Semantic Segmentation of Low Earth Orbit Satellites using Convolutional Neural Networks

Julia Yang
The Boeing Company
550 Lipoa Pkwy
Kihei, HI 96753
julia.l.yang@boeing.com

Jacob Lucas
The Boeing Company
550 Lipoa Pkwy
Kihei, HI 96753
jacob.a.lucas@boeing.com

Trent Kyono
The Boeing Company
550 Lipoa Pkwy
Kihei, HI 96753
trent.m.kyono@boeing.com

Michael Abercrombie
The Boeing Company
550 Lipoa Pkwy
Kihei, HI 96753
michael.d.abercrombie@boeing.com

Andrew Vanden Berg
Air Force Research Laboratory
550 Lipoa Pkwy
Kihei, HI 96753
andrew.vanden_berg.1@us.af.mil

Justin Fletcher
United States Space Force Space Systems Command
550 Lipoa Pkwy
Kihei, HI 96753
justin.fletcher.14.ctr@us.af.mil

2022 IEEE Aerospace Conference (AERO) | 978-1-6654-3760-8/22/$31.00 ©2022 IEEE | DOI: 10.1109/AERO53065.2022.9843511

***Abstract*—Ground-based imaging of objects in Low Earth Orbit (LEO) is complicated by atmospheric turbulence, which make it difficult to identify key features or components on the object of interest. Many automated image reconstruction techniques are in use, but expert labor is needed to subjectively discern and identify truth features on a partially reconstructed image. In this paper, we present a deep learning approach for semantic segmentation of ground-based images of LEO objects. We investigate the performance under various atmospheric turbulence strengths in terms of the Fried parameter (r_0) and show the viability of this method.**

TABLE OF CONTENTS

1. INTRODUCTION

Ground-based imaging of Low Earth Orbit (LEO) satellites is essential to comprehensive space domain awareness (SDA). However, image quality can be severely reduced by atmospheric turbulence. In such conditions, tasks such as labeling the components of a satellite can be difficult, and is susceptible to mistake when done by a human. One such task, segmenting images into their individual components, has seen revolutionary improvement through machine learning techniques [1], [2], [3], [4], [5], [6].

In this paper, we explore the feasibility of applying convolutional neural networks for semantic segmentation of ground-based images of LEO satellites. We first create datasets of multiple satellites consisting of renders of the satellites both with and without atmospheric turbulence. Then, we design and build the inference models used for the experiments. After the datasets and model are complete, we establish a baseline of performance by training the model on a simple dataset containing images of a single satellite absent of any atmospheric turbulence. Then, we train the model on single levels of turbulence, followed by training the model with multiple turbulence levels. Finally, we increase the complexity of the task by training a single model to segment images of a variety of satellites.

The key contributions of this paper are as follows. First, we propose a neural network approach for semantic segmentation of ground-based images of LEO satellites. Second, to the best of our knowledge, our approach is the first to show that this approach can be applied to these images subject to different atmospheric turbulence conditions. These contributions can advance practical SDA, remove the human-in-the-loop required for segmentation of images of satellites, and provides a method for image understanding that had not previously been used.

We present past research in the field of semantic segmentation, and formalize the problem and approach. Then, we describe our experiments including data generation, metrics and models, methods and results. Finally, we discuss some future work and concluding remarks.

2. RELATED WORKS

Convolutional neural networks (CNNs) are one of the most widely accepted methods in computer vision and are often used for image classification, segmentation, and detection tasks. With the recent success of these algorithms in image understanding tasks, such as ImageNet in [7], [8], those in [9] and [10], etc., coupled with the advancements in deep learning libraries, GPU hardware acceleration, and data availability, these methods have received heightened interest in the SDA. These recent advancements in image processing with CNNs inform our application of a neural network for semantic segmentation of satellites.

Our work draws on several related works demonstrating

significant success of using CNNs for semantic segmentation [2], [3], [5], [6], [11]. There has also been significant improvements in performance for interpreting noisy and degraded images using CNNs [12], [13], [14]. Specifically, stacked denoising autoencoders were presented as early as 2010 by [15] which has been iterated on to develop new architectures, such as U-Net, yielding state-of-the-art in many image segmentation tasks [6]. We draw motivation from this architecture and recent success and apply it to semantic segmentation of pristine and noisy images of LEO satellites.

3. PROBLEM FORMULATION

Let X be a set of astronomical images corresponding to a dataset of n images, where one input image is denoted as $x_i \in X$. Let Y be the dataset's truth segmentation set whereby $y_i \in Y$ we denote the ith input image's truth segmentation map. The input image x_i has size $h \times w \times c$ where each dimension represents the height in pixels, width in pixels, and number of channels. The truth segmentation map y_i has size $h \times w \times l$ where l represents the number of classes. In other words, y_i consists of a stack of l masks where each mask is $h \times w$ and each mask represents a class.

