# Pose Bowl Spacecraft Detection

## 2nd place – DungNB

## I. Basic information for winner announcement

- Name: Dung Nguyen Ba
- Hometown: Hanoi, Vietnam
- URL: https://www.linkedin.com/in/dungnb1333/

## II. Model documentation and write-up

### 1. Who are you (mini-bio) and what do you do professionally?

I am a Principal AI Engineer/Tech Lead at FPT Smart Cloud, Vietnam. I have 8 years of experience in deep learning and computer vision, 1 year of experience in embedded software development and 5 years of experience in android application development. I am also a Kaggle grandmaster (6 gold medals, 4 wins)

### 2. What motivated you to compete in this challenge?

At first, this competition was interesting. Moreover, I want to expand my connections and look for opportunities after this competition.

### 3. High level summary of your approach: what did you do and why?

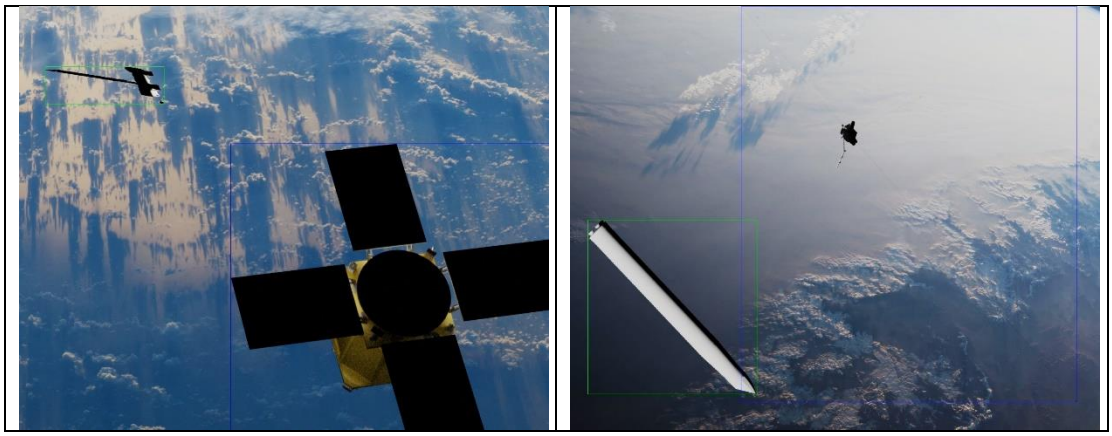
I choose yolov8s 1280 as suitable for the runtime constraints and high accuracy. The key of my solution is synthetic data generation, external data, post-processing.

#### 3.1. Synthetic data generation

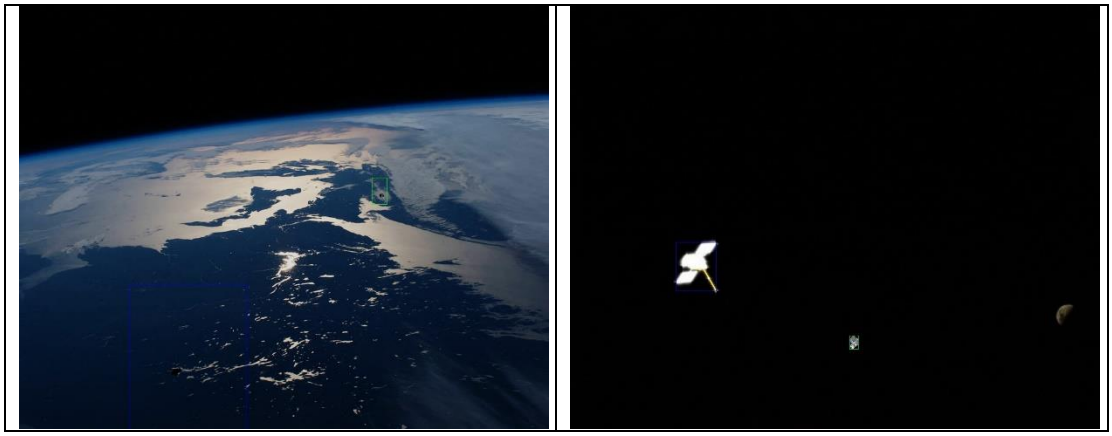
Since the competition provides spacecraft image without background, I thought of generate synthetic data by overlay spacecraft image with basic transform (rotate, random brightness contrast) to the background image. This increased my public LB score significantly from 0.83 to 0.9190, enough for me to win the top 3 private LB (0.9196) at the start of the competition.

I used the following 5 types of data generation:

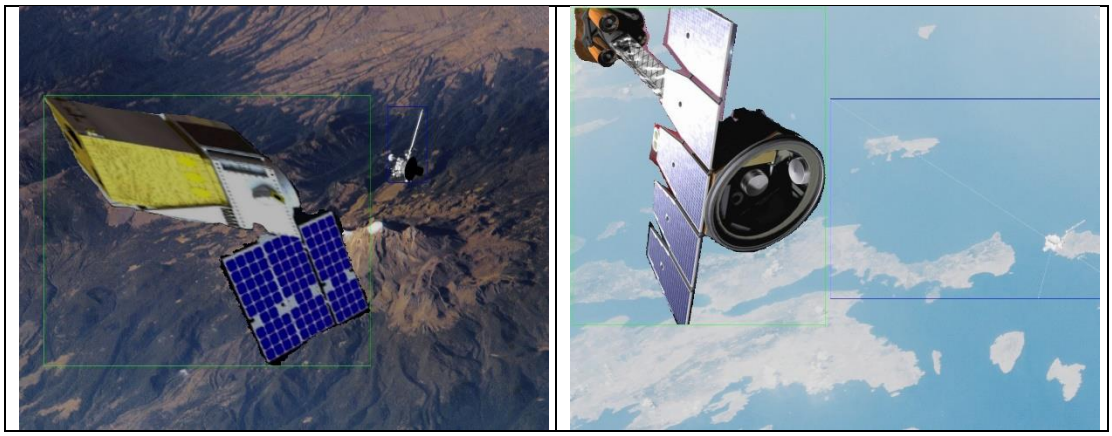
- randmix: Keep size of the spacecraft image, overlay spacecraft image to random position in the background image. Total 46652 images



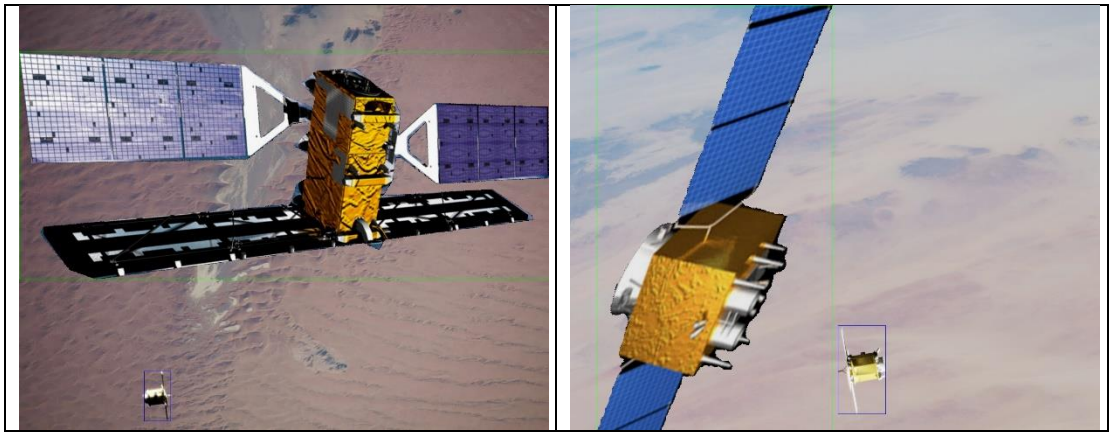
- randmix_small: resize spacecraft to random size (max width, height from 10px to 64px), overlay spacecraft image to random position in the background image. Total 25639 images



- randmix_big: resize spacecraft to random size (max width, height from 350px to max size of background image), overlay spacecraft image to random position in the background image. Total 16796 images



- randmix_max_size: resize spacecraft to max size of background image (width or height), overlay spacecraft image to random position in the background image. Total 2305 images



- add_antenna: fake antenna (white line or black line based on background color) to spacecraft image. Total 2385 images

*Note: blue is original box, green is synthetic box*

### 3.2. External data

The competition provided a very small spacecraft images (14 different spacecraft types), so to increase the generality of the model I searched for external public data that meet the competition rules. I found a CVPR paper, they provided 3100 satellite images with masks. After adding this external dataset, my score increased by 0.003 on public LB and 0.005 on private LB

Paper:
https://openaccess.thecvf.com/content/CVPR2021W/AI4Space/papers/Dun

g_A_Spacecraft_Dataset_for_Detection_Segmentation_and_Parts_Recognition_CVPRW_2021_paper.pdf

Github: https://github.com/Yurushia1998/SatelliteDataset

This external data was approved in discussion https://community.drivendata.org/t/external-dataset-use-detection-track/10642

### 3.3. Augmentation

- Random blur, median blur
- Random clahe
- Random grayscale
- Random flip (left-right, up-down)
- Random brightness contrast
- Mosaic

### 3.4. Dataset

| | Number of images |
|---|---|
| Driven data competition | 25,800 |
| Public external data | 3,117 |
| Synthetic data generation | 93,778 |
| Total | 122,695 |

### 3.5. Postprocessing in submission

Finally, I did some post-processing which helped my score increase about 0.003 both public and private score.

In prediction, choose box with highest confidence

If max(box_width/image_width, box_height/image_height) > 0.7 -> pad input image 150px.

I did this because from my experience from many previous object detection competitions, yolo is very bad with large objects (object size = image size).

### 3.6. Export model to openvino

The CPU inference speed of the openvino model is 3 times faster than original pytorch model.

### 3.7. Model

Yolov8 from ultralytics (https://github.com/ultralytics/ultralytics)
Input image size: 1280
Epoch: 70
Learning rate: 0.01
I tested 2 models of yolov8, yolov8s 1280 for high accuracy and a lighter yolov8n 1280 for speed bonus prize

| Model | Public LB | Private LB | Runtime | Rank |
|---|---|---|---|---|
| Yolov8s | 0.9285 | 0.9226 | 1 hour 40 minutes | 2 |
| Yolov8n | 0.9173 | 0.9098 | 46 minutes | 3 |

**4. Please provide the machine specs and time you used to run your model?**

- CPU (model): AMD EPYC 7742 64-core
- GPU (model or N/A): 1xNvidia A100 40GB
- Memory (GB): 64 GB
- OS: Ubuntu 22.04
- Train duration: yolov8s 1280 (~60 hours), yolov8n 1280 (~24hours)
- Inference duration: yolov8s 1280(1 hour 40 minutes), yolov8n 1280(46 minutes)