REPORT DOCUMENTATION PAGE

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED	
		START DATE	END DATE
04/23/2024	Research		

4. TITLE AND SUBTITLE

Comparative Analysis of Optical Sensor Artifacts across Neural Network-based ATR Algorithms

5a. CONTRACT NUMBER	5b. GRANT NUMBER	5c. PROGRAM ELEMENT NUMBER
5d. PROJECT NUMBER	**5e. TASK NUMBER**	**5f. WORK UNIT NUMBER**
		996C61

6. AUTHOR(S)

Quinton Davidson

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

US Naval Research Laboratory
4555 Overlook Ave SW
Washington, DC 20375

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

US Naval Research Laboratory
4555 Overlook Ave SW
Washington, DC 20375

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Distribution Statement A. Approved for public release: distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

With the proliferation of space-based optical systems and corresponding increased volume of overhead imagery data, there is a growing need for automatic target recognition (ATR) algorithms that both effectively identify objects of interest and remove the burden of analysis from a finite number of human ground operators. Although recent state-of-the-art (SOTA) deep learning architectures like Convolutional Neural Networks (CNNs) and Detection Transformers (DETRs) have shown great performance in accomplishing these tasks, this performance can degrade substantially when operating on data containing unexpected anomalies outside their original training sets. Likewise, the increased amount of automation in these processes magnifies the negative impact that an ATR algorithm can cause prior to a human analyst recognizing any problem in the data downstream. Space-based optical systems rely on accurate calibration to create clean imagery, and as such, these ATR algorithms are subject to sensor calibration artifacts. Previous work has characterized common calibration artifacts such as sensor noise and failed detector artifacts as they affect the performance of an Inceptionv1-based ATR algorithm. This paper looks to extend this analysis to multiple CNN and transformer-based object detection architectures to characterize differences in performance degradation across various SOTA ATR algorithms. Notably, we found the RT-DETR architecture is more robust to uniformly distributed random scaling factors and random pixel failures than YOLOv8, YOLOv9, and Faster-RCNN particularly when detecting large objects like container ships and tankers. These results are useful in summarizing the expected performance impact of common calibration artifacts on ATR algorithms as well as informing algorithm selection when designing systems that leverage overhead imagery.

15. SUBJECT TERMS

Convolutional Neural Network (CNN), Detection Transformer (DETR), neural network, machine learning, detection, sensor artifact, image perturbation, remote sensing

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES
a. REPORT Unclassified	b. ABSTRACT Unclassified	C. THIS PAGE Unclassified		16

19a. NAME OF RESPONSIBLE PERSON	19b. PHONE NUMBER *(Include area code)*
Quinton Davidson	202-767-1413

INSTRUCTIONS FOR COMPLETING SF 298

1. REPORT DATE.
Full publication date, including day, month, if available. Must cite at least the year and be Year 2000 compliant, e.g. 30-06-1998; xx-06-1998; xx-xx-1998.

2. REPORT TYPE.
State the type of report, such as final, technical, interim, memorandum, master's thesis, progress, quarterly, research, special, group study, etc.

3. DATES COVERED.
Indicate the time during which the work was performed and the report was written.

4. TITLE.
Enter title and subtitle with volume number and part number, if applicable. On classified documents, enter the title classification in parentheses.

5a. CONTRACT NUMBER.
Enter all contract numbers as they appear in the report, e.g. F33615-86-C-5169.

5b. GRANT NUMBER.
Enter all grant numbers as they appear in the report, e.g. AFOSR-82-1234.

5c. PROGRAM ELEMENT NUMBER.
Enter all program element numbers as they appear in the report, e.g. 61101A.

5d. PROJECT NUMBER.
Enter all project numbers as they appear in the report, e.g. 1F665702D1257; ILIR.

5e. TASK NUMBER. Enter all task numbers as they appear in the report, e.g. 05; RF0330201; T4112.

5f. WORK UNIT NUMBER.
Enter all work unit numbers as they appear in the report, e.g. 001; AFAPL30480105.

6. AUTHOR(S). Enter name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. The form of entry is the last name, first name, middle initial, and additional qualifiers separated by commas, e.g. Smith, Richard, J, Jr.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES). Self-explanatory.

8. PERFORMING ORGANIZATION REPORT NUMBER.
Enter all unique alphanumeric report numbers assigned by the performing organization, e.g. BRL-1234; AFWL-TR-85-4017-Vol-21-PT-2.

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES). Enter the name and address of the organization(s) financially responsible for and monitoring the work.

10. SPONSOR/MONITOR'S ACRONYM(S). Enter, if available, e.g. BRL, ARDEC, NADC.

11. SPONSOR/MONITOR'S REPORT NUMBER(S). Enter report number as assigned by the sponsoring/monitoring agency, if available, e.g. BRL-TR-829; -215.

12. DISTRIBUTION/AVAILABILITY STATEMENT. Use agency-mandated availability statements to indicate the public availability or distribution limitations of the report. If additional limitations/ restrictions or special markings are indicated, follow agency authorization procedures, e.g. RD/ FRD, PROPIN, ITAR, etc. Include copyright information.

13. SUPPLEMENTARY NOTES. Enter information not included elsewhere such as: prepared in cooperation with; translation of; report supersedes; old edition number, etc.

14. ABSTRACT. A brief (approximately 200 words) factual summary of the most significant information.

15. SUBJECT TERMS. Key words or phrases identifying major concepts in the report.

16. SECURITY CLASSIFICATION. Enter security classification in accordance with security classification regulations, e.g. U, C, S, etc. If this form contains classified information, stamp classification level on the top and bottom of this page.

17. LIMITATION OF ABSTRACT. This block must be completed to assign a distribution limitation to the abstract. Enter UU (Unclassified Unlimited) or SAR (Same as Report). An entry in this block is necessary if the abstract is to be limited.

PREVIOUS EDITION IS OBSOLETE.

STANDARD FORM 298 (REV. 5/2020)
Prescribed by ANSI Std. Z39.18

A Comparative Analysis of Optical Sensor Artifacts across Neural Network-based ATR Algorithms

Quinton T. Davidson

US Naval Research Laboratory, 4555 Overlook Ave, Washington, DC, USA

ABSTRACT

With the proliferation of space-based optical systems and corresponding increased volume of overhead imagery data, there is a growing need for automatic target recognition (ATR) algorithms that both effectively identify objects of interest and remove the burden of analysis from a finite number of human ground operators. Although recent state-of-the-art (SOTA) deep learning architectures like Convolutional Neural Networks (CNNs) and Detection Transformers (DETRs) have shown great performance in accomplishing these tasks, this performance can degrade substantially when operating on data containing unexpected anomalies outside their original training sets. Likewise, the increased amount of automation in these processes magnifies the negative impact that an ATR algorithm can cause prior to a human analyst recognizing any problem in the data downstream. Space-based optical systems rely on accurate calibration to create clean imagery, and as such, these ATR algorithms are subject to sensor calibration artifacts. Previous work has characterized common calibration artifacts such as sensor noise and failed detector artifacts as they affect the performance of an Inceptionv1-based ATR algorithm. This paper looks to extend this analysis to multiple CNN and transformer-based object detection architectures to characterize differences in performance degradation across various SOTA ATR algorithms. Notably, we found the RT-DETR architecture is more robust to uniformly distributed random scaling factors and random pixel failures than YOLOv8, YOLOv9, and Faster-RCNN particularly when detecting large objects like container ships and tankers. These results are useful in summarizing the expected performance impact of common calibration artifacts on ATR algorithms as well as informing algorithm selection when designing systems that leverage overhead imagery.

