SSA data analysis with a two-pronged approach including machine learning for RSO detection

Sam O. M. Wright
Spaceflux, Ltd[*]*& University College London*[†]
Marco Rocchetto, Ingo P. Waldmann
Spaceflux Ltd

ABSTRACT

Obtaining accurate optical measurements of resident space objects (RSOs) is an important pillar of an effective space situational awareness (SSA) system. The raw data for such measurements are the output of an optical sensor network, while the derived measurement data are the product of further data analysis. In this work we present a data analysis pipeline for this task which covers raw data reduction, and uses modern, deep learning powered object detection to locate RSO point sources with the YOLOv5 neural network. We demonstrate that this method is capable of identifying RSO points with high precision even when trained with a relatively small data set. We show that the machine learning model can yield a $\sim$45x speed improvement versus the 'classic' approach pipeline segments, given an achievable setup at the edge. This equates to a $\sim$30% speed improvement for a complete run of the pipeline. We show that the machine learning model can yield detection speeds that allow real-time data analysis of data streams at the edge and represents a valuable technique deployable on a variety of edge-based architectures.

1. INTRODUCTION

To effectively conduct on demand SSA with optical observations it is necessary to not only have access to high quality, well targeted raw data but also to have an effective, robust and fast data analysis pipeline. Such a pipeline should incorporate data cleaning processes and the data reduction steps common to optical telescope data analysis workflows.

A crucial function of such a data analysis pipeline is to identify point sources and streaks within a frame; these correspond to observed RSOs and stars in the case that the observations are taken in tracking rate mode (TRM) and vice versa for sidereal stare mode (SSM). The information associated with both the RSOs and stars are important for generating the tracking data for the subject of the observation campaign; the former as it concerns the object itself and the latter since star positions are required to run the astrometry necessary for high precision tracking data. A number of techniques exist to isolate streaks from optical frames for instance matched filters, including steerable variants [2] or informed by a Radon transform analysis [7] or across a set of possible streak filters [9]. Such a set could be further narrowed down by taking into account prior constraints on possible filters originating from the observation equipment and circumstances. The classic contour method of edge detection is also possible to detect streaks. Over the last decade machine learning approaches have played a growing role as deep learning has grown in prevalence, fuelled by access to far greater compute power.

The machine learning approach benefits from the ubiquity of open source machine learning libraries and frameworks for object detection. These frameworks have proliferated from the large amount of research done in this field for applications in computer vision including the development of self-driving vehicles. As in many fields, researchers have recently started to apply machine learning to tasks within SSA [6] since this class of algorithms performs well at computer vision tasks it is very natural to explore this approach. One such series of these libraries is 'You Only Look Once' or YOLO and it is its fifth iteration (YOLOv5) which we use in this work [5]. It is based on the convolutional neural network architecture which has proved so powerful in computer vision tasks, learning convolutional filters to extract relevant features from images.

[*]Spaceflux Ltd, 71-75 Shelton Street, WC2H 9JQ, London, United Kingdom
[†]Department of Physics and Astronomy, University College London, Gower Street,WC1E 6BT London, United Kingdom

In parallel, the pipeline also supports the option of using a 'classic' approach. This centers around fitting Gaussian distributions to the sources in the frame, from which streak / spot characterisations are obtained. The motivation for this approach comes from the astronomy technique of fitting the point spread function (PSF) when performing photometry; the point spread function describes the response of the optical sensor to incident light from the observed objects [4]. The extension to a 2D Gaussian naturally extends well to quantifying the shape of the source in the frame.

2. METHODS

2.1 Pipeline Overview

Two primary components comprise the data analysis pipeline; the first provides data reduction functionality, while the second performs astrometry, identification of light sources and source characterisation. The reduction component ingests sensor-specific calibration frames and generate master calibration frames from these. It can then reduce a given data frame by: subtracting the detector bias, subtracting the dark current and dividing out the flat field.

After these basic data reduction steps have been performed, the 'reduced' data is fed to the analysis part of the pipeline, which performs the following steps:

1. Calculation of basic image and signal-to-noise statistics to inform the analysis.
2. Locating the centroids of sources in the image.
3. Characterisation of sources as points or streaks.
4. Astrometry to obtain positions.

The resulting position information for the campaign is reformatted into relevant tracking data message formats.