Our primary design goal is to train a semantic segmentation network $f : X \rightarrow Y$, which takes as input x_i and provides a segmentation inference of y_i. We approach this task using U-Net which outputs for each input image x_i an inference p_i which also has size $h \times w \times l$. However, here, each element of p_i represents an inferred probability that the pixel belongs to a certain class.

4. EXPERIMENTS

This section presents the data used for this work, establishes a metric for measuring performance, and our experimental settings (training architecture and regimes), and our experimental results.

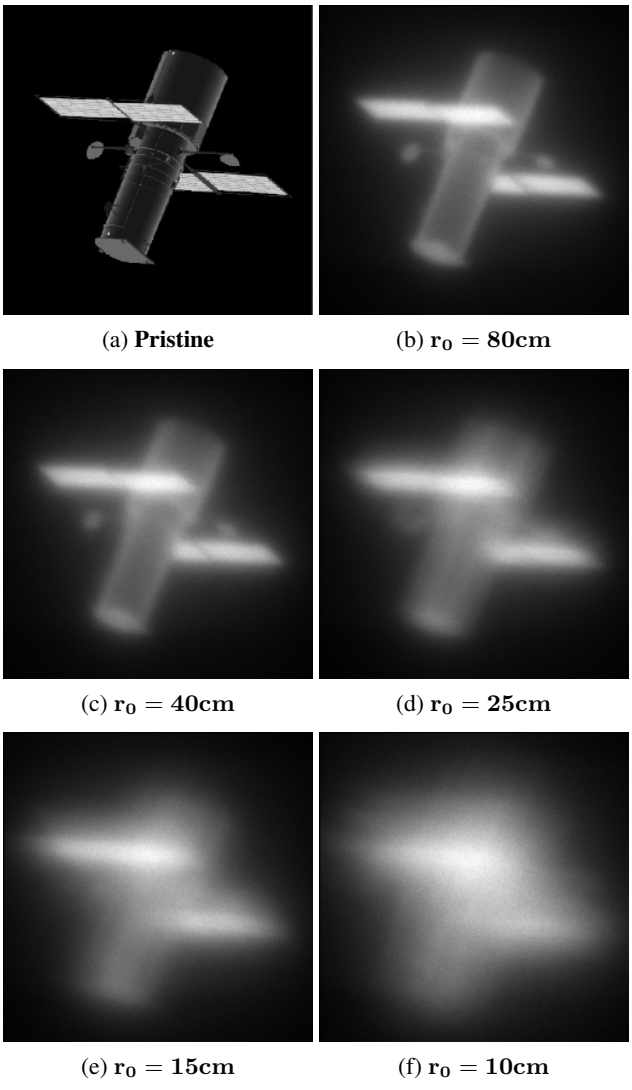
Dataset

This study uses supervised learning and requires an extensive set of (image, truth) pairs for model training. The dataset used consists of multiple satellites, each rendered across a complete range of discrete poses as if viewed from the 3.6 m AEOS (Advanced Electro-Optical System) telescope at the summit of Haleakala.

The satellites used were six real satellites: Cosmic Background Explorer (COBE), Hubble Space Telescope (HST), MightySat, Technology for Autonomous Operational Survivability (TAOS), Television Infrared Observation Satellites (TIROS), and Wide Field Infrared Explorer (WIRE). In addition to these six real satellites, we also used a simplified representative satellite consisting of a cubic bus, two solar panels, and an antenna. This simplified satellite will be referred to as Boxsat.

Renders of these satellites as well as their truth class labels were produced using COAST/FIST with image properties reflecting that of diffraction limited images from the AEOS telescope. The truth class labels consisted of six different classes: bus, solar panels, thrusters, payloads, antenna, and background.

Imaging at the AEOS telescope only occurs in conditions without cloud cover, but in order to better represent real-world conditions of atmospheric turbulence, we also generated datasets with realistic degradation applied to the images. The images were degraded by SILO-G [16] to 5 different turbulence levels as represented in Table 1. The turbulence levels were characterized by the Freid parameter (r_0) [17]. We chose r_0 values that corresponded with poor, average, good, exceptional, and typical adaptive optics seeing conditions. Examples of renders in each of these turbulence levels can be seen in Figure 1.



(a) **Pristine** (b) $\mathbf{r_0 = 80cm}$

(c) $\mathbf{r_0 = 40cm}$ (d) $\mathbf{r_0 = 25cm}$

(e) $\mathbf{r_0 = 15cm}$ (f) $\mathbf{r_0 = 10cm}$

Figure 1: **Examples of renders of Hubble at the various turbulence levels**

Table 1: **Representative seeing conditions from Haleakala for AEOS Telescope.**

Seeing Condition	r_0 (cm)
Poor	10
Average	15
Good	25
Exceptional	40
Typical AO	80

SILO-G realistically degrades the images by convolving the

renders and pre-generated point spread functions (PSFs) and adding in camera effects of transmission losses, shot noise, quantum efficiency (QE), and read noise. The parameters used to generate these degraded images are shown in Table 2. The degraded datasets will be referred to by their r_0 values, and the dataset without any degradation will be referred to as pristine. We are restricted from sharing these datasets and their corresponding truth labels, so we will be sharing inferences only.