Keywords: Convolutional Neural Network (CNN), Detection Transformer (DETR), neural network, machine learning, detection, sensor artifact, image perturbation, remote sensing

1. INTRODUCTION

Since the launch of Ikonos in 1999, commercial remote sensing satellites have been invaluable in providing images of the earth, enabling a number of different monitoring capabilities across various scientific and military endevours.[1] Recent years have seen improvements in hardware components that have increased the computing power and reduced the size and cost of deploying these sensors.[2] Coupled with the proven viability of commercial rocket launches,[3] this has led to several order of magnitude increases in the number of available commercial satellites for remote sensing missions, see Figure 1. This trend is only going to continue as the 7562 satellites orbiting Earth in May 2023[4] are expected to increase by another 58000 satellites by the year 2030.[5]

One area making use of these sensors is maritime domain awareness (MDA). Remote sensors allow for broad surveillance of the Earth's oceans which in turn enables the ability to monitor global ship traffic, enforce border control, and track illegal fishing activity.[6–8] The huge surface area of this search space combined with massive influx of electro-optical (EO) data coming from proliferated imaging constellations mean that the current number of image analysts responsible for performing this job cannot scale to match.[9] This has led to a shift where the previously manual work is being tasked to automated deep learning algorithms. Specifically, neural networks in the form of convolutional neural networks (CNNs) are being used to solve the maritime domain awareness problem.[10–13]

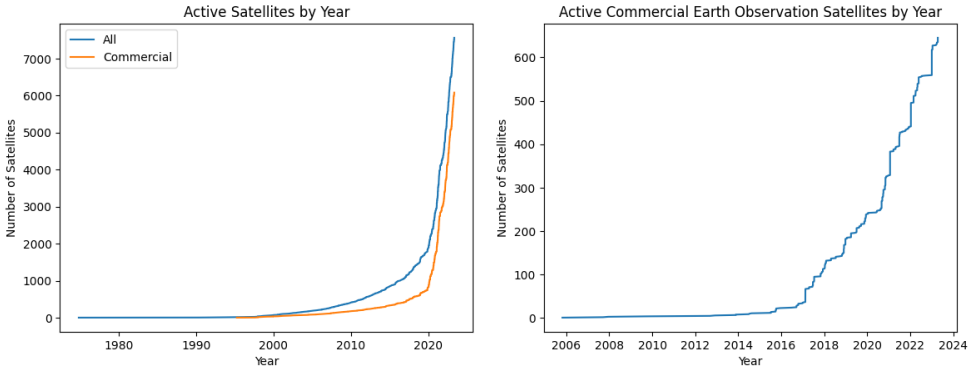

Figure 1. Figure shows active number of satellites by year. The left graph describes the number of overall and commercial satellites, and the right describes specifically commercial EO imaging satellites. Graphs are derived from the UCS Satellite database.[4]

As in other industries and applications however, machine learning algorithms can be extremely brittle when operating on data outside their training sets. It can be difficult to perfectly model all potential variables in an operational environment, and it is likewise difficult to collect and annotate the requisite data to generalize to an unknown test distribution.[14] Operational models can fail in ways that seem unexpected, and the reason a model made a decision on a piece of data can be traced to overfitted features from the training set that don't correlate to semantic ones that a human would use to perform the same task.[15]

On the sensor side, the proliferation in the number of imaging satellites will increase the difficulty of monitoring each individual sensor's health and maintaining the quality of the image products being inferenced by these algorithms. Sensor calibration artifacts resulting from normal use in the radiation of space could lead to intrinsic image characteristics that are both unintended and go unnoticed. Although much work has gone into robustness of deep learning algorithms as they relate to adversarial attacks, much less has gone into evaluating these algorithms to the less targeted sensor artifacts that an imaging satellite could experience on orbit.[16]

This paper seeks to extend research completed in 17 on how sensor calibration artifacts impact binary ship classifier performance. Although this work was meaningful in characterizing the effects on identifying an image as a ship, it was limited to a single model, and it doesn't cover the other half of the maritime monitoring problem, that is actually localizing the object of interest to be classified. With advances in neural network architectures over the last decade, object detector algorithms are tasked to perform both the localization of maritime objects of interest in addition to the classification of said objects.[18] The goal of this paper is to address that gap by characterizing the impact of calibration artifacts on object detection algorithms.

The subsequent sections in this paper outline and present the results of an experiment assessing the robustness of four state-of-the-art (SOTA) object detection algorithms to the anticipated imaging sensor artifacts that may occur over the lifecycle of a commercial imaging satellite. Section 2 describes the data, models, anomalies, and metrics used in the experiment. Section 3 presents the data collected from the experiment and interprets it as it relates to neural networks leveraging space based imaging sensors. A summary of the work and conclusions can be found in Section 4.

2. METHODOLOGY

This experiment uses commercial satellite EO imagery to fine-tune several object detection models to identify ships in overhead imagery. These models are then evaluated on images that have had artificial calibration artifacts injected into them, and model performance is compared across model and artifact type. This section defines the data, models, inferencing procedure, and metrics used to perform the experiment.

2.1 Data

The following experiment uses EO imagery from the WorldView-2 (WV2) and WorldView-3 (WV3) commercial imaging satellites. These satellites operate in low earth orbit (LEO) and provide panchromatic (PAN) imagery at ground sampling distances (GSD) as low as 0.5m and 0.3m respectively. Both systems image in the visible spectrum and produce images with a native bit depth of 16. Refer to 19 and 20 for additional information regarding WV2 and WV3.

For the purposes of this experiment, the original 16-bit images are downsampled to 8-bit to match the expected input bit depth of the object detection models. From here, the data is further curated into a training set for model fine-tuning and a test set for evaluating the impact of sensor calibration artifacts on model performance.

The training set consists of 9165 bounding box labeled image chips taken from the WorldView source images. Each sample is 512x512 PAN as 3-band RGB (each band is the same) and maintains the same resolution as the source image. Of the 9165 image chips, 7332 are used for training and 1833 for validation. Of the 7332 training image chips, 1000 are unannotated background images to help reduce the false positive rates of the models. Although the train/validate images are intentionally split 80-20, the number of annotated/unannotated training images are more indicative of the available image chips during training.

Likewise, the test set consists of 1000 bounding box labeled WorldView images. These images are approximately 5555x5555 pixel PAN as 3-band RGB. The larger image size better reflects the type of operational data computer vision algorithms are being tasked to find ships in. Given this, the distribution of ships by image is relatively sparse by pixel area. See Figure 2 for a distribution of object count by image.

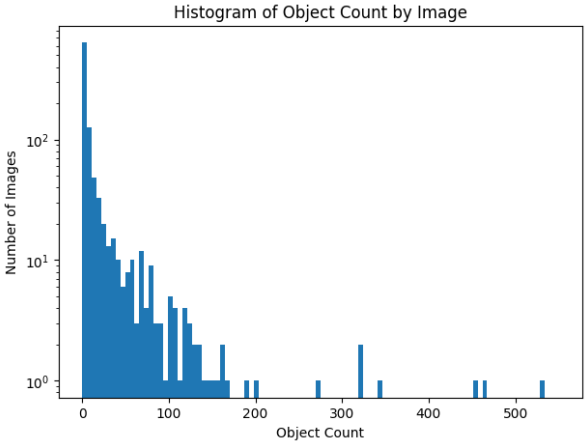

Figure 2. Figure shows the distribution of ship count by test image. The test set contains 15841 ships across 1000 images with each image having at least 1 ship. 765 images contain between 1 and 10 ships while 7 images contain over 200 ships.

Both the training and test data were labeled with bounding boxes using the LabelStudio open source data labeling tool.[21] Images were preannotated with an existing fine-tuned YOLOv8 ship detector, and those predicted labels were validated/augmented by human labelers. See Figure 3 for examples of training and test images.

