Convolutional neural network approaches for deep-space object detection in wide field of view camera arrays

Austin Ibele *(Kung Fu AI),* **Spencer Romo** *(Kung Fu AI),*
Brian Williams *(Slingshot Aerospace),* **Steve Kramer** *(Kung Fu AI),* **Tate Noster** *(Kung Fu AI)*

Abstract
Wide field of view (WFOV) optical sensors can provide a tremendous amount of satellite observation data. For example, the PANDORA (Persistent AND Optically Redundant Array) system is a 5×9 array of commercial-off-the-shelf cameras located at the Air Force Maui Optical and Supercomputing site. It captures 20°×120° WFOV images of the night sky at a rate of two frames per minute. Identifying satellites in these images is time consuming and error prone when performed manually. Convolutional neural networks (CNNs) are a natural deep learning approach for this problem, but standard object detection CNNs are not well suited for detecting objects that are so small and faint. We make adaptations to RetinaNet object detection neural networks and make use of extensive image augmentations to train models for detecting these small objects. Our CNNs achieve max F1 scores against simulation data of 0.90 and we fine-tune our networks to achieve a max F1 of 0.73 against real PANDORA imagery.

Introduction

Optical telescopes are important for space situational awareness, particularly for tracking satellites and other resident space objects (RSOs) in higher orbit regimes, such as geostationary (GEO) orbits. As RSOs proliferate in near Earth space, demand for robust tracking capabilities increases at least proportionally.

Wide field of view (WFOV) optical sensors can potentially help meet these needs in a more scalable manner than narrow field of view (NFOV) telescopes. While NFOV telescopes must be tasked for specific data collections, WFOV telescopes can survey a sizable fraction of the sky. As an example, the PANDORA (Persistent AND Optically Redundant Array) system, a 5×9 array of commercial-off-the-shelf cameras located at the Air Force Maui Optical and Supercomputing site provides persistent nightly data collections across a 20x120 field of view oriented along the GEO belt. PANDORA produces hundreds of gigabytes of data each night, but the imagery is low resolution and light from stars dominates the fainter signals from RSOs.

Computer vision has proven to be very useful for many object detection tasks, where they are trained to put bounding boxes around objects of interest. Convolution neural networks (CNNs) have been applied successfully to object detection tasks such as cancer detection in medical images [1], facial recognition [2], and general object detection and segmentation for autonomous driving [3], however, they have not been a common approach for locating RSOs in WFOV images. There are two significant barriers to using CNNs for RSO detection in WFOV. First, traditional object detection neural networks do not generally perform well detecting extremely small objects, for example, when the object of interest is an RSO that is only a few pixels wide; and second, the Graphics Processing Unit (GPU) memory requirements for a CNN that can process native-size WFOV images are prohibitively large.

We show how modifications to a common CNN-based objector detector, together with image processing techniques, can be used to overcome these barriers, resulting in state-of-the-art detection performance with low computing-demand and fast inference capability. Our results show that our custom object detector exceeds performance of wide area motion imagery (WAMI) techniques by a large margin on GEO satellite detection in highly realistic images generated by SatSim [4], a high fidelity space scene generator. On GEO detection in real PANDORA images, the object detector achieves a maximum F1 detection score of 0.73. As a result of approximately 400 experiments, we were able to refine the neural network architecture for small size and fast inference without sacrificing performance. This, together with our method of performing detection on small crops of the full size image before later reconstitution, allows inference on a full 4224x5776 pixel stacked PANDORA images, comprising a 14x19 FOV, to take place in just 25s using a single Tesla V100 GPU.

Background and Related Work

Computer vision applications to detecting small objects in telescopic imagery date back to at least the 1990s, where [5] applies a series of image processing steps including background estimation and feed-forward neural networks for the detection of stars and galaxies in large (60Kx60K) images. More recently, [6] used wide area motion imagery (WAMI) techniques such as histograms of oriented gradients (HOGOR) to detect GEO satellites in simulated WFOV images.

CNNs have been widely employed in computer vision literature for object classification [7], object detection [8] and instance segmentation [9, 10] among other problems. Object detection problems turn a standard size output from a CNN into a vector of bounding boxes around detected objects. These are typically denoted by a 6xN-dimensional vector, where the first position denotes the certainty of detection, the second denotes the class of the detection (i in the set of k), and the 3-6th positions denote the X_{min}, Y_{min}, X_{max}, Y_{max} coordinates of the bounding box, respectively.

Object detection architectures typically produce a vector of this shape for each 'keypoint', which could number in the millions of possible detections per image. These keypoints are then reduced using techniques such as non-maximum suppression [21] and thresholding to return a vector of predictions.

Object detection benchmark datasets, such as COCO [11] and ImageNet [12**]** have been widely used [13]. These data sets focus on human vantage points, with common objects in the scene. The objects in these images tend to be large in frame, and few in number. This leads to models whose performances are tuned to a distribution that is dissimilar to our problem space.

[14] and [15] use YOLO-based object detection networks for RSO detection in NFOV and WFOV imagery respectively. Both papers mitigate the problem of small objects, in part, by utilizing spatial-temporal networks that exploit data from multiple frames to improve accuracy.