Table 2: **SILO-G parameters used to generate degraded images**

Parameter	Value	Unit
Image Size	256x256	pixels
IFOV	100	nrad
PSF Generation Rate	800	Hz
Frame Rate	40	Hz
Waveband	Bessel I	-
Turbulence Profile	MK50P	-
Transmission Noise	0.7	-
Quantum Efficiency	0.9	-
Read Noise	4	ADU
Fried Parameter r_0	10, 15, 25, 40, 80	centimeters

The best performing inference for each satellite at each turbulence level can be seen in Figure 3.

Metrics

For this study we measure performance using three metrics: the F_1 score, intersection over union (IoU), and pixel accuracy.

The F_1 score, also known as the Sørensen–Dice coefficient, is defined by Equation 1. The IoU, also referred to as the Jaccard Index, is defined by Equation 2 where y_i and p_i represent the true and predicted segmentation as described in Section 3. Both F_1 score and IoU is calculated for each class for each batch of images and then averaged across the entire test set. An ϵ is used to ensure F_1 score and IoU are 1.00 when measuring performance on classes that are not represented in the batch.

$$F_1 = \frac{2(precision * recall + \epsilon)}{precision + recall + \epsilon} \tag{1}$$

$$IoU = \frac{p_i * y_i + \epsilon}{p_i + y_i - p_i * y_i + \epsilon} \tag{2}$$

A visual representation of the three metrics is shown in Figure 2. The metrics reported in this paper are from the epoch with the lowest loss unless otherwise stated.

Architecture

In this work, we use a U-Net network based on previous success of U-Net for semantic segmentation [2]. We used both a depth 3 U-Net depicted in Figure 4a and a depth 5 U-Net depicted in Figure 4b.

All convolutional layers of the depth 3 U-Net used a ReLU activation function except the last. We initialized each kernel to Glorot Uniform [18]. The depth 5 U-Net also used ReLU

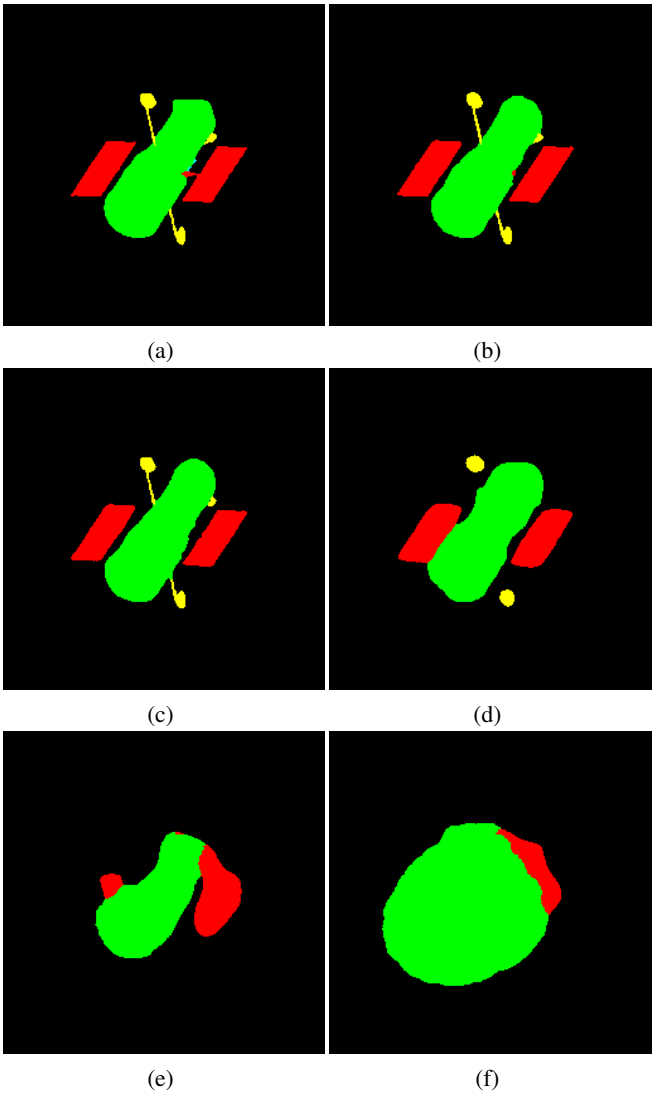


Figure 2: **Examples of F_1 scores, IoU, and pