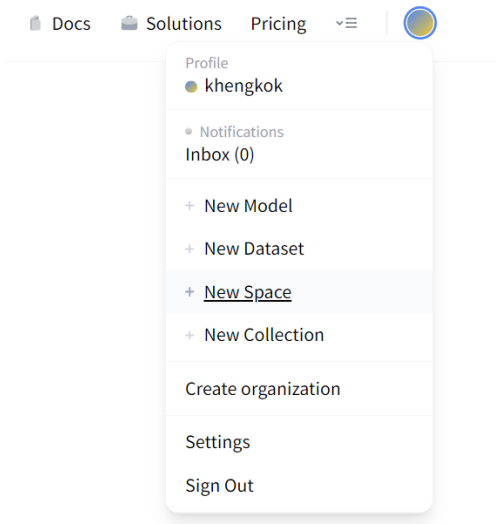
APPENDIX

Instructions to deploy to Hugging Face Space

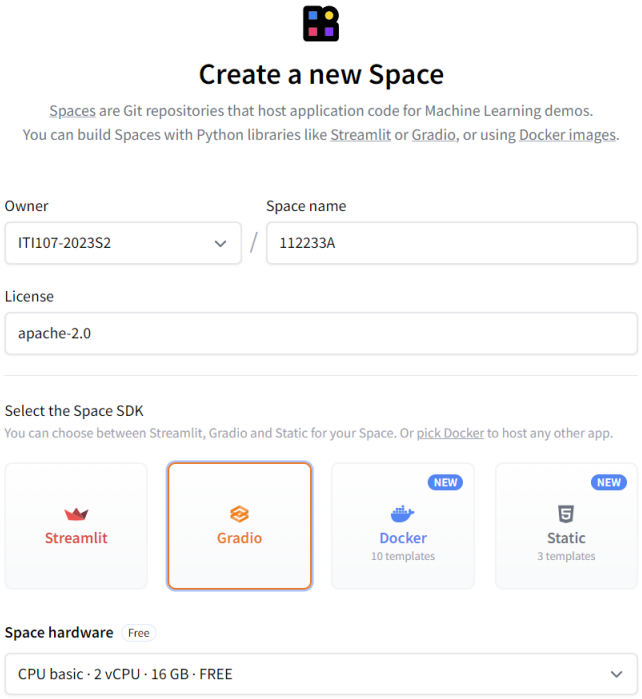
1. Create an account at Hugging Face: https://huggingface.co/
2. Join the organization **ITI107-2024S2** using this link:

   https://huggingface.co/organizations/ITI107-2024S2/share/qAtJtfEDVWcLOWzUcqfwXRnPcHMEDoDFND

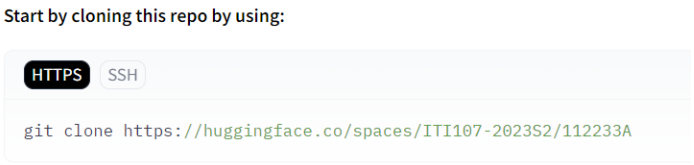
3. Click on your profile, and choose New Space:



4. In “Create New Space” page, choose “ITI107-2024S2” as Owner and use your student number for space name (e.g., 112233A), use Apache 2.0 License, and choose Gradio for Space SDK. Select Public for the space visibility. Click “Create Space”.



5. Follow the instructions to clone the git repository to your local PC.



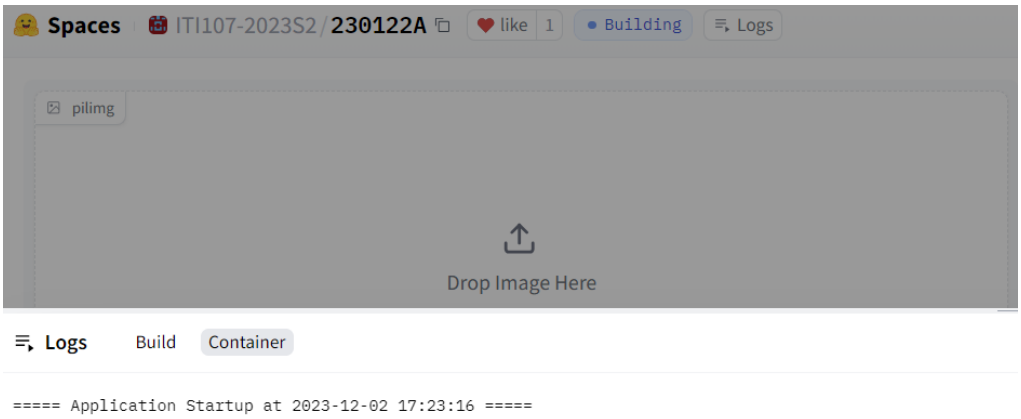
6. Create your Gradio app to demo your model in the repository and then commit and push.

   You can modify the codes from the following demo app:

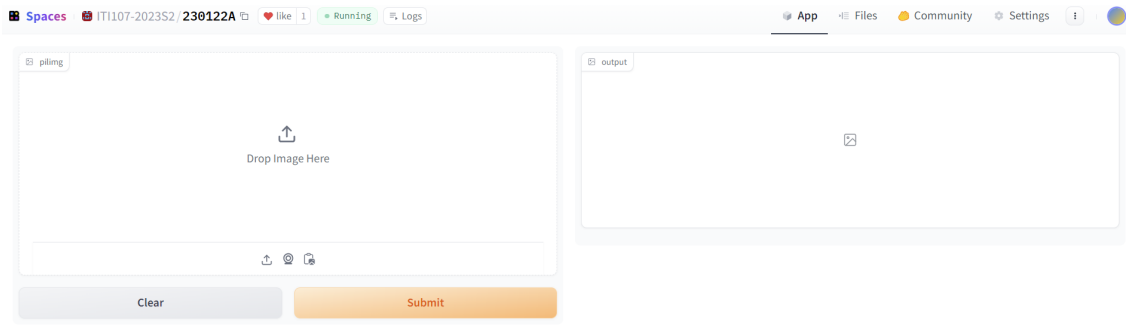
   https://huggingface.co/spaces/khengkok/balloon_od/tree/main

   Modify the codes, requirements file, README.md file accordingly. For the README file, change the title to your student number, e.g., 112233A.

   This will start the build process on HuggingFace, resulting in a docker container running your application. You can click on the space to view your build process.



7. Once the container starts up, you can now access your Gradio demo app:



8. If there are any changes to the code, just push the changes to repo and this will rebuild your app on HuggingFace space.