2.2 Models Evaluated

The goal of the experiment was to evaluate the robustness of several modern SOTA detection models to the calibration artifacts that an EO imaging satellite would experience over its lifecycle. Among these, detection models can be broadly categorized as two stage or one stage detectors. Two stage detectors typically use one network to generate bounding box proposals for objects in an image and another to classify those objects into classes and perform bounding box regression.[22–24] One stage detectors forgo the object proposal step and perform a single regression problem to generate bounding boxes and class probabilities.[25,26]

For our two stage detector, we selected the Faster-RCNN model architecture, introduced in 24. Faster-RCNN introduced a Region Proposal Network (RPN) that shared convolutional features with its detection network,

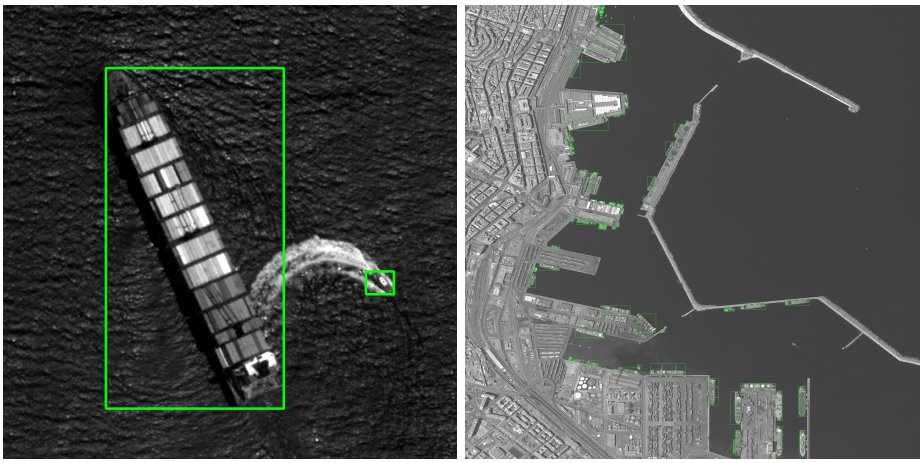

Figure 3. Figure shows an example annotated training image side by side with an example annotated test image. The training images have a resolution of 512x512, and the test images have a resolution of approximately 5555x5555. Original images ©MAXAR.

eliminating the region proposal bottleneck of its predecessor and speeding up inference times. We use the second version of Faster-RCNN with Resnet50 backbone provided by Pytorch.[27]

Among one stage detectors, we examine differences between two of the predominant deep learning architectures in computer vision, CNN-based and transformer-based. Broadly, CNNs work by convolving filters across images to produce features that are pooled together and passed to downstream networks for the particular task of interest (classification, detection, segmentation, etc.).[28] For transformer-based models, we use detection transformers (DETRs). DETRs use a CNN backbone to generate image features before flattening and passing them with a positional encoding to a transformer network.[29,30] This transformer encodes these tokenized features and finds relations between them with attention layers before decoding them to bounding boxes to be evaluated at the end of the overall network.

For our DETR, we use the extra large Real-Time-Detection-Transformer (RT-DETR) model introduced in 31. RT-DETR improves on the original DETR architecture by introducing an efficient hybrid encoder and IOU-aware query selection to allow it to perform high accuracy real time object detection. We study the rtdetr-x model specifically.

For the strictly CNN-based detectors, we test several YOLO based detectors, namely YOLOv8 and YOLOv9. YOLOv8 was originally selected as the SOTA YOLO model for this study. It introduced backbone improvements and implemented an anchor-free detection head that improved the speed and localization accuracy over its predecessors.[32] However, due to YOLOv9 recently surpassing it as the SOTA YOLO model, we examine YOLOv9 to see how its robustness compares to YOLOv8. YOLOv9 introduced progammable gradient decent (PGI) to counter information lost in deep recurrent networks as well as a lightweight network architecture that reduces the overall parameter count.[33] We study the yolov8x and yolov9e models for YOLOv8 and YOLOv9 respectively.

Both the YOLO and RT-DETR models used in this experiment were finetuned from pretrained models released by Ultralytics.[32] Refer to Table 1 for more information about the selected models.

Table 1. Description of specific models with parameter count and computing power.

Model	# Parameters (M)	FLOPs (G)
faster_rcnn_resnet_50_fpn_v2	43.3	561.6
rtdetr-x	67.3	149.9
yolov8x	68.2	165.2
yolov9e	57.4	123.3

2.3 Training

All models were trained on a NVIDIA RTX 6000 Ada Generation GPU[34] using the training data described in Section 2.1. These models were initialized with weights pretrained on the MS COCO 2017 dataset and finetuned on the training set for 300 epochs. The Stochastic Gradient Descent (SGD) optimizer was chosen for training, and each model used a specified learning rate with a 3 epoch linear warmup followed by a linear decay for the remaining epochs as given in Table 5. Mosaic augmentation was applied during training but was turned off for the final 10 epochs. Additionally, random perspective and HSV augmentations were applied, and images were flipped horizontally and vertically with a probability of 0.5. Automated mixed precision was enabled during training. For more information on the training hyperparameters, please refer to Table 5 in the appendix. It is likely that individual model performance could have been improved with hyperparameter tuning on a per-model basis, however the goal of the experiment was to study the effects of calibration artifacts rather than maximizing baseline performance metrics. As a result, hyperparameter tuning was omitted.

2.4 Image Calibration Artifacts

As an extension of research performed in 17, we study the impact of the calibration artifacts introduced in that previous work. These remain an anticipated set of EO sensor anomalies that a deep learning model would experience in its image input given calibration procedures for satellite imaging sensors.[35–38] In the following subsections, we redefine these artifacts and note the differences in this paper's experimental implementation of them. See Figure 4 for examples of the simulated calibration artifacts visualized on a sample image.

Figure 4. Figure shows an example image compared to the same image with sensor artifacts added. From left to right, the images are as follows: 1) Unaltered. 2) Bad Gain with $\lambda = 0.3$. 3) Bad offset with $\lambda = 30$. 4) Failed pixels with 25% drop rate. 5) Gaussian Blur with (7, 7) kernel. Original images ©MAXAR.

2.4.1 Photoresponse Nonuniformity

Ideally, pixel array manufacturing for digital imaging sensors (CCD and CMOS) would produce identical pixels across each optical unit. In practice however, irregularities during etching or doping can introduce local offsets where some pixels gain area at the expense of their surrounding neighbors. The resulting variation in detector size and coating thickness causes corresponding variations in pixel responsivity when the device is illuminated. This variation in pixel responsivity is called photoresponse nonuniformity (PRNU).[39] Normally, the individual pixel gain values required to normalize the pixel array are computed during calibration and applied on subsequent image captures to correct for PRNU.

We simulate PRNU by multiplying each image by a mask of random scaling factors representing individual pixel gains. Given an input image i, the bad gain image I is defined in Equation 1.

$$I(x, y) = (\mu + \lambda * \beta(x, y)) * i(x, y) \tag{1}$$

where μ is always set to 1. $\beta(x, y)$ is a random number sampled from a normal Gaussian distribution. This experiment examines the impact of bad gain for $\lambda \in \{0.01, 0.05, 0.1, 0.3\}$.

2.4.2 Dark Current/Offset Correction

Most digital imaging sensors experience a lighting-independent dark response during image capture that stems from current leakage in the photo-sensing diode.[39] This current leakage is caused by thermal generation in or around the sensor, and it increases with the camera integration time. Dark current varies between pixels, and this variation is referred as fixed-pattern-noise (FPN). To correct for this, individual pixel offsets will be calibrated by imaging a black body and measuring the respective pixel dark responses.[40] Combined with the PRNU gain, this offset provides a mapping from sensed Digital Number (DN) and measured light level.

To simulate a bad dark current offset, we shift each image by a mask consisting of Poisson distributed random numbers. Given an input image i, the bad offset corrected image I is defined in Equation 2.

$$I(x, y) = i(x, y) + P((x, y); \lambda) \tag{2}$$

where the random number is sampled from the Poisson distribution defined as

$$P\left((x, y); \lambda\right) = \frac{e^{-\lambda}\lambda^{x}}{x!} \tag{3}$$

This distribution $P\left((x, y); \lambda\right)$ describes the probability of events happening over the image sized (x, y) within the observed interval λ. This experiment examines the impact of bad offset error for $\lambda \in \{5, 10, 20, 30\}$.

2.4.3 Bad Pixels

Digital imaging sensors may experience pixel failures for a number of reasons including but not limited to poor quantum efficiency, high dark current, excessive read noise, and non-linearity from sensor faults.[41] As with the other artifacts highlighted previously, calibration efforts aim to identify and remove the negative impact of these failed pixels, however due to normal maintenance cadence, it is not always possible to have them constantly accounted for.

In the previous experiment, a single pseudo-random sequence of pixel coordinates was generated to mask all the test images evaluated. The use of a single mask was chosen for traceability, however the mask had a relatively small number of pixels get dropped ($< 1\%$ pixels). As a result, the dropped pixels mostly impacted background water area that would have otherwise produced a low classification score. Additionally, the ground truth ships affected were located in the same area across the images. We seek to solve these issues by applying bad pixels in a way that affects objects agnostic of location and allows us to discriminate between differences in model performance by the relative number of pixels that failed.

To do this, we apply a different random mask to each test image where the number of dropped pixels is determined by a probability p. For an image of size (h, w) and failure probability $p \in [0, 1]$, we create an array of length $h \times w$ and set the first $h \times w \times p$ to 0. The rest of the array is set to 1. This array is randomized and then reshaped to (h, w) before being multiplied element-wise with the original image to get our final transformed image. This experiment examines random bad pixel errors for $p \in \{0.1, 0.25, 0.5, 0.75\}$.

2.4.4 Failed Region

The failed region artifact from the original experiment is omitted from this analysis. This is because dropping an entire set of pixels in an area of the image is the same as occluding any objects that occur in the corresponding location. Existing data augmentation strategies like mosaicing already exist to counter occlusion, and the prior work saw minimal effects when implementing a static regional pixel failure. For these reasons, we omit the failed region artifact.

2.4.5 Blurring

Digital imaging sensors used in remote earth observation often require focus calibration to guarantee the clarity of produced imagery. During satellite imaging calibration, there can be an elapsed period several months while focus is adjusted to match the residual settling of optical components.[42]

We expand the research in this area from the original work. A Gaussian blur filter is applied to our test images to simulate defocused satellite imagery. In addition to the 7 pixel kernel used in the previous experiment, we examine Gaussian blurring with 5 and 9 pixel kernels to measure any differences between these intensities.

2.4.6 Test Parameters

Across calibration artifact types, we look at varying intensities of each artifact to characterize how the corresponding model performance degradation changes. Table 2 summarizes the specific cases examined in this experiment.

Table 2. Table of calibration artifact cases examined.

Artifact	Case
Gain	$\sigma = 0.01$
Gain	$\sigma = 0.05$
Gain	$\sigma = 0.1$
Gain	$\sigma = 0.3$
Offset	$\sigma = 5$
Offset	$\sigma = 10$
Offset	$\sigma = 20$
Offset	$\sigma = 30$
Bad Pixels	$p = 0.1$
Bad Pixels	$p = 0.25$
Bad Pixels	$p = 0.5$
Bad Pixels	$p = 0.75$
Blur	kernel=(5, 5)
Blur	kernel=(7, 7)
Blur	kernel=(9, 9)

2.5 SAHI

Detecting ships in satellite EO imagery is a particularly challenging problem for deep learning computer vision algorithms. Images are very high resolution (upwards of 10000x10000 pixels), and objects of interest are very small in comparison (on the magnitude of tens of pixels depending on sensor GSD). Training models to fit the input size of those images is infeasible due to the model size and compute power required. On the other hand, resizing the images down to the common input sizes of popular object detection architectures (anywhere from 212x212 to 640x640) causes a loss of image features that are required to identify the relatively small objects present.

Slicing Aided Hyper Inference (SAHI) was introduced as a generic framework for slicing aided inference for small object detection.[43] SAHI turns images into a series of overlapping patches which are then resized while maintaining their individual aspect ratios. Object detection inference is performed independently on each patch before combining the object predictions and merging them back to the original image size. An optional inference can be made on the entire image to help detect large objects.

We use SAHI to perform all the object detection inferencing on the test data. Instances of SAHI are creating for each of the 4 detection models examined in this paper. Each SAHI instance is configured with a confidence threshold of 0.5, slice height of 512, slice width of 512, overlap height ratio of 0.25, and overlap width ratio of 0.25. Since none of the objects of interest are large (with respect to image dimensions), we do not perform a full inference with SAHI on the base image. These parameters could be further tuned to increase baseline performance, but again the goal is to measure relative performance levels given artifacts, so further tuning was not done. For each model, we run it on the baseline test images in addition to copies of these images augmented by each of the calibration artifacts outlined in Section 2.4.6

2.6 Metrics

To evaluate the performance of our fine-tuned object detection models, we use metrics introduced by the Common Objects in Context (COCO) object detection challenge.[44,45] We utilize the TorchMetrics[46] library to generate these for each of our models and calibration artifacts. Refer to 3 for the entire list of metrics.

Table 3. Table describes the object detection performance metrics used from the COCO object detection challenge.

Average Precision (AP)	
AP	AP at IoU=.50:.05:.95 (primary challenge metric)
AP^{50}	AP at IoU=.50 (PASCAL VOC metric)
AP^{75}	AP at IoU=.75 (strict metric)
AP^{small}	AP for small objects: $area < 32^2$
AP^{medium}	AP for medium objects: $32^2 < area < 96^2$
AP^{large}	AP for large objects: $area > 96^2$
Average Recall (AR)	
AR	AR at IoU=.50:.05:.95
AR^{small}	AR for small objects: $area < 32^2$
AR^{medium}	AR for medium objects: $32^2 < area < 96^2$
AR^{large}	AR for large objects: $area > 96^2$

Among them, average precision (AP), or mean average precision (mAP) when looking at multiclass detectors, is the most commonly used metric to measure object detector performance. Average precision is defined as the area under the precision-recall curve.[47] For the main AP calculation, this is calculated at intersection over union (IoU) thresholds from 0.5 to 0.95 in 0.05 increments and averaged between them. Calculating it this way provides a better metric for localization than just using the Pascal VOC AP at an IoU threshold of 0.5. For this analysis, we consider AP at each of COCO's IoU and object area size thresholds. Evaluating across size thresholds is particularly important as performance can vary widely in identifying a small object (i.e. a sailboat or pleasure vessel) vs a large object (i.e. a container ship or oil tanker)

Average recall describes the average of detector recalls calculated across IoU thresholds [0.5, 1.0].[48] We examine the overall AR as well as AR at the COCO object area size thresholds.

3. RESULTS

Table 4 describes the baseline performance of each of the detector models inferenced with SAHI on the unaltered test dataset. Each model performs within expectations given the size and complexity of the datasets. Across most AP and AR metrics, RT-DETR performs the best, although performance is generally comparable, particularly between RT-DETR and both YOLO models.

Table 4. Baseline performance of models with SAHI on test dataset.