2.2 Gaussian Fitting Approach

The point and streak detection phase begins by taking the bright points as the centroids of sources. In the Gaussian Fitting approach these sources are then categorised as either point sources or streaks; the DBSCAN clustering algorithm [3] is used to group the points and the shape of the associated group's bounding box to classify the source as a point or streak. In the next step the sources are fit with 2D Gaussian distributions; this step provides a fit for the brightness distribution but also provides refinement of the earlier point-streak categorisation through the shape of the fit 2D distribution. When fitting Gaussians for the streaks the pipeline iteratively uses prior streak shape to inform next fit of streaks. Information about the streak angle and length is transferable between streak fits because these two parameters are influenced by the observational setup itself.

2.3 Machine Learning Approach

The machine learning approach provides an alternate approach to locate point sources as in steps 2 & 3 of the pipeline data analysis process outlined in section 2.1. For the machine learning approach we take the Shenanigans v0.3.0 data set created by the SatSim simulator [1] and distributed on the Unified Data Library[1] (UDL). From this data set we took a single example from each simulated observation campaign of RSOs in GEO with nominal status: a total of 211 frames on which we then made an approximately 80/20 train, validate split to yield 168 training samples and 43 validation samples. We selected the yolov5s (small) architecture to keep the number of trainable parameters low and then trained this network from scratch using an Nvidia A100 GPU. This training from scratch contrasts with the approach of using pretrained weights for a network already trained on the Common Objects in Context (COCO) 2017 object detection data set [8]. We took this approach to training as the point sources we aimed to identify are less visually complicated than the objects in the COCO data set and do not exist within the same domain of 'common objects' for which the COCO data set is intended to be used.

[1] `https://accounts.unifieddatalibrary.com`

3. RESULTS

3.1 Machine Learning Approach

3.1.1 Training Progression

We monitor the training progression of the network over the 300 epochs (or iterations over the training data) it makes; this is to ensure the model learns as it progresses and that it does not overfit to the training set and thus lose its ability to generalise. The results of this monitoring are shown in Fig. 1. By observing that the loss functions continue to decrease when evaluated over both the training and validation sets. The characteristic signature of over fitting is not seen here: there is no continued decrease in training loss with a plateau in validation loss. In fact the continuing decrease in losses for both sets could indicate that we have under fit, opening up the potential for improved performance in the future through simply training for more epochs.

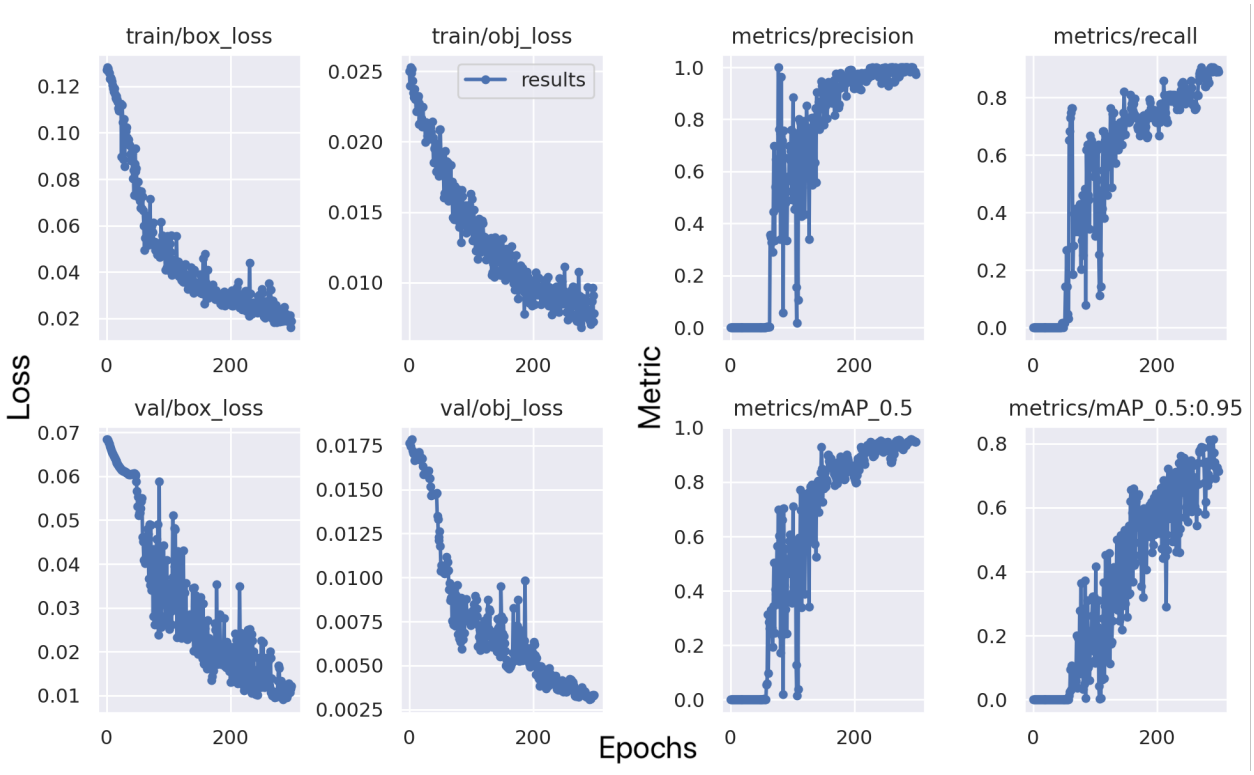

Fig. 1: Training progression for the neural network showing two of YOLOv5's loss metrics in the left half (evaluated for the training set in the top row and the validation set in the bottom row) and performance metrics in the right half (evaluated over the validation set).

3.1.2 Accuracy Metrics

The best trained model performed well across a suite of standard object detection metrics when evaluated on the validation set, as shown in table 1. Precision indicates the ratio of correctly identified examples to all of the examples which have been predicted as point sources. Recall indicates the ratio of the correctly identified examples to all of the examples which are actually point sources. The two mAP metrics are the precision values averaged over two thresholds for a measure of overlap of the predicted and ground truth point source bounding boxes: intersection over union (IoU). This is simply the ratio of the area of bounding box overlap to the area of the union between the bounding boxes. In the case of mAP 0.5 this is simply half, while for mAP 0.5:0.95 the metric is evaluated averaging over IoUs thresholded between 0.5 and 0.95 at 0.05 increments. Intuitively it can be seen that the latter is lower since it averages in a tighter constraint on how much the predicted bounding box should overlap.