We build our object detection network off of the RetinaNet architecture [16]. The RetinaNet model makes use of a feature pyramid network to detect objects at multiple scales, and introduces a novel loss function, Focal Loss, that handles sparse detections in a tractable and mathematically stable way. By deemphasizing easy-to-classify examples, Focal Loss forces the network to focus on learning harder-to-classify instances.

The RetinaNet architecture follows a common pattern in modern deep learning, which is to use a base network such as Resnet[17], VGG[18] and ResNext[19] and then a "head" to perform task-specific objectives [20].

These base architectures are convolutional in nature, containing successive depth-wise blocks that perform feature extraction. As the signal travels through successive layers, spatial resolution (height and width dimensions) are successively reduced, while signal in the 3rd dimension (RGB in color images) is enhanced with the use of convolutional filter blocks. These blocks are in effect, filters similar to Fourier transformations, with tunable weights to enable them to extract various patterns. In succession they can be trained through backpropagation to take on extraction capabilities to enable them to feed into a downstream task.

Convolutional base networks are almost exclusively benchmarked and pre-trained on image classification tasks. Then through the use of transfer learning, the networks can be applied to other tasks with substantially shortened training time [21].

The head component of the architecture is appended to the base network to enable a specific task to be accomplished. In the case of classification, the required layers are simple, and terminate in a 1xN where N is the number of classes that can be predicted. This network produces a vector of predictions, to which a SoftMax activation is applied, resulting in a max class ID, and a prediction. In the case of object detection, the head is much more complex. The reader is referred to [16] for more details.

Our Method

We train CNNs that are purpose-built for detecting the small objects present in WFOV optical imagery. In this paper, we exclusively consider the problem of detection in single frame imagery, which [22] identifies as critical for successful multi-frame detection algorithms. The multi-frame problem is considered in [15] and we discuss additional multi-frame approaches at the end of this paper.

We leverage the ResNet architecture [17] for our base model. ResNet offers a few different model sizes including ResNet18, ResNet34, ResNet50 and ResNet101. The number component of the names of these architectures refers to the depth of the convolutional filters applied. More filters leads to more parameters, but also more capability of the model. For complex image classification or ther tasks with human-level tasks, deeper networks perform better than shallow networks, and the effect of increasing depth and numbers of parameters has been widely studied. Our experiments showed that, contrary to object detection of everyday objects in earth-based scenes, larger ResNet backbones (ResNet-50 and ResNet-101) provided no performance benefit for this problem relative to the smaller ResNet-18 backbone. We thus selected ResNet-18 as the backbone for our model, so that we could benefit from the faster compute times that it enables.

The head for our networks was built off of RetinaNet. RetinaNet employs a 'Feature Pyramid' mechanism that enables detections to be produced from various depths of the base network. As the network traverses its depth, the resolution of the height and width decreases (i.e., 512x512 → 256x256 → 128x128, etc…). This decrease in resolution contributes to fewer detection points, at each successive level. The feature pyramid allows for detections to be generated without those detections needing to leverage the entire depth of the network. This can be advantageous for simple objects, and it can be advantageous for objects of widely varying size in scene. Given the small size of the objects to be detected in our datasets, we found that tuning this feature pyramid was of particular importance. To this end, we implemented tooling which enabled us to understand which level of the feature pyramid was most instrumental in making detections in order to tune the network upstream and downstream, and increase the efficiency of training and inference.

CUSTOMIZATIONS OF RETINANET

In order to characterize the detection performance of the various layers in the candidate RetinaNet architectures, detection scores were tracked before and after non-maximum suppression (NMS). If predictions on the same regions of interest are made on multiple different layers of the feature pyramid network, RetinaNet discards lower-scoring candidates to avoid duplicate detections. We wrote a custom function to return two metrics that describe the number of detections per feature map in RetinaNet. The two metrics we used were:

- ***balance*** = (# detections in layer with most detections - # detections in layer with least detections) / (total detections)
    - When balance is 1, all detections are occurring in a single layer. When balance is 0, detections are spread equally amongst layers
- ***delta*** = (# detections in first layer - # detections in last layer) / (total detections)
    - When delta is positive, detections are taking place in more shallow feature maps. When delta is negative, more detections are taking place in later feature maps.
    - When balance and delta are both 1, all detections are taking place on the first feature map.

Fig. 1 below shows the balance metric by training epoch over time. The blue line (hidden behind orange and green lines) corresponds to the standard RetinaNet architecture. In this case, balance was always 1 and all detections were always taking place on a single feature map. The orange and green lines correspond to a special 'columnar' feature pyramid design that we implemented, which maintains resolution through successive feature maps (at the penalty of greatly increased model size and processing time). Even with this special design though, all detections took place on the same feature map after 50,000 training steps.