	AP	AP^{50}	AP^{75}	AP^{s}	AP^{m}	AP^{l}	AR	AR^{s}	AR^{m}	AR^{l}
Faster-RCNN	37.0	57.5	39.4	33.5	46.2	46.6	50.9	46.9	57.9	62.3
RT-DETR	**40.9**	**61.3**	43.1	39.0	**54.8**	**55.6**	**55.0**	**49.4**	**64.6**	**71.2**
YOLOv8	40.3	58.1	43.7	**39.4**	54.0	50.7	53.1	47.7	63.6	67.0
YOLOv9	40.7	58.4	**44.0**	39.4	53.6	51.4	52.5	47.2	62.6	66.1

Figure 5 shows the AP score comparison for the various detector models across individual sensor artifact and artifact intensity. Overall the degree of performance drop by AP increases from bad offset to bad gain to blur to bad pixels for the artifact intensities that we examined. Bad gain and blur see the models stratify somewhat in performance, while these models converge to similar AP scores at higher bad pixel and bad offset intensities.

Each model saw a roughly linear drop in AP as the bad gain λ increased. They experienced a sharp drop in AP for minor bad offset intensity $\lambda = 5$ which leveled off as intensity increased. Bad pixels caused a stark two-phase trend in artifact intensity and AP score. Dropping 10% of pixels caused between a 35% and 40% decrease in AP across models, and further increasing the intensity lead to much smaller change. For blur, model AP decreased linearly across kernel size k.

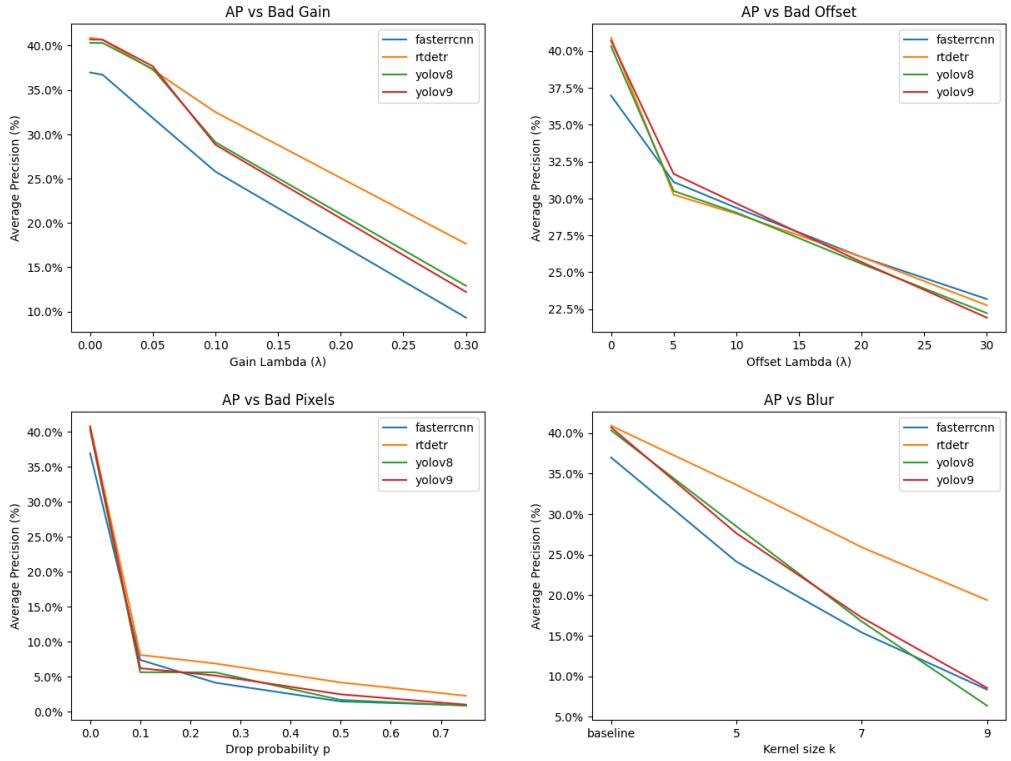

Figure 5. Figure shows AP scores for each of the calibration artifacts across varying artifact intensities.

Across most artifacts and artifact intensities, RT-DETR was the most robust model examined in this experiment. It maintained the highest AP across all bad pixel and blur intensities and was roughly equal or better across all bad gains. Looking at the blur artifact specifically, RT-DETR outperformed all other models in every measured statistic.

The most significant difference in model robustness came when looking at large vessels. Looking at Figure 6, RT-DETR significantly outperformed the YOLO and Faster-RCNN models in AP^l regardless of artifact or artifact intensity. Whereas RT-DETR performed similarly to the other models when looking at overall AP and AP^s, the performance gap on large vessels generally widened between RT-DETR and those models as artifact intensity increased. RT-DETR's AP^l with blur $k = 9$ almost outperformed the baseline Faster-RCNN in the same metric (44.7% vs 46.6%). This was only a 10.9% drop from RT-DETR's baseline AP^l compared to a 24.5% drop for Faster-RCNN with blur $k = 9$.

The original DETR paper[29] noted its architecture's significantly better performance on large objects due to non-local transformer computations. This experiment not only supports this fact but also adds that its large object performance is significantly more robust to image perturbations than comparable CNN-based architectures.

Both YOLO models performed very similarly across image artifacts. YOLOv9 outperformed YOLOv8 in AP for smaller bad offset intensities $\lambda \in \{5, 10, 15\}$, but the opposite is true for bad offset $\lambda = 30$. Looking at AP^l, YOLOv9 outperformed YOLOv8 at each intensity for bad offset and bad pixels artifacts. These differences were very small either way, however. The relative performance between each YOLO model varied significantly between calibration artifact and intensity.

Across most artifacts, Faster-RCNN was the worst performing and least robust. It had the lowest AP for every bad gain and experienced the largest drops in AP for bad gain with $\lambda \in \{0.01, 0.05\}$. It was the worst performing in AP^l across all Blur and Bad Offset intensities. Faster-RCNN was especially bad relative to the other models when detecting small vessels. Interestingly, across all levels of bad offset, it did outperform both YOLOs and RT-DETR for AP^{50}. Faster-RCNN also had the highest AP across all models at bad offset $\lambda = 30$.

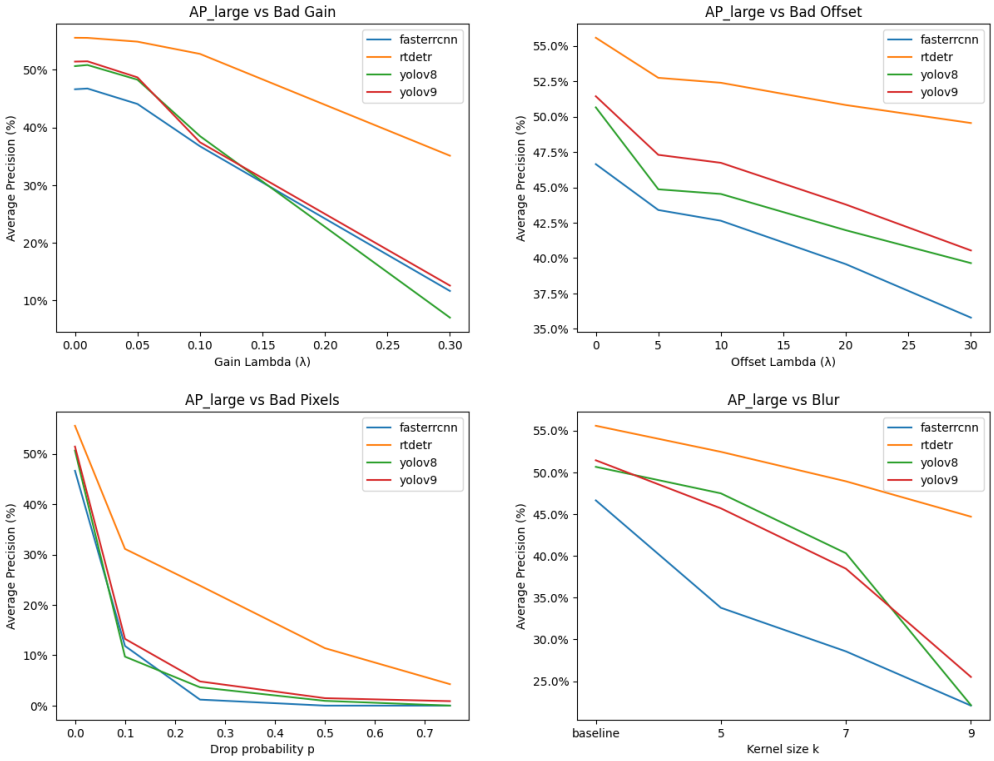

Figure 6. Figure shows AP^l scores for each of the calibration artifacts across varying artifact intensities.