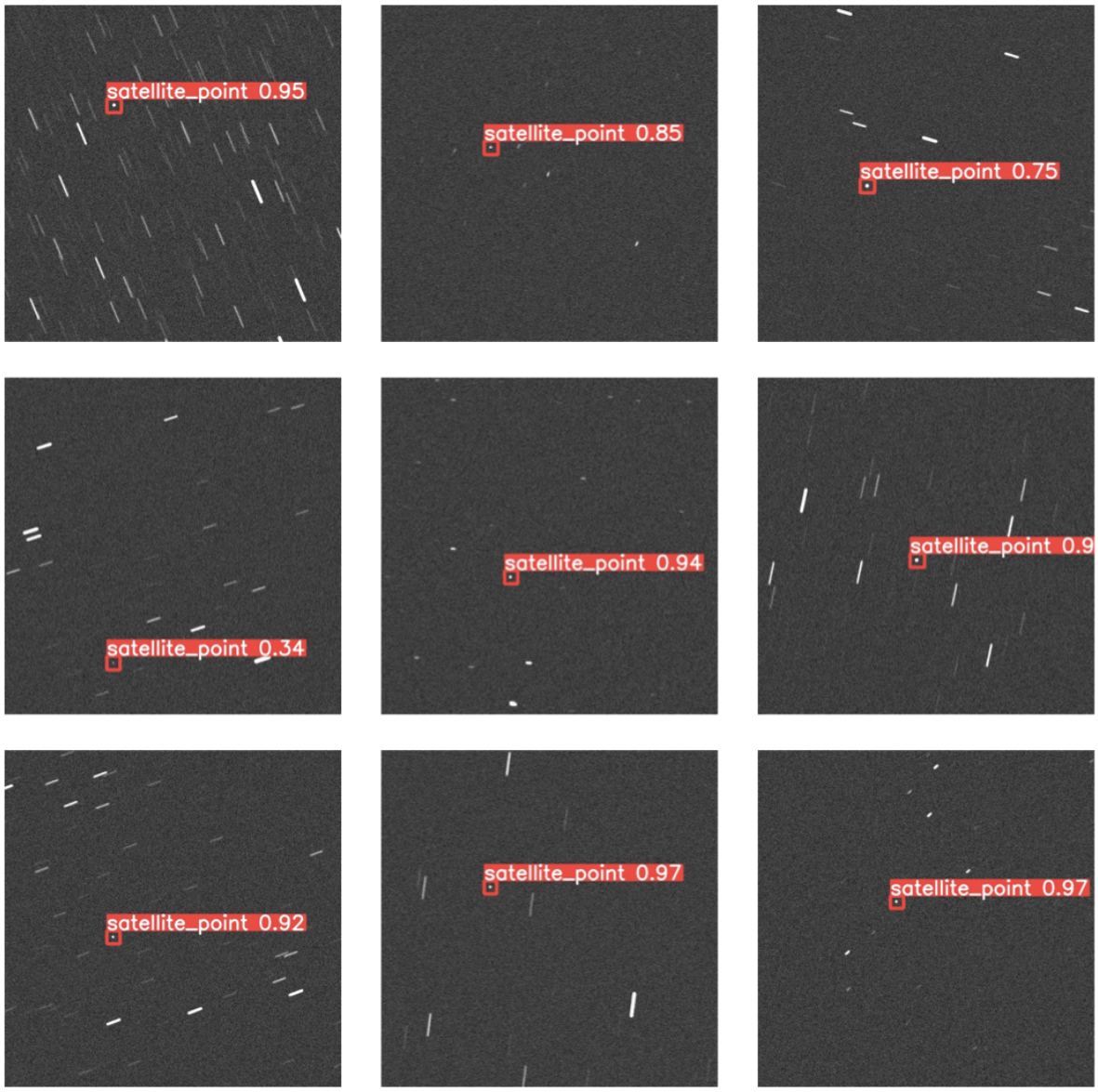

Fig. 2: Examples of RSO detection in various fields. The reported numbers in the labels are the detection confidences expressed as fraction.

Metric	Score
Precision	1.0
Recall	0.90373
mAP 0.5	0.95726
mAP 0.5:0.95	0.81372

Table 1: Object classification metrics evaluated on the validation set: precision, recall, mAP 0.5 and mAP 0.5:0.95 shown for the best performing model.

Single values do not express all the information necessary to evaluate the model however. To obtain a more nuanced understanding of the model's performance it is necessary to consider how the metrics vary as a function of the model's confidence in its predictions: precision and recall as a function of confidence can be seen in Fig. 3 & 4 respectively. In the case of the precision-confidence curve, the model has high precision even for examples where it is less confident, with precision rising to 1 for confidence values of 29.4% and above. For the recall-confidence curve the model exhibits reasonably good performance, although it shows a tendency for the model to have a lower recall for higher confidence; recall can also be called the true positive rate so it is under-identifying true point source examples with a higher confidence.

Fig. 5 shows the variation in precision for recall, an area under the curve as close to unity as possible is desired since this represents a model with both perfect recall and perfect precision. The high area under the curve for this model shows that it is able to identify accurately but not at the expense of omitting predictions. The F1 score also captures information about the recall and precision at the same time since it is the harmonic mean of both. In Fig. 6 we can see this plotted as a function of confidence. The model performs well with an F1 score over 0.8 for the majority of confidence values. In Fig. 2 we show a few examples of the detected satellites and associated detection confidences.

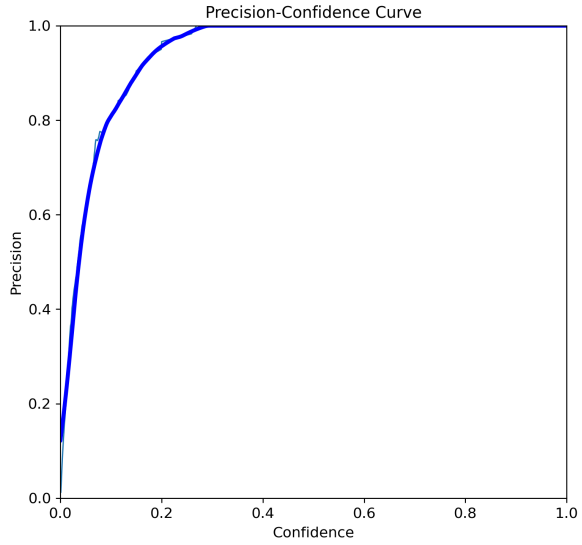

Fig. 3: The precision-confidence curve showing precision as a function of confidence for the model's predictions.