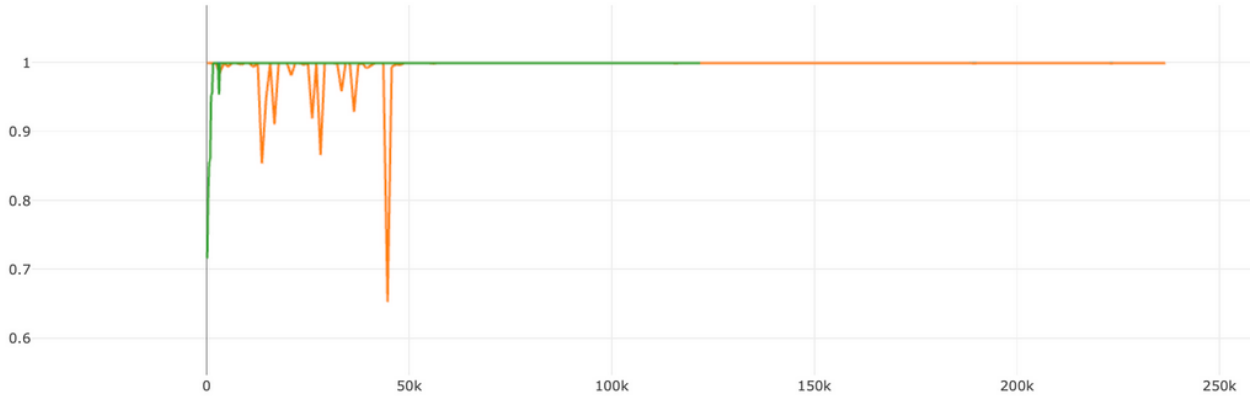

Fig. 1. Balance Metric

Based on these results and the ones for the delta metric, we established that all detections are taking place only within the first feature map. We thus eliminated all but the first (highest-resolution) feature map, given that no detections were made on lower-resolution feature maps. This modification allowed us to achieve performance equal to the base RetinaNet configuration but with faster, more reliable training, faster inference, and smaller model size.

Finally, we investigated the effects on training time and model performance of various anchor box configurations. We found that anchor boxes should be the smallest size possible that maintains overlap with ground truth boxes. If boxes are too large or too small, training takes too long or might not complete successfully. In the default RetinaNet detector, anchor boxes of 3 different aspect ratios are used. We found that using a single anchor box of aspect ratio 1.0 had identical performance to 3 different anchor boxes of varying aspect ratios. Thus, we chose to use a single bounding box in our customized RetinaNet-based detector.

Table 1 below summarizes the key modifications of relevant parameters and settings for our custom CNN compared with the standard version from the original implementation by [16]. Settings marked with an asterisk (*) impact the overall number of model parameters.

Table 1. Default RetinaNet vs. Our Implementation

	Original Version of Lin, *et al.*	**Our Configuration**	**Reason**	**Effects**
Stride*	2, 2, 2, 2, 2	1, 2, NA, NA, NA	High-resolution feature map	Increased detection performance
Feature Pyramid Network (FPN)*	5 feature maps + 1	1 feature map + 1	All detections take place on first feature map	Faster training, faster inference – no effect on detection performance
Bounding Boxes*	sizes = (8 - 512)*5, ARs = (0.5, 1, 1.5)	sizes = (20)*1, ARs = (1)	Faster, more reliable convergence	No effect on detection performance (if the model trains)
Focal Loss	gamma = 2, alpha = .25	gamma = 1.5 - 2.5, alpha = 0.15 - 0.27	Important parameter when dataset has large class imbalance, more reliable/faster convergence	Possibly faster training, possibly slightly better detection performance

Scale up	512x512px → 800x800px	512x512px → 1024x1024px	Scale up is essentially up sampling. This helps the model learn finer resolution features.	No difference was seen because the ratio of scale-up was not changed much. However, if the scale up were 128x128px → 1024x1024px, for example, we would expect a noticeable increase in performance. This was not performed however as it would cause a 16x increase in inference time.

TRAINING DATA

We trained our networks against both simulated and real-world data. For our simulated data, we leveraged SatSim [4], a GPU-accelerated space scene simulator. Reference [6] investigated the ideal SatSim parameters for simulating GEO images based upon PANDORA camera specifications, as well as known image processing parameters and the results of a hyperparameter optimization study they performed. We generated our own Monte Carlo datasets, which contained GEO satellites randomly placed on a starry background, based upon these parameters. However, we used smaller, 512x512 pixel images instead of 4224x5776px images and we adjusted the Horizontal FOV and Vertical FOV accordingly so that the pixels/degree FOV was identical between our dataset and that of [6]. Our Monte Carlo dataset was then split into a training dataset containing approximately 40,000 images as well as a test dataset containing 50 images and 1108 GEO satellites. An example image from our Monte Carlo dataset is shown in Fig 3.

Our real-world dataset consisted of images from the PANDORA array. We considered both "unstacked" single camera images and stacked images, which are constructed by overlaying images obtained from neighboring cameras for noise-reduction. Our unstacked dataset consists of 29,302 image clips of 515x512 pixels each. Our stacked dataset consists of 5,790 images at a resolution of 4224x5776 pixels. Stacked and unstacked images were organized by the camera position from which they came. This organization schema was important when deciding which data to withhold for testing versus training. PANDORA has 9 rows of 5 cameras each. The stacked datasets we used came from 6 of the 9 camera rows, which we label as "STACK0" through "STACK5". Each stacked image consists of 5 overlaid images; one from each camera in the camera row in a PANDORA array. Cameras in the same row photograph the same area of the sky.

The real-world dataset included human annotations of not only GEO satellites, but of satellites in lower orbits (LEO and/or MEO) as well, which we did not train our algorithms to detect. These LEO and MEO objects appeared as long, narrow streaks. We filtered these from the dataset based upon their annotation box size.

Additionally, unlabeled GEO satellites were observed in the annotated datasets. We did nothing with objects we believed to be unlabeled GEO satellites. Detection predictions on these objects are counted as false positives in our results.

TRAINING PROCEDURE

Model training was performed on the 40,000-image Monte Carlo Training Dataset using a Nvidia DGX box (4x V100 GPUs, 256GB RAM, 48 CPU Cores). Random image augmentations were added during training, making the number of unique training images seen by the model much higher. Detection models for detection on real PANDORA images were first pre-trained using SatSim-simulated data before fine tuning on a subset of the real images.

EVALUATION PROCEDURE

Object detection algorithms can be evaluated in a number of ways. We leverage the common 'intersection-over-union' metric. Intersection over union (IOU) is a measure of the total area in the intersection of two detections over the union of the total area of two objects. For objects that perfectly overlap, this value is a 1.0. For objects that are non-intersecting, the value is 0. In our evaluation methodology, we look for detections above a threshold value for detection, and then remove that prediction and ground truth object from consideration in other matches.

The objects in the ground truth annotations and the object detection results are then associated to produce three categories:

- True Positives, where a a ground truth annotation intersects sufficiently with a prediction
- False Positives where there is a prediction without a corresponding ground truth annotation
- False Negatives, where there is a ground truth annotation that doesn't have a corresponding prediction

Table 2 provides a visual guide to these metrics.

Table 2. True Positives, False Positives and False Negatives

		Predicted Value	
		Positive	**Negative**
Actual Value	**Positive**	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)

For our evaluation, we used an IOU threshold of 0.5 Thus prediction boxes that overlapped a ground truth box with an IOU > 0.5 were considered true positives. Prediction boxes that did not overlap any ground truth boxes with an IOU > 0.5 were considered false positives. Finally, any ground truth box that was not overlapped by any prediction box with an IOU > 0.5 was considered a false negative.

Using TP to denote the number of true positives, FP to denote the number of false positives and FN to denote the number of false negatives, we construct precision and recall values. Recall, defined as

$$Recall = TP / (TP + FN)$$

denotes the fraction of all targets that were correctly detected, and precision defined as

$$Precision = TP / (TP + FP)$$

denotes the fraction of correctly detected targets among all detected targets.

These two metrics are often combined into an F1 score, which is the harmonic mean of precision and recall:

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN}$$

F1 is a balanced metric, equally valuing precision and recall, and better approximating the true ability of the model to achieve the target objective. However, a model can be tuned to prefer precision by increasing the threshold for detections, which results in fewer predictions with less false positives, or we can lower the threshold of detection, and tune models for increased recall.

For this reason, and to enable comparison with existing results, we use the maximum F1 score (F1*) as our primary metric of model performance. F1* is computed by finding the max F1 score across different detection thresholds:

$$F1^* = max_{t \in [0,1]} F1(t)$$

where *F1(t)* is the F1 score when the detection threshold is set to *t*. We computed F1* with thresholds in the range [0, 1] with increments of 0.01.

LARGE IMAGE INFERENCE

One of the impediments to using CNNs for object detection in WFOV images, is that the size of a model for training and inference on full-size WFOV images is prohibitively large. Such a model would not fit into memory of a single GPU, and even if it did, training time would be extremely long due to the greatly increased number of parameters such a model would have. On the other hand, reducing the resolution of these WFOV images would make the small objects we are interested in detecting be even smaller. Our highest quality PANDORA images have a resolution of 4224x5776px.

Thus, in order to inference on these images without down-sampling, we slice the original 4224x5776px image into 135 overlapping 512x512px images. The 512x512px images are then inferred individually, and detections are combined together and readjusted to reflect their proper position on the original 4224x5776px image. If any detections are reported twice because the detection occurred in the overlapping region, one of the detections is eliminated in order to avoid double counting. A visualization of detections on a full size image is shown in Fig. 2.

An added benefit of this method of inference is that it scales well with multiple GPUs. For faster inferencing, the 135 512x512 slices can be divided up amongst multiple GPUs using multiprocessing. Currently, inferencing a single 4224x5776px image takes approximately 25 seconds on a single Tesla V100 GPU. Using multiprocessing and 4 GPUs, inferencing can be completed in approximately 6-7 seconds.

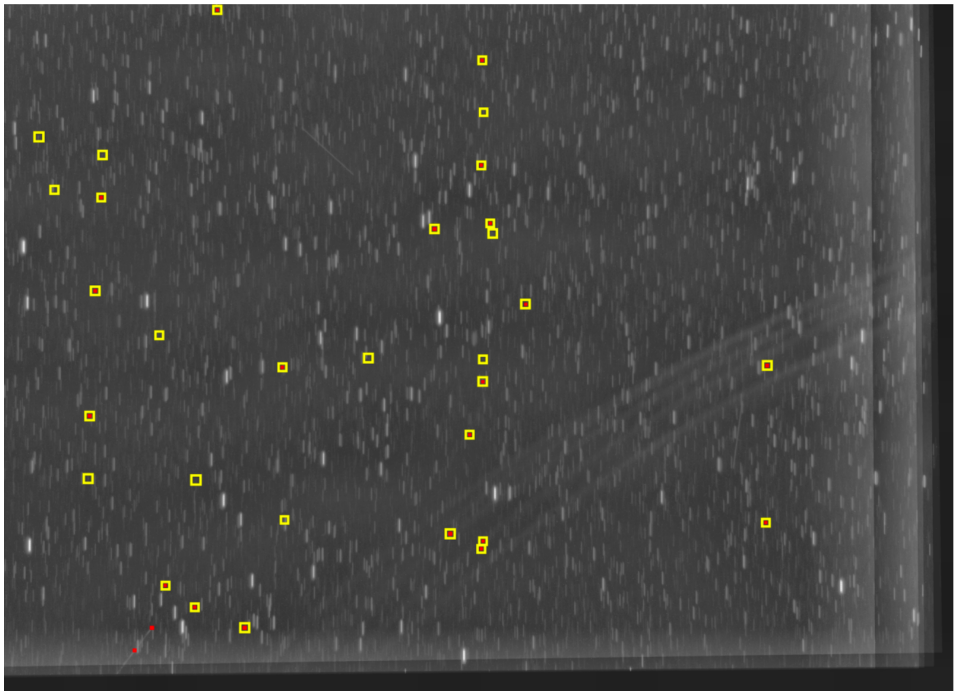

Fig. 2. Results of inference on a full size 4224x5776px image. Yellow boxes are ground truth boxes. Red squares are predictions/detections. Thus, yellow boxes with red squares inside are true positives; yellow boxes alone are false negatives; red squares alone are false positives.

IMAGE AUGMENTATION AND PROCESSING

During our investigation of both the unstacked, single-frame and stacked PANDORA images, we identified significant differences from the SatSim-generated images, which we believe are largely a result of the stacking process. Some key differences between SatSim and PANDORA images include contrast, satellite sizes, presence of artifacts, degree of histogram homogeneity, and star traversal direction (strictly horizontal for SatSim and variable orientations for PANDORA). Representative examples are shown below in Fig. 3.

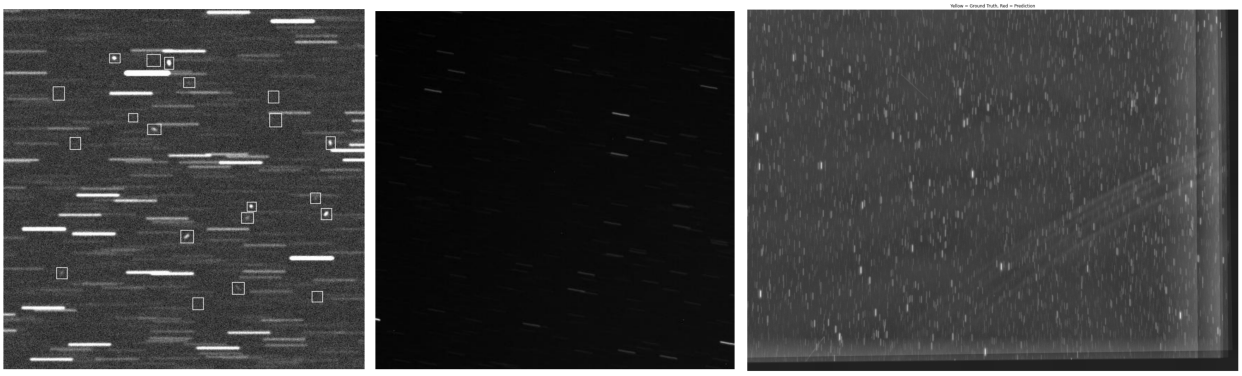

Fig. 3. An image from the SatSim Monte Carlo Dataset (left), a representative unstacked (single-frame) PANDORA image (center) and a representative stacked (multi-frame) PANDORA image (right)

In order to make our model more robust to differences between real PANDORA images and simulated training data, we used a variety of standard as well as novel image augmentation techniques.

Firstly, we performed histogram matching. Histogram matching entails modifying both the intensity of pixels and the frequencies at which the intensities occur in order to match an image's histogram to a reference image. Images in a SatSim dataset have very homogenous histograms, while the histograms of real unstacked PANDORA images vary greatly. Thus by randomly selecting the real reference image during training, we are able to create very heterogeneous training data that have histograms representative of the real data. A representative result of applying histogram matching is shown in Fig. 4..

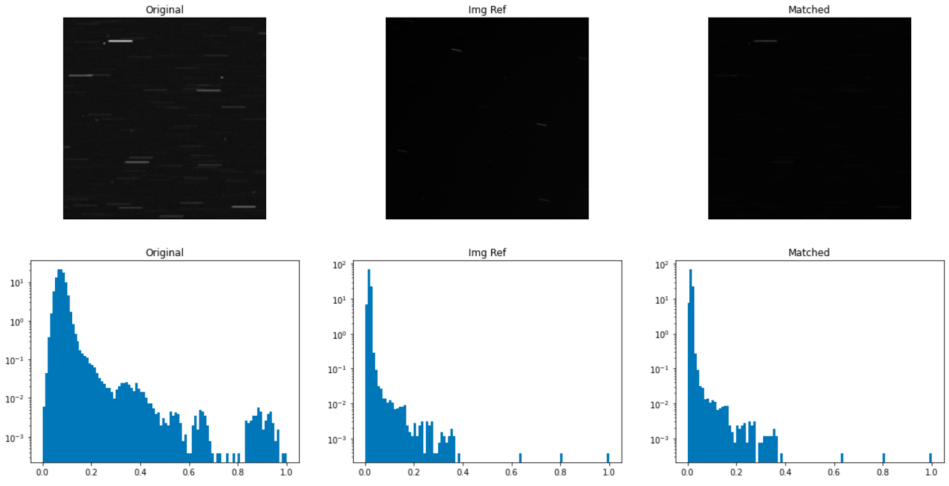

Fig. 4. An example of histogram matching. Left column is the original SatSim image.
Center column is the real reference image. The right column is the histogram-matched SatSim image.

In addition to histogram matching, we performed image rotation to match star streak directionality of SatSim images to that of real PANDORA images. The rest of the augmentations we employed were novel augmentations or adaptations of augmentations in standard image augmentation libraries that needed to be adapted to function.

1. Custom Salt: This augmentation is a random injection of a high intensity pixel in order to simulate "hot pixel" artifacts inherent to digital camera images. This augmentation was necessary because of false positive detections on "hot pixels". We found standard "Salt" augmentations from image augmentation libraries to be ineffective because the intensity of injected pixels was greater than that of real hot pixel artifacts. We manually identified hot pixel artifacts and recorded their pixel intensities in order to create our "Custom Salt" augmentation, which closely approximates hot pixel artifact intensities.
2. Hot blotches: This augmentation is a random injection of small contiguous groups of high intensity pixels (between 1x2px to 3x3px). This augmentation is designed to simulate contiguous groups of hot-pixels, or single hot-pixels which later "bleed" into the surrounding pixels during image post-processing.
3. LEO satellites: We inserted simulated LEO satellites randomly into our training images by pasting high-noise low-intensity varying length 1-2px width streaks within training images. This made our model robust to real LEO satellites within images and greatly reduced false positive detections.
4. Hot pixel removal: In addition to augmenting the training data, we also created a custom gaussian filter in order to remove hot pixels from real PANDORA images. This also reduced our false positive detections.

Results

RESULTS ON SATSIM SIMULATED DATA

A custom RetinaNet model with a ResNet18 backbone trained to completion on the 40,000-image SatSim dataset was evaluated on the test dataset, and the detection model's performance was characterized as a function of the relative brightness of the simulated satellites, based on their signal-to-noise (SNR) values.

Our RetinaNet model was shown to achieve a maximum F-Score of **0.75** for all objects compared to the benchmark score of 0.366 reported by [6]. For bright satellite detection, our model achieved an F1* score of **0.904**, compared to 0.491 for the baseline method.

Table 3. Performance on the 50-image Test Dataset for different SNR thresholds

	All Objects	**Objects with any pixel SNR > 1**	**Objects with every pixel SNR < 1**
F1*	0.750	0.904	0.02
mAP	0.655	0.892	0.02

1. num_objects = Total number of objects in the dataset subset
2. score_threshold = Score threshold at which the maximum FScore (F1*) is found

Fig. 5 below shows that the F1 score is robust across the score (confidence) threshold range of ~0.2 to ~0.8, indicating that the detection model can maximize either precision or recall without a significant negative effect on its overall performance.

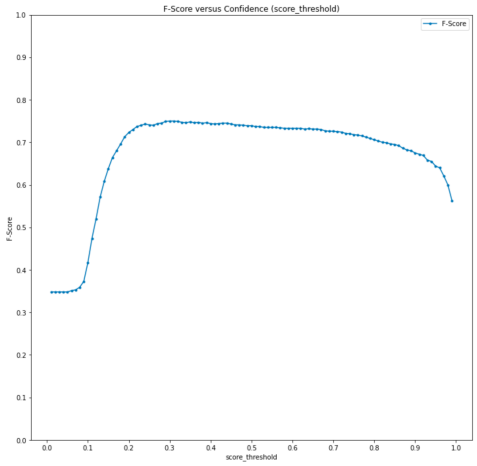

Fig. 5: F-score of the model on the All SNRs data subset at different prediction confidences

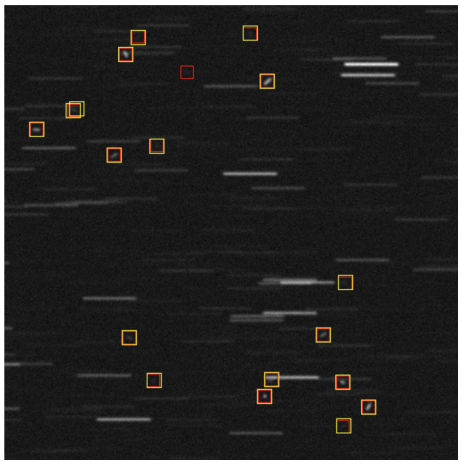

Fig. 6: Example 1 of Any SNR > 1, score_threshold = 0.31

RESULTS ON THE UNSTACKED DATASET

Custom RetinaNet models with ResNet18 backbones were trained for two epochs of 10,000 SatSim images before being evaluated on the 29,302-image unstacked dataset. Details on the models are as follows:

- **Model 1:** Trained with the following image augmentations: Histogram Matching, Rotation, "Custom Salt", **1-2**px LEO satellites, **hot blotches**. On validation: remove hot pixels from validation data.
- **Model 2:** Trained with the following image augmentations: Histogram Matching, Rotation, "Custom Salt", **1px** LEO satellites. On validation: remove hot pixels from validation data.
- **Model 1 + Model 2:** An unoptimized ensemble of model 1 + model 2. In order to count as a true positive or a false positive, both models must agree on the prediction. The ensemble is unoptimized because both models are forced to use the same score threshold.

A maximum F1 score of 0.45 was achieved using a naive ensemble of Model 1 + Model 2. This simple ensemble, which used a heuristic to determine detections, resulted in a 0.03 increase in F1 score. Greater increases in performance can likely be gained from an ensemble wherein the outputs of each model are joined together and input to another neural net responsible for integrating the results from both models and producing an overall result.

Table 4. Comparison of model performance on full unstacked PANDORA dataset

Experiments	F1*	mAP	score_ threshold	Precision	Recall	True Positive (TP)	False Positive (FP)	False Negative (FN)
Model 1	0.42	0.24	0.52	0.35	0.53	3376	6266	3025
Model 2	0.38	0.21	0.51	0.31	0.48	3050	6779	3361
Model 1 + Model 2	0.45	0.26	0.36	0.37	0.56	3603	6078	2808

RESULTS ON THE STACKED DATASET

Due to the size of the stacked dataset, models were evaluated on a random 20% sample of stacked images from a single camera array row (STACK2), containing approximately ~2300 GEOs. Accordingly, we did not expose our model to any images from STACK2 during training.

A model trained on SatSim data alone achieved an F1 score of **0.68** and was later used as the basis for fine-tuning with the raw, unmodified stacked dataset. The fine tuning resulted in a slight performance increase, with an F1 score of **0.73** recorded on a random 20% subset of STACK2.

The fine tuning procedure was as follows: The training dataset for fine-tuning was taken from STACK0, STACK1, STACK3, STACK4 and STACK5, so that the model could avoid exposure to the test dataset. Since each camera in a STACK photographs a specific area of the sky, by avoiding exposure to STACK2 entirely during training, we avoid bias and assure that the object detector has no prior knowledge of the area of sky that will be used for evaluation. A random 512x512px crop was taken from each image in the training stacked dataset. An epoch was complete after a random crop was taken from all images. New random crops were taken in each epoch.

Details on the models are as follows:

- **SatSim Only:** The model with ResNet18 backbone was trained for 20 epochs of 1,000 images on SatSim data. The only augmentation to the SatSim data was 90 degree rotation, in order to match the star orientation.
- **Fine-tuned SatSim:**
    - **Fine-tuned SatSim 1:** The SatSim only model above was fine-tuned for 24 epochs on random 512x512px crops of STACK0-STACK1 and STACK3-STACK5.
    - **Fine-tuned SatSim 2**: The SatSim only model above was trained for 54 epochs on random 512x512px crops of STACK0-STACK1 and STACK3-STACK5.
- **Base Model:** A custom RetinaNet model with ResNet18 backbone was trained from scratch on an augmented version of STACK0-STACK1 and STACK3-STACK5 for 18,000 steps. STACK2 was withheld from the training data so that it could be used for validation. The only augmentation was translation of the training data.
- **SatSim after Real:** The Base Model above with ResNet18 backbone was trained for 2 epochs of 10,000 images on SatSim data. The performance was similar to the Base Model.

Table 5. Comparison of model performance on full unstacked PANDORA dataset

Experiments	F1*	mAP	score_ threshold	Precision	Recall	True Positive (TP)	False Positive (FP)	False Negative (FN)
SatSim Only	0.68	0.62	0.12	0.63	0.74	8447	4877	3022
Fine-tuned SatSim 1*	**0.73**	**0.65**	0.15	**0.67**	**0.79**	1876	917	473
Fine-tuned SatSim 2*	0.71	0.64	0.13	**0.67**	0.74	1698	820	601
Base Model	0.13	0.05	0.57	0.10	0.16	1848	15856	9621
SatSim after Real*	0.13	0.03	0.92	0.12	0.14	314	2366	1932

Future Work

Reference [14] found that YOLO models with spatio-temporal components can achieve RSO detection with better accuracies than single frame YOLO models. We believe multi-frame approaches could have similar benefits for RetinaNet-based architectures. Our existing neural network approach can be modified to synthesize multiple frames in a single step, and produce predictions across the multiple frames. Networks that are designed to predict across multiple frames have the distinct advantage of being able to synthesize information across frames, and overcome poor SNR, photogrammetric issues, or even physical obfuscation in the right conditions [22]. Approaches to this include leveraging recurrent neural layers in the CNN architecture [23] and [14], using transformer layers [24] and Kalman-filter-based approaches [15].

Conclusion

Our RetinaNet-based object detector greatly exceeded the performance of traditional single-image WFOV object detection techniques on simulated data. Our model showed robust precision scores across the spectrum of detection sensitivities. When considering bright satellites with an SNR > 1, our model achieved an F1* score of 0.904. When considering satellites of all brightness together, we achieved an F1* of 0.75.

Our detector also performed well on real PANDORA images, achieving an F1* score of 0.73 on a random sample of stacked PANDORA images. We made extensive use of standard and custom image augmentation techniques to increase robustness to digital camera artifacts such as hot pixels and hot blotches, as well as to decrease false positive detections on other space objects such as LEO satellites.