Overall trends in AR across model/artifact are similar to the AP ones discussed above. Notably, RT-DETR had the highest AR across all artifact and artifact intensities. Refer to tables 6, 7, 8, 9 in the appendix for a complete summary of all evaluated metrics.

4. CONCLUSION

We compared how four SOTA object detection models fine-tuned to locate ships in commercial overhead imagery performed across images containing fifteen different imaging calibration artifacts. A test set of one thousand roughly 5000x5000 WV2 and WV3 images was examined. We were interested in extending prior work examining calibration artifacts on classifiers to more sophisticated detection models that also performed the task of localizing objects within an image. This work informs future ATR algorithm selection and computer vision system design as a result for maritime applications using overhead imagery.

Our results demonstrated that SOTA object detection model performance is substantially impacted by the introduction of image calibration artifacts. Model experienced up to a 40% decrease in AP values depending on the artifact introduced.

Our results show that the RT-DETR model outperformed YOLOv8, YOLOv9, and Faster-RCNN on the test data and was generally more robust to the examined image calibration artifacts. In particular, RT-DETR was much more robust to stronger perturbations when detecting large objects. Without implementing techniques to alter robustness, this model could be better suited for an application like tracking large shipping vessels. Yolov8 and Yolov9 are generally comparable in their robustness to calibration artifacts. Faster-RCNN is generally the worst model, although it outperforms the YOLO models in AP^{50} under more intense blurring and bad offset calibrations.

Future work should look into how these anticipated calibration artifacts impact model performance across models of varying size. The models examined in this work were very large and as a result more closely represent

those that would be used on ground stations after data downlink. With the proliferation of remote imaging constellations, there is a push for these ATR algorithms to physically run on-board satellite hardware thus necessitating lighter models with fewer parameters and energy costs. It is important to characterize differences in these models' robustness to inform model selection.

Additional work should look into techniques that improve the robustness of a particular model to these calibration artifacts, whether that be by applying random artifacts as data augmentation during training or by creating more specialized loss functions that reward the algorithm for inferencing based on features that are more semantically salient to the human eye. Depending on intensity, these calibration artifacts can be almost indiscernible to a person, and that gap between neural network and human inferencing needs to be bridged.

Ultimately, for proliferated imaging constellations to be successful in performing ATR for MDA applications, future work is needed to develop more robust methods to mitigate the effects of these common image artifacts.

ACKNOWLEDGMENTS

This research was made possible by the US Naval Research Laboratory Base Program. The author would like to acknowledge and thank their funding sponsors.

REFERENCES

[1] Glasner, J., "One giant step for imaging." Wired, 25 September 1999 https://www.wired.com/1999/09/one-giant-step-for-imaging/. (Accessed: 29 February 2024).

[2] Sweeting, M. N., "Modern small satellites-changing the economics of space," *Proceedings of IEEE* **106**(3), 343–361 (2018).

[3] Patschke, G., "Key trends shaping the future of space technology." Aerospace Manufacturing, 29 December 2023 https://www.aerospacemanufacturing.com/news/key-trends-shaping-the-future-of-space-technology-most-read-2023/. (Accessed: 29 February 2024).

[4] Sebastian, K. A., "Ucs satellite database." UCS, 1 May 2023 https://www.ucsusa.org/resources/satellite-database. (Accessed: 28 February 2024).

[5] Howard, K. and Ah, A. V., "Large constellations of satellites: Mitigating environmental and other effects," Tech. Rep. GAO-22-105166, United States Government Accountability Office, Washington, DC (September 2022).

[6] Alessandrini, A., Alvarez, M., Greidanus, H., Gammieri, V., Arguedas, V. F., Mazzarella, F., Santamaria, C., Stasolla, M., Tarchi, D., and Vespe, M., "Mining vessel tracking data for maritime domain applications," in [*2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)*], 361–367 (2016).

[7] Bannister, N. P. and Neyland, D. L., "Maritime domain awareness with commercially accessible electro-optical sensors in space," *International Journal of Remote Sensing* **36**(1), 211–243 (2015).

[8] Kocak, D. M. and Browning, P., "Real-time ais tracking from space expands opportunities for global ocean observing and maritime domain awareness," in [*OCEANS 2015-MTS/IEEE Washington*], 1–7, IEEE (2015).

[9] Sandiford, M., "When it comes to maritime domain awareness, navy looks to space for data." Federal News Network, 2 November 2023 https://federalnewsnetwork.com/navy/2023/11/when-it-comes-to-maritime-domain-awareness-the-navy-looks-to-space-for-data/. (Accessed: 29 February 2024).

[10] Ward, C. M., Harguess, J., and Hilton, C., "Ship classification from overhead imagery using synthetic data and domain adaptation," in [*OCEANS 2019 MTS/IEEE Charleston*], 1–5 (2019).

[11] Ophoff, T., Puttemans, S., Kalogirou, V., Robin, J.-P., and Goedemé, T., "Vehicle and vessel detection on satellite imagery: A comparative study on single-shot detectors," *Remote Sensing* **12**(7) (2020).

[12] Yao, Y., Jiang, Z., Zhang, H., Zhao, D., and Cai, B., "Ship detection in optical remote sensing images based on deep convolutional neural networks," *Journal of Applied Remote Sensing* **11**(4), 042611 (2017).

[13] Hu, B. and Miao, H., "An improved deep neural network for small-ship detection in sar imagery," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **17**, 2596–2609 (2024).

[14] Chung, Y., Haas, P. J., Upfal, E., and Kraska, T., "Unknown examples & machine learning model generalization," *arXiv preprint arXiv:1808.08294* (2018).

[15] Hawkins, D. M., "The problem of overfitting," *Journal of Chemical Information and Computer Science* **44**(1), 1–12 (2004). PMID: 14741005.

[16] Drenkow, N., Sani, N., Shpitser, I., and Unberath, M., "A systematic review of robustness in deep learning for computer vision: Mind the gap?," *arXiv preprint arXiv:2112.00639* (2021).

[17] Warner, J. G., Davidson, Q., Tietz, M., Scharpf, W., and Keene, C., "Characterizing cnn-based vessel detection algorithm sensititivy to optical sensor artifacts," in [*2023 International Conference on Applied Machine Learning (ICMLA)*], 756–763 (2023).

[18] Chen, W., Han, B., Yang, Z., and Gao, X., "Mssdet: Multi-scale ship-detection framework in optical remote-sensing images and new benchmark," *Remote Sensing* **14**(21) (2022).

[19] Anderson, N. T. and Marchisio, G. B., "Worldview-2 and the evolution of the digitalglobe remote sensing satellite constellation: introductory paper for the special session on worldview-2," in [*Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery XVIII*], **8390**, 166–180, SPIE (2012).

[20] Longbotham, N., Pacifici, F., Malitz, S., Baugh, W., and Camps-Valls, G., "Measuring the spatial and spectral performance of worldview-3," in [*Hyperspectral Imaging and Sounding of the Environment*], HW3B–2, Optica Publishing Group (2015).

