DEEP LEARNING-BASED ROCKET OBJECT DETECTION IN VIDEO

CAP 5415 FINAL REPORT, FALL 2022

Nashir A. Janmohamed
Department of Computer Science
University of Central Florida, BS
Orlando, FL 32826
`nashir@knights.ucf.edu`

ABSTRACT

This work introduces a single-class object detector trained to recognize SpaceX Falcon 9 rockets. We use the Tensorflow Object Detection API and fine-tune a RetinaNet model that was pretrained on the COCO 2017 dataset. We hand-label a small dataset of 33 images of rockets with bounding boxes. Our method is evaluated on a video of a SpaceX rocket launch.

Keywords Object Detection · Deep Learning · Transfer Learning · Astronautics

1 Introduction

This work introduces a single-class object detector trained to recognize SpaceX Falcon 9 rockets. We use the Tensorflow Object Detection API [Huang et al., 2017] and fine-tune a RetinaNet Lin et al. [2017] model that was pretrained on the COCO 2017 dataset Lin et al. [2014]. We hand-label a small dataset of 33 images of rockets with bounding boxes using LabelStudio Tkachenko et al. [2020-2022]. 26 of these images are used for training, and the other seven are held out for validation. The method is evaluated on a video of a SpaceX rocket launch.

We release the code as open source here.

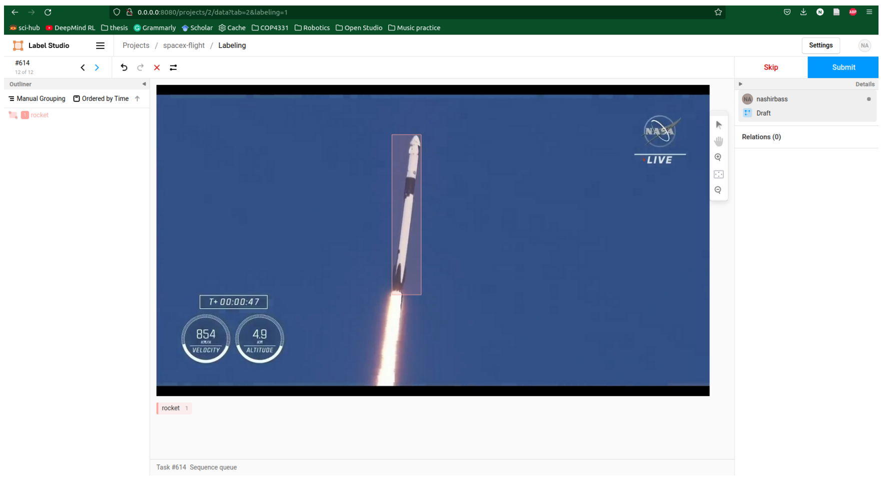

Figure 1: Labeling rocket data in LabelStudio

2 Method

2.1 Data curation

Since there was no suitable dataset available, we curated a dataset of rocket images from a video of a SpaceX rocket launch. We used OpenCV ope to iterate through the video frames, extracting every 50th frame to use as training/validation data. We then hand-labeled bounding boxes in the extracted frames with LabelStudio, shown in Figure 1.

LabelStudio labels are exported in the YOLO Redmon et al. [2016] format; "images" and "labels" directories are created. We randomly select approximately 80% of the data for training and the rest for validation, creating train and test directories for the corresponding images and annotations.

2.2 Fine tuning

Tensorflow provides a reference notebook for fine-tuning; we adapted the code from their example to fine-tune a RetinaNet model from the Tensorflow model zoo on our curated dataset of rocket images.

Using a batch size of 4, we train in two stages:

- 50 epochs with learning rate = 0.01
- 200 epochs with learning rate = 0.001

The loss (classification, localization, and combined) is shown in Figure 2.

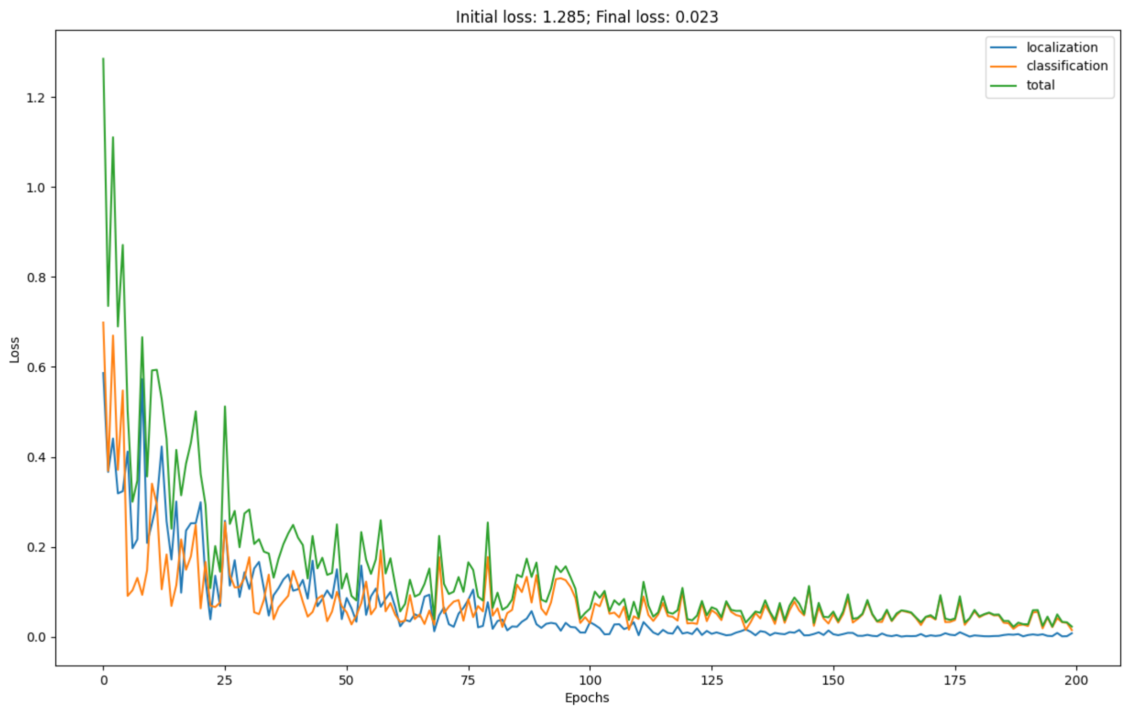

Figure 2: Model loss on training data.

3 Results

After the model is trained, we evaluated it on a few sample images (see Figure 3, 4, 5, 6).

Once we were satisfied with the performance, we ran the model on every image extracted from the original video, then stitched together the annotated frames using ffmpeg Tomar [2006]. The result is published here.

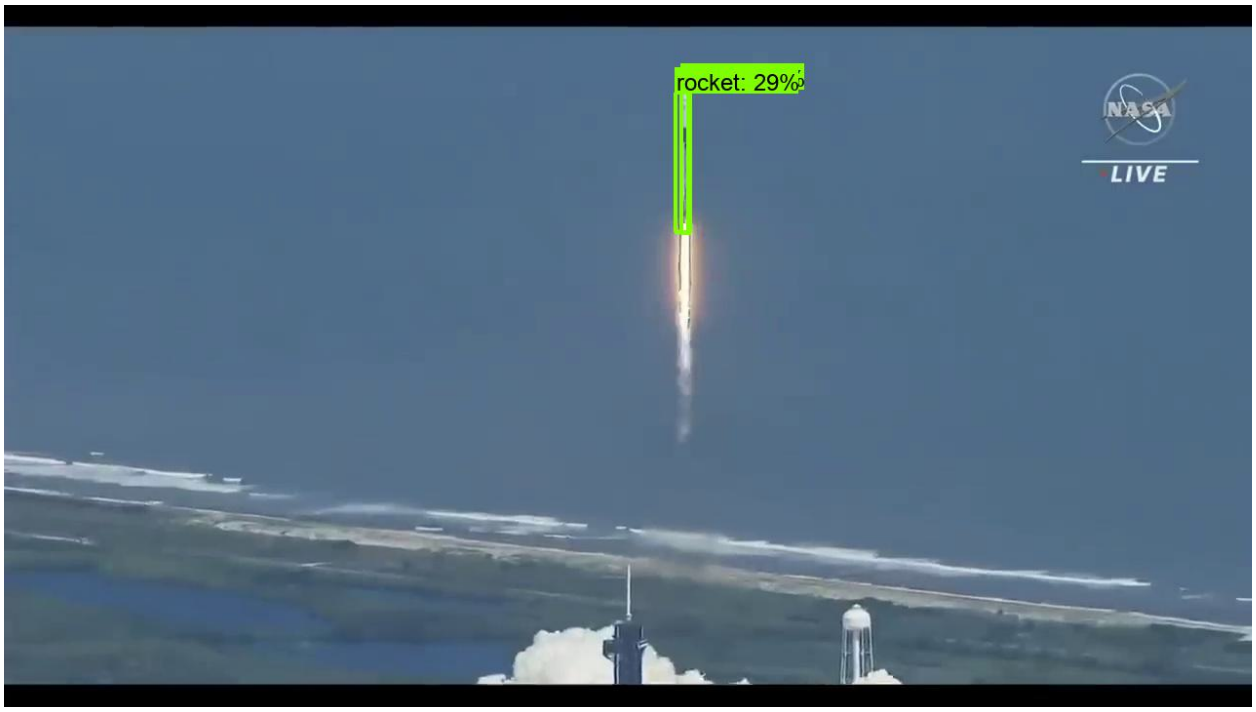

Figure 3: Sample detection, early flight.