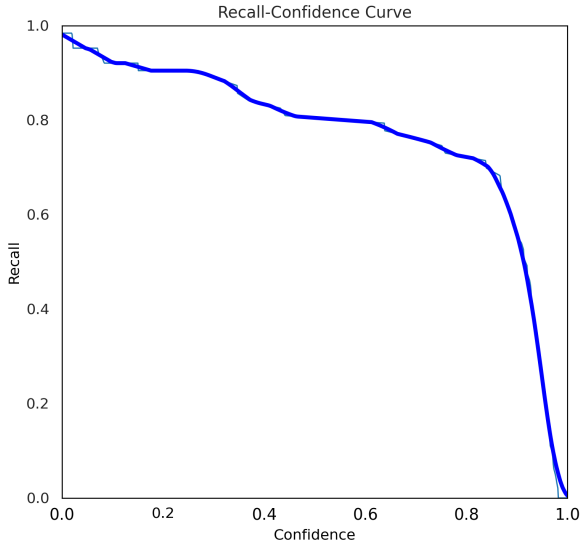

Fig. 4: The recall-confidence curve showing recall as a function of confidence for the model's predictions.

3.1.3 YOLO Inference Times

In the interest of deployment to the edge and with the goal of achieving real time data processing, we evaluated inference times for the machine learning model across a number of software and hardware deployment strategies. These deployment strategies included running the inference on hardware with varying levels of floating point tensor

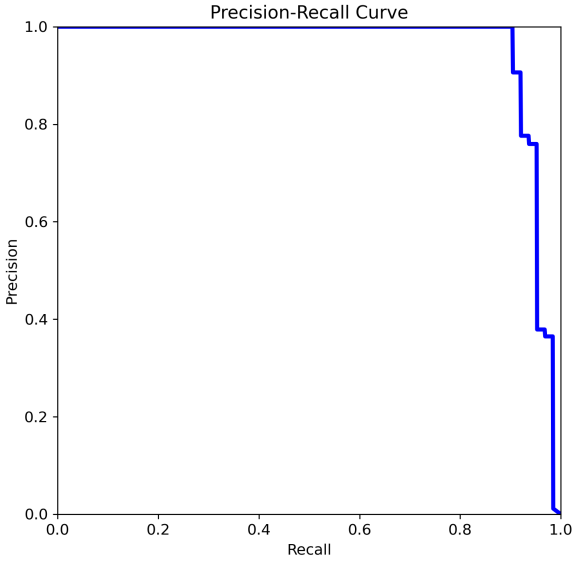

Fig. 5: The precision-recall curve showing how precision varies with recall.

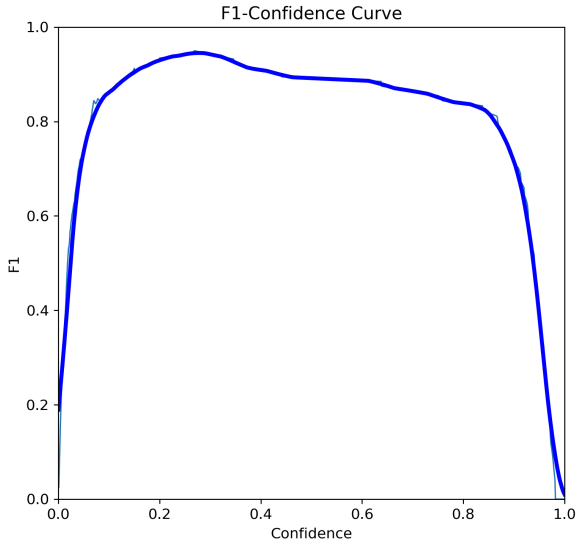

Fig. 6: Interpolation for Data 2

optimisation (CPU, GPU and Neural Engine) and software compilation to an inference optimised format (ONNX). The results for these tests are shown in table 2.

Hardware	Model Inference Format	Mean Inference Time
Intel Xeon Silver 4314 CPU	PyTorch Default	385 ± 3.12 ms
Apple M1 Pro CPU	PyTorch Default	340 ± 30.1 ms
Apple M1 Pro CPU	ONNX	80.3 ± 3.89 ms
Apple M1 Pro Neural Engine	CoreML	29.4 ± 1.59 ms
Nvidia A100 GPU	PyTorch Default	18.5 ± 3.83 ms

Table 2: Mean inference time for each model format and execution hardware obtained by averaging over 10 loops for the same example frame.

3.2 Overall Processing Times

Table 3 shows the processing times for a single example frame broken down by pipeline step. Four of the steps are common to both the 'classic' Gaussian fitting approach and the machine learning approach: 'Load & reduce', 'Statistics', 'Querying' and 'Astrometry'. The steps unique to the 'classic' approach are: 'Find centroids', 'Characterise sources' and 'Refine sources'; with the machine learning model, all three are replaced by a single inference run.

4. CONCLUSION AND FUTURE RESEARCH

In this work we demonstrate an SSA data analysis pipeline using a deep learning approach for RSO identification from optical sensor data. The machine learning approach leverages state-of-the-art technology from an ecosystem which is the product of a great amount of openly available research. We can conclude that the machine learning object detection approach can provide a capable and fast alternative to more traditional techniques which can support an SSA data analysis pipeline, even when trained on a small data set. In fact there are several open avenues of future research which could improve performance further.

Since we see no evidence of overfitting for training over the 300 epochs used here, it is likely that we could continue the training process for more epochs to produce better performance. Such an extension can make use of the training algorithm's early stopping function to terminate the training process when no further improvement is to be had.

Pipeline Steps	Classical (ms)	YOLO (ms)
Load & reduce	559	559
Statistics	47	47
Find centroids	438	
Characterise sources	1158	
Refine sources	2025	
YOLO detection		$\sim$20
Querying	838	838
Astrometry	1285	1285
Total	6350	2749

Table 3: Single frame processing times for each stage of the pipeline in 'classic' and 'YOLO' configuration. By bypassing time intensive detection steps, we can significantly improve on data processing times.

Additional performance improvements are likely also possible through some hyperparameter optimisation; training multiple models while varying values such as the training algorithm's learning rate hyperparameters or enabling the framework's data augmentation options. In addition we can explore YOLOv5 architectures larger than the 'small' variant used. The effect of this will be to add more parameters to the model. Although this will increase training time, it will be tractable within a few days on an Nvidia A100 GPU and will result in a minimal increase to the inference times shown in Fig. 2.

Given a total replaceable processing time of 3,621 ms with the classic technique, the machine learning approach is between $\sim$9 and $\sim$240 times faster in detecting the RSO, depending on the hardware used and model deployment configuration. A reasonable analogue to an edge setup is the ONNX compiled model running on an M1 Pro CPU; this configuration incurs minimal overhead in exporting to the ONNX optimised framework and is consistent with modern CPU hardware at the edge but does not assume GPU availability. In this setup we could expect a $\sim$43 and $\sim$47 times speed increase. On average, this would equate to $\sim$2800 ms for the whole pipeline.

The strong performance of the network when trained on a small data set shows that a machine learning solution for this task is not beholden to the availability of large data sets. In fact, a data set of this size means that the manual labelling of custom training sets is possible since it would not be too labour intensive. This notably includes sets which are not simulated and so lack a procedurally generated ground truth.

5. REFERENCES

[1] Alexander Cabello and Justin Fletcher. SatSim: a synthetic data generation engine for electro-optical imagery of resident space objects. In *Sensors and Systems for Space Applications XV*, volume 12121 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, page 1212107, June 2022.

[2] Vojtech Cvrcek and Radim Sara. On Fast Matched Filter for Streak Detection and Ranking. volume 8 of *European Conference on Space Debris Series*, 2021.

[3] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, page 226–231. AAAI Press, 1996.

[4] J. N. Heasley. Point-Spread Function Fitting Photometry. In Eric R. Craine, David L. Crawford, and Roy A. Tucker, editors, *Precision CCD Photometry*, volume 189 of *Astronomical Society of the Pacific Conference Series*, page 56, January 1999.

[5] Glenn Jocher. YOLOv5 by Ultralytics, 5 2020.

[6] Jarred Jordan, Daniel Posada, David Zuehlke, Angelica Radulovic, Aryslan Malik, and Troy Henderson. Satellite Detection in Unresolved Space Imagery for Space Domain Awareness Using Neural Networks. *arXiv e-prints*, page arXiv:2207.11412, July 2022.

[7] Martin P. Levesque and Sylvie Buteau. Image processing technique for automatic detection of satellite streaks. 2007.

[8] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2014.

[9] A. Vananti, K. Schild, and T. Schildknecht. Improved detection of faint streaks based on a streak-like spatial filter. *Advances in Space Research*, 65(1):364–378, 2020.