[21] Tkachenko, M., Malyuk, M., Holmanyuk, A., and Liubimov, N., "Label Studio: Data labeling software," (2020-2022). Open source software available from https://github.com/heartexlabs/label-studio.

[22] Girshick, R., Donahue, J., Darrell, T., and Malik, J., "Rich feature hierarchies for accurate object detection and semantic segmentation," in [*Proceedings of the IEEE conference on computer vision and patter recognition*], 580–587 (2014).

[23] Girshick, R., "Fast r-cnn," in [*Proceedings of the IEEE international conference on computer vision*], 1440–1448 (2015).

[24] Ren, S., He, K., Girshick, R., and Sun, J., "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems* **38** (2015).

[25] Redmon, J., Divvala, S. K., Girshick, R. B., and Farhadi, A., "You only look once: Unified, real-time object detection," *CoRR* **abs/1506.02640** (2015).

[26] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C., [*SSD: Single Shot MultiBox Detector*], 21–37, Springer International Publishing (2016).

[27] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S., "Pytorch: An imperative style, high-performance deep learning library," in [*Advances in Neural Information Processing Systems 32*], 8024–8035, Curran Associates, Inc. (2019).

[28] Lecun, Y., Bengio, Y., and Hinton, G., "Deep learning," *nature* **521**(7553), 436–444 (2015).

[29] Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S., "End-to-end object detection with transformers," in [*European conference on computer vision*], 213–229, Springer (2020).

[30] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I., "Attention is all you need," *Advances in neural information processing systems* **30** (2017).

[31] Lv, W., Zhao, Y., Xu, S., Wei, J., Wang, G., Cui, C., Du, Y., Dang, Q., and Liu, Y., "Detrs beat yolos on real-time object detection," *arXiv preprint ArXiv:2304.08069* (2023).

[32] Jocher, G., Chaurasia, A., and Qiu, J., "Ultralytics yolov8." Ultralytics v8.0.0, 2023 https://github.com/ultralytics/ultralytics.

[33] Wang, C.-Y., Yeh, I.-H., and Liao, H.-Y. M., "Yolov9: Learning what you want to learn using programmable gradient information," *arXiv preprint arXiv: 2402.13616* (2024).

[34] "Nvidia rtx 6000 ada generation datasheet." NVIDIA, 2023 https://www.nvidia.com/content/dam/en-zz/Solutions/design-visualization/rtx-6000/proviz-print-rtx6000-datasheet-web-2504660.pdf. (Accessed: 7 March 2024).

[35] Dinguirard, M. and Slater, P. N., "Calibration of space-multispectral imaging sensors: A review," *Remote Sensing of Environment* **68**(3), 194–205 (1999).

[36] Slater, P. N., Biggar, S. F., Palmer, J. M., and Thome, K. J., "Unified approach to pre-and in-flight satellite-sensor absolute radiometric calibration," in [*Advanced and Next-Generation Satellites*], **2583**, 130–141, SPIE (1995).

[37] Krause, K. S., "Worldview-1 pre and post-launch radiometric calibration and early on-orbit characterization," in [*Earth Observing Systems XIII*], **7081**, 338–348, SPIE (2008).

[38] Kuester, M. A., Ochoa, M., Dayer, A., Levin, J., Aaron, D., Helder, D. L., Leigh, L., Czapla-Meyers, J., Anderson, N., Bader, B., et al., "Absolute radiometric calibration of the digitalglobe fleet and updates on the new worldview-3 sensor suite," in [*Report of JACIE Civil Commercial Imagery Evaluation Workshop of DigitalGlobe Inc.; DigitalGlobe Inc.: Westminster, CO, USA*], (2017).

[39] Holste, G. C. and Lomheim, T. S., [*CMOS/CCD Sensor and Camera Systems*], SPIE Press, second ed. (2011).

[40] "Correction of ccd image imperfections." Diagnostic Instruments Inc., 2004 https://www.spotimaging.com/downloads/public/pdf/ImageCorrections.pdf. (Accessed: 8 March 2024).

[41] Smith, R. M., Hale, D., and Wizinowich, P., "Bad pixel mapping," in [*High Energy, Optical, and Infrared Detectors for Astronomy VI*], **9154**, 372–387, SPIE (2014).

[42] Mulawa, D., "On-orbit geometric calibration of the orbview-3 high resolution imaging satellite," *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci* **35**, 1–6 (2004).

[43] Akyon, F. C., Onur Altinuc, S., and Temizel, A., "Slicing aided hyper inference and fine-tuning for small object detection," in [*2022 IEEE International Conference on Image Processing (ICIP)*], 966–970 (2022).

[44] "Detection evaluation." COCO, 2020 https://cocodataset.org/#detection-eval. (Accessed: 1 March 2024).

[45] Lin, T.-Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C. L., and Dollár, P., "Microsoft coco: Common objects in context," in [*Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2024, Proceedings, Part V 13*], 740–755, Springer (2014).

[46] "Mean-average-precision." LightningAI v1.3.1, 2024 https://lightning.ai/docs/torchmetrics/stable/detection/mean_average_precision.html. (Accessed: 5 March 2024).

[47] Padilla, R., Passos, W. L., Dias, T. L. B., Netto, S. L., and da Silva, E. A. B., "A comparative analysis of object detection metrics with a companion open-source toolkit," *Electronics* **10**(3) (2021).

[48] Hosang, J., Benenson, R., Dollar, P., and Schiele, B., "What makes for effective detection proposals?," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **38**, 814–830 (Apr. 2016).

APPENDIX A. TRAINING CONFIGURATION

Table 5. The table shows training parameters for each of the models used in the experiment. An effort to keep parameters consistent across models was made except for specific cases. Notably, the RT-DETR model wouldn't converge with an initial training rate of 0.01 and encountered GPU memory issues training at a batch size of 64. Loss gains are omitted for Faster-RCNN as it used a separate training loss function.

hyper parameter	Faster-RCNN	RT-DETR	YOLOv8	YOLOv9
epochs	300	300	300	300
batch	64	32	64	64
optimizer	SGD	SGD	SGD	SGD
initial learning rate	0.01	0.001	0.01	0.01
final learning rate	0.0001	0.00001	0.0001	0.0001
learning rate decay	linear	linear	linear	linear
momentum	0.937	0.937	0.937	0.937
weight decay	0.0005	0.0005	0.0005	0.0005
warm-up epochs	3	3	3	3
warm-up momentum	0.8	0.8	0.8	0.8
warm-up bias	0.1	0.1	0.1	0.1
HSV hue augmentation	0.0015	0.0015	0.0015	0.0015
HSV saturation augmentation	0.7	0.7	0.7	0.7
HSV value augmentation	0.4	0.4	0.4	0.4
perspective scale augmentation	0.5	0.5	0.5	0.5
perspective translate augmentation	0.1	0.1	0.1	0.1
flip up-down augmentation	0.5	0.5	0.5	0.5
flip left-right augmentation	0.5	0.5	0.5	0.5
box loss gain	—	7.5	7.5	7.5
class loss gain	—	0.5	0.5	0.5
DFL loss gain	—	1.5	1.5	1.5

APPENDIX B. EXPERIMENTAL DATA

Table 6. COCO evaluation metrics for Faster-RCNN by calibration artifact.