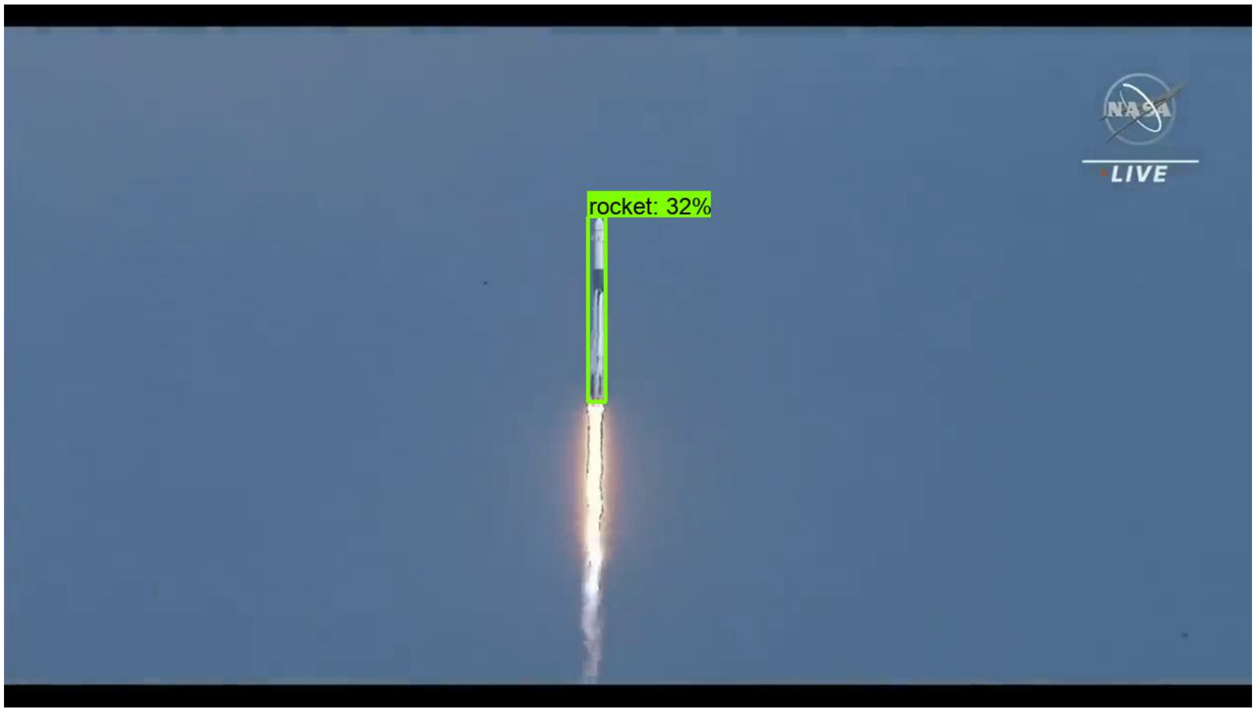

Figure 4: Sample detection, ground no longer in view.

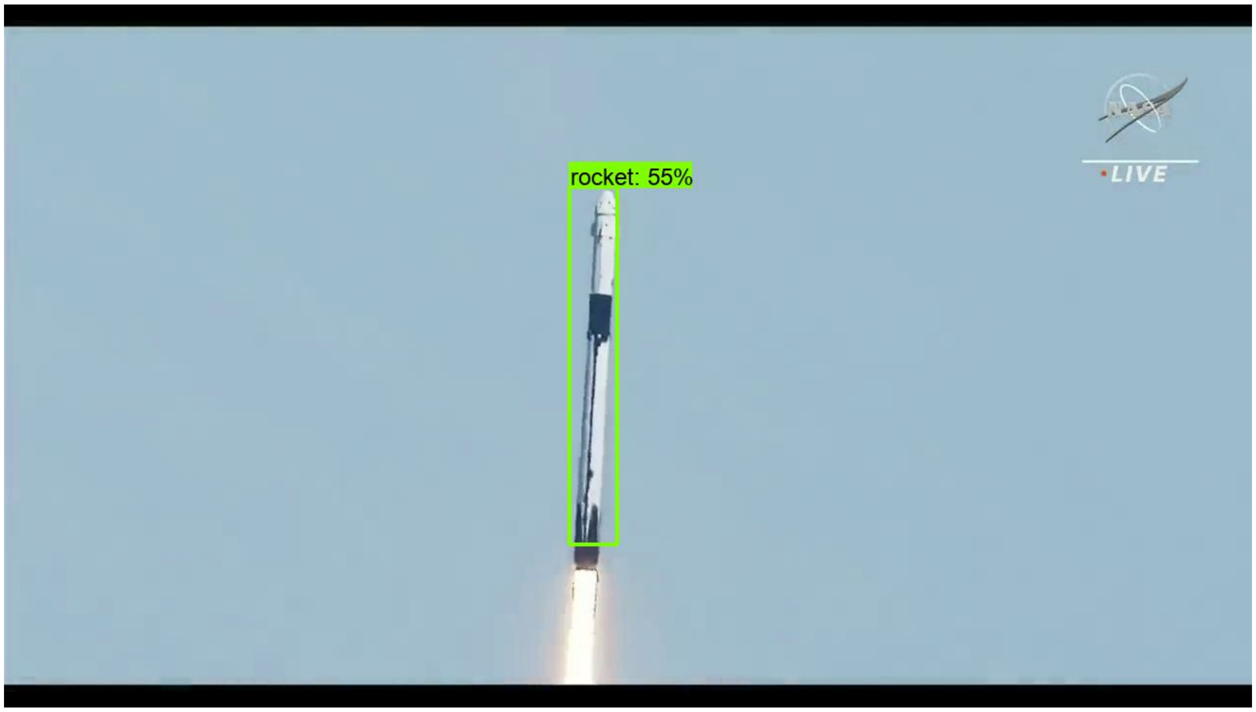

Figure 5: Sample detection, light background.

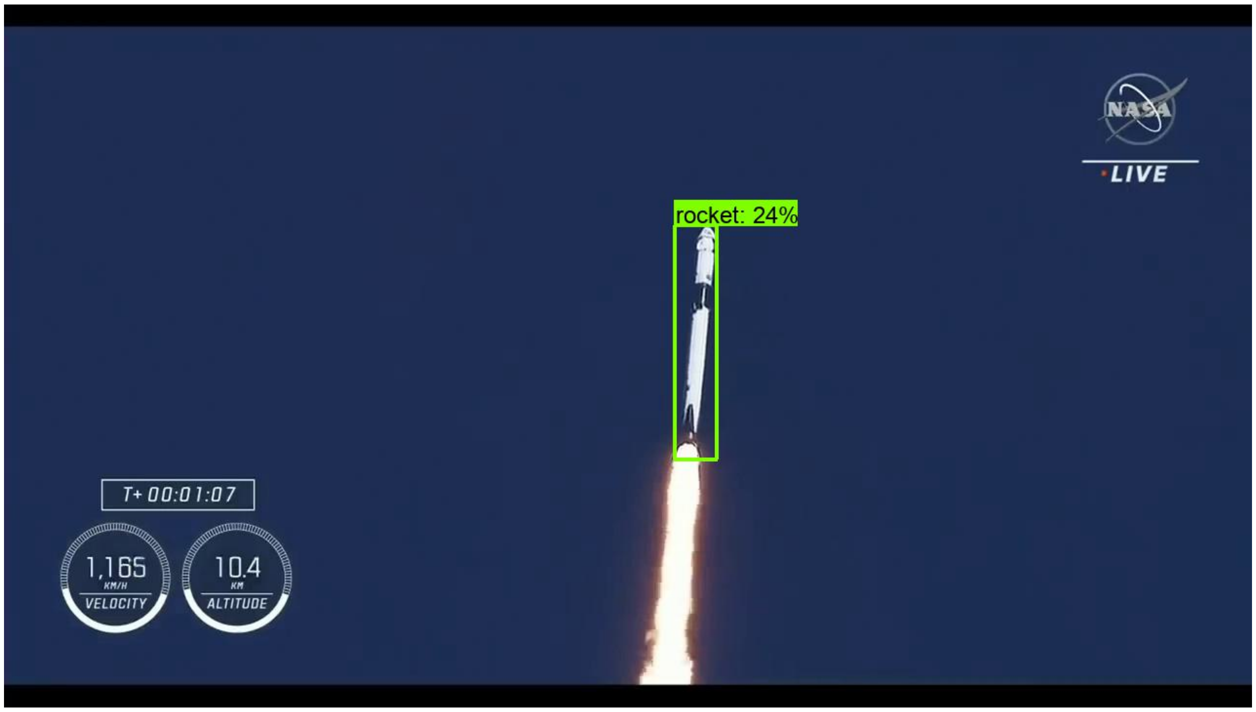

Figure 6: Sample detection, dark background.

4 Analysis and discussions

4.1 Could not save model after fine-tuning

We changed the architecture of the pretrained RetinaNet by changing the number of output classes of classification head from 90 to 1). The Tensorflow docs for saving fine-tuned models were out of date, and we struggled with saving the trained model, as did others.

4.2 Slow computation time

Each inference the model makes is computationally expensive - the model can process about 7 or 8 frames per second. Other object detectors with a quicker inference time have the tradeoff of lower mAP (e.g., MobileNet Howard et al. [2017] could not converge with the amount of labeled data available.

4.3 Low confidence

The low confidence of the fine-tuned model could be due to the small amount of training data. Improving this score will be the subject of future work.

4.4 Not all frames detected

Occlusions caused the model to fail (see Figure 7). However, some frames that were not occluded also have no bounding box predictions above the minimum threshold specified (.2).