We overcame the challenges that using CNN detectors on large images presents by slicing 4224x5776 pixel images into 135 512x512 pixel crops, for which detections can be performed individually. This allowed us to maintain the resolution of target objects, and thus harness the full capability of the detector. Approximately 400 systematically undertaken experiments enabled us to optimize the neural network structure. We were able to use the smallest Res-Net backbone, as well as to discard 80% of feature maps used in default RetinaNet object detectors. Finally, we reduced the amount of prediction bounding boxes used by the detector to a single size bounding box with a single aspect ratio. Together, these optimizations allowed us to infer detections on 4224x5776 pixel stacked PANDORA images, comprising a 14x19 FOV, in less than half a minute on a commercially available GPU. Our results show that while CNNs are generally not thought of as suitable for small object detection in large images, they can in fact produce results that greatly exceed that of traditional methods, with very reasonable inference-compute times when the appropriate adaptations are made.

References

1. Cireşan, Dan C., et al. "Mitosis detection in breast cancer histology images with deep neural networks." *International conference on medical image computing and computer-assisted intervention*. Springer, Berlin, Heidelberg, 2013.
2. Sun, Yi, et al. "Deepid3: Face recognition with very deep neural networks." *arXiv preprint arXiv:1502.00873,* 2015.
3. Tseng, Yu-Ho, and Shau-Shiun Jan. "Combination of computer vision detection and segmentation for autonomous driving." *2018 IEEE/ION Position, Location and Navigation Symposium (PLANS)*. IEEE, 2018.
4. Alexander Cabello and Justin Fletcher "SatSim: a synthetic data generation engine for electro-optical imagery of resident space objects", Proc. SPIE 12121, *Sensors and Systems for Space Applications XV*, 1212107 (3 June 2022); https://doi.org/10.1117/12.2618733
5. Bertin, Emmanuel, and Stephane Arnouts. "SExtractor: Software for source extraction" *Astronomy and astrophysics supplement series* 117.2: 393-404 (1996).
6. Fitzgerald, Garrett, *et al.*, "Toward Deep-Space Object Detection in Persistent Wide Field of View Camera Arrays." *Advanced Maui Optical and Space Surveillance Technologies Conference (AMOS)* (September 2021)
7. Krizhevsky Alex, Sutskever Ilya, and Hinton Geoffrey E.. 2012. ImageNet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems 25*, F. Pereira, Burges C. J. C., L. Bottou, and Weinberger K. Q.(Eds.). Curran Associates, Inc., 1097–1105.
8. Zhao, Zhong-Qiu, Peng Zheng, Shou-tao Xu, and Xindong Wu. "Object detection with deep learning: A review." *IEEE transactions on neural networks and learning systems* 30, no. 11 (2019): 3212-3232.
9. Wu, Xiongwei, Doyen Sahoo, and Steven CH Hoi. "Recent advances in deep learning for object detection." *Neurocomputing* 396 (2020): 39-64.
10. Hafiz, A.M., Bhat, G.M. A survey on instance segmentation: state of the art. *Int J Multimed Info Retr* 9, 171–189 (2020). https://doi.org/10.1007/s13735-020-00195-x
11. Lin, Tsung-Yi, et al. "Microsoft coco: Common objects in context." *European conference on computer vision*. Springer, Cham, 2014.
12. Deng, Jia, et al. "Imagenet: A large-scale hierarchical image database." *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009.
13. Zaidi SSA, Ansari MS, Aslam A, Kanwal N, Asghar MN, Lee BA (2021) A survey of modern deep learning based object detection models. *arXiv*:2104.11892, 2021.
14. Fletcher, Justin, et al. "Feature-Based Satellite Detection Using Convolutional Neural Networks" *Proceedings of the Advanced Maui Optical and Space Surveillance Technologies Conference*, 2019
15. Garrett Fitzgerald, Ruixu Liu, Vijayan Asari, "Geosynchronous satellite detection and tracking with WFOV camera arrays using spatiotemporal neural networks (GEO-SPANN)," Proc. SPIE 12101, Pattern Recognition and Tracking XXXIII, 1210104 (27 May 2022); https://doi.org/10.1117/12.2618047
16. Lin, Tsung-Yi, Priya Goyal, Ross Girshick, and Piotr Dollar. 2017. "Focal Loss for Dense Object Detection." *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (October 2017): 2999-3007.
17. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
18. Sengupta, Abhronil, et al. "Going deeper in spiking neural networks: VGG and residual architectures." *Frontiers in Neuroscience* 13 (2019): 95.
19. Xie, Saining, et al. "Aggregated residual transformations for deep neural networks." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.
20. Gao, Chao, Martin Müller, and Ryan Hayward. "Three-Head Neural Network Architecture for Monte Carlo Tree Search." *IJCAI*. 2018.
21. Weiss, Karl, Taghi M. Khoshgoftaar, and DingDing Wang. "A survey of transfer learning." *Journal of Big Data* 3.1 (2016): 1-40.
22. Gioele Ciaparrone, Francisco Luque Sánchez, Siham Tabik, Luigi Troiano, Roberto Tagliaferri, Francisco Herrera, "Deep learning in video multi-object tracking: A survey" Neurocomputing, Volume 381, 2020, pp 61-88, doi: https://doi.org/10.1016/j.neucom.2019.11.023.
23. Alex Broad, Michael J. Jones, and Teng-Yok Lee. 2018. Recurrent Multi-frame Single Shot Detector for Video Object Detection. In *Proceedings of the 29th British Machine Vision Conference.* BMVA, Northumbria, UK, 94.

24. Z. Yang et al., "3D-MAN: 3D Multi-frame Attention Network for Object Detection", *arXiv:2103.16054*, 2021.