	AP	AP^{50}	AP^{75}	AP^s	AP^m	AP^l	AR	AR^s	AR^m	AR^l
Baseline	37.0	57.5	39.4	33.5	46.2	46.6	50.9	46.9	57.9	62.3
Gain(0.01)	36.7	56.9	39.0	32.8	45.8	46.8	50.6	46.6	57.9	61.8
Gain(0.05)	31.8	48.8	33.7	26.3	41.8	44.1	47.6	44.0	54.7	57.0
Gain(0.1)	25.8	39.3	27.5	20.3	36.1	36.8	43.9	40.8	51.3	49.3
Gain(0.3)	9.3	14.7	9.6	7.0	14.9	11.7	22.2	19.8	30.8	21.5
Offset(5)	31.1	48.2	32.9	26.5	41.7	43.4	44.5	39.4	54.8	56.7
Offset(10)	29.4	45.8	30.8	23.9	40.8	42.6	42.6	37.2	53.8	55.1
Offset(20)	26.0	41.0	27.4	20.0	38.0	39.6	39.3	33.7	50.9	51.8
Offset(30)	23.2	36.7	24.2	17.1	35.3	35.8	36.1	30.5	48.8	47.8
Pixels(0.10)	7.4	11.9	7.6	6.6	11.1	11.8	20.8	16.9	30.4	26.9
Pixels(0.25)	4.1	7.6	3.7	3.4	7.7	1.2	5.7	5.3	9.3	0.9
Pixels(0.5)	1.4	2.3	1.5	1.2	2.9	0.0	1.3	1.1	2.7	0.0
Pixels(0.75)	0.9	1.0	1.0	0.6	0.9	0.0	0.0	0.0	0.1	0.0
Blur(5)	24.1	40.9	24.7	22.8	37.6	33.8	37.1	31.4	47.6	52.8
Blur(7)	15.4	28.1	14.6	13.0	27.8	28.6	26.4	19.8	37.5	45.8
Blur(9)	8.4	16.0	7.4	5.2	16.8	22.1	15.6	9.3	24.8	37.4

Table 7. COCO evaluation metrics for RT-DETR by calibration artifact.

	AP	AP^{50}	AP^{75}	AP^s	AP^m	AP^l	AR	AR^s	AR^m	AR^l
Baseline	40.9	61.3	43.1	39.0	54.8	55.6	55.0	49.4	64.6	71.2
Gain(0.01)	40.7	60.9	42.9	38.6	54.7	55.6	55.0	49.5	64.5	71.1
Gain(0.05)	37.3	55.6	39.3	33.2	51.8	54.9	53.6	48.3	62.8	69.3
Gain(0.1)	32.5	48.5	34.3	26.3	46.7	52.8	49.8	44.3	59.3	66.0
Gain(0.3)	17.7	26.8	18.5	13.5	27.7	35.1	34.1	28.9	43.9	48.0
Offset(5)	30.3	45.4	31.3	30.0	44.1	52.7	50.1	43.4	62.0	69.1
Offset(10)	29.0	43.6	30.1	27.7	43.0	52.4	48.7	41.8	61.4	68.2
Offset(20)	26.0	39.3	27.2	24.7	39.7	50.8	46.6	39.5	59.2	66.7
Offset(30)	22.8	34.5	23.7	20.9	34.9	49.5	43.7	36.6	55.9	65.2
Pixels(0.10)	8.1	12.2	8.5	10.6	13.9	31.1	33.4	26.7	43.8	55.0
Pixels(0.25)	6.9	10.6	6.8	5.8	10.9	23.8	20.1	13.7	29.7	41.3
Pixels(0.5)	4.2	6.9	4.0	3.0	7.1	11.4	8.5	4.8	15.2	18.9
Pixels(0.75)	2.3	4.0	1.9	1.3	4.4	4.3	3.2	1.5	7.3	6.3
Blur(5)	33.6	53.8	34.9	29.1	50.7	52.5	43.7	35.1	58.3	69.2
Blur(7)	25.9	44.8	24.9	19.4	43.5	48.9	34.3	24.2	50.5	66.0
Blur(9)	19.4	35.5	17.7	12.1	35.1	44.7	26.1	15.8	41.1	61.7

Table 8. COCO evaluation metrics for YOLOv8 by calibration artifact.

	AP	AP^{50}	AP^{75}	AP^s	AP^m	AP^l	AR	AR^s	AR^m	AR^l
Baseline	40.3	58.1	43.7	39.4	54.0	50.7	53.1	47.7	63.6	67.0
Gain(0.01)	40.3	57.5	43.7	39.2	54.0	50.8	53.1	47.6	63.7	67.1
Gain(0.05)	37.4	53.1	40.2	35.4	50.2	48.3	51.1	46.3	60.5	63.0
Gain(0.1)	29.1	41.1	31.6	25.1	41.3	38.5	46.0	42.8	54.1	50.8
Gain(0.3)	12.9	18.7	14.2	11.4	19.5	7.0	24.6	25.8	28.8	9.1
Offset(5)	30.5	43.9	32.7	32.1	44.4	44.9	49.0	42.2	61.9	66.9
Offset(10)	29.0	41.7	31.1	30.0	43.0	44.5	46.8	39.8	60.0	65.6
Offset(20)	25.6	37.0	27.5	27.1	38.3	42.0	44.1	37.1	56.8	64.3
Offset(30)	22.2	32.1	23.6	24.0	34.0	39.6	41.1	33.7	53.8	62.9
Pixels(0.10)	5.6	8.3	5.9	12.3	7.9	9.7	23.3	19.3	32.9	30.4
Pixels(0.25)	5.6	8.5	6.0	5.3	9.2	3.6	8.3	7.5	12.8	4.7
Pixels(0.5)	1.7	1.9	1.8	1.5	3.6	1.0	1.4	0.9	3.4	0.4
Pixels(0.75)	0.8	1.0	1.0	0.7	0.9	0.0	0.1	0.0	0.2	0.0
Blur(5)	28.5	43.8	30.5	24.1	45.2	47.5	37.3	28.6	52.1	63.2
Blur(7)	16.8	26.9	17.0	11.4	30.4	40.3	23.0	14.2	35.1	54.7
Blur(9)	6.4	10.3	6.3	3.7	12.0	22.1	10.4	5.0	14.4	35.9

Table 9. COCO evaluation metrics for YOLOv9 by calibration artifact.

	AP	AP^{50}	AP^{75}	AP^s	AP^m	AP^l	AR	AR^s	AR^m	AR^l
Baseline	40.7	58.4	44.0	39.4	53.6	51.4	52.5	47.2	62.6	66.1
Gain(0.01)	40.7	58.4	44.1	39.3	53.3	51.5	52.3	47.1	62.4	66.2
Gain(0.05)	37.7	53.6	40.5	35.6	50.0	48.7	50.2	45.4	59.7	62.4
Gain(0.1)	28.8	40.4	31.3	25.4	40.8	37.4	45.0	41.9	53.5	49.0
Gain(0.3)	12.2	17.5	13.1	10.5	17.4	12.6	26.0	25.4	32.8	17.4
Offset(5)	31.7	45.6	34.1	31.5	46.3	47.3	46.8	39.9	60.3	64.6
Offset(10)	29.7	42.6	31.6	29.5	44.0	46.7	44.8	37.5	58.9	64.1
Offset(20)	25.7	36.8	27.6	25.6	38.7	43.8	41.3	33.4	56.0	62.8
Offset(30)	21.9	31.3	23.5	21.9	34.6	40.5	38.0	29.7	53.0	61.3
Pixels(0.10)	6.2	8.8	6.6	11.0	7.4	13.3	27.5	22.4	38.9	37.6
Pixels(0.25)	5.2	7.9	5.5	5.3	8.0	4.8	10.0	8.2	16.9	8.7
Pixels(0.5)	2.5	3.6	2.4	1.9	5.0	1.5	2.5	1.7	5.6	1.4
Pixels(0.75)	1.0	1.0	1.0	0.6	2.2	0.9	0.5	0.2	1.7	0.2
Blur(5)	27.6	42.6	29.6	24.3	44.9	45.7	37.3	29.0	51.8	61.2
Blur(7)	17.3	27.8	17.6	12.0	31.4	38.5	23.7	15.1	37.2	52.4
Blur(9)	8.6	13.6	8.6	5.0	14.6	25.5	12.0	6.3	17.8	36.2