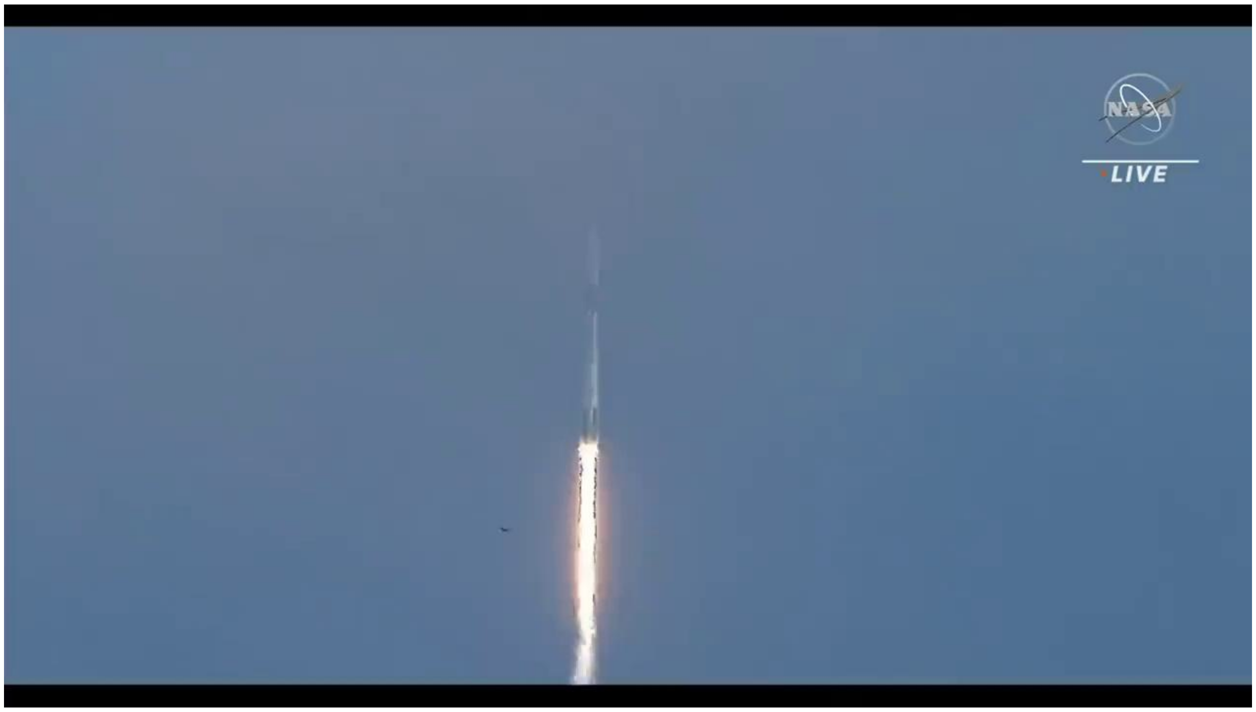

Figure 7: Sample occluded image, model doesn't properly detect.

5 Conclusion

This project introduced a rudimentary object detector fine-tuned on a small set of labeled data that detects SpaceX Falcon 9 rockets in images and video. Future work could explore 1) different model choices, 2) more robust performance analysis (i.e., compute IoU and mAP for validation data), 3) training on and identifying other rocket types, 4) detecting occluded rockets, and 5) detecting rockets at night.

References

Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, et al. Speed/accuracy trade-offs for modern convolutional object detectors. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7310–7311, 2017.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

Maxim Tkachenko, Mikhail Malyuk, Andrey Holmanyuk, and Nikolai Liubimov. Label Studio: Data labeling software, 2020-2022. URL `https://github.com/heartexlabs/label-studio`. Open source software available from https://github.com/heartexlabs/label-studio.

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

Suramya Tomar. Converting video formats with ffmpeg. *Linux Journal*, 2006(146):10, 2006.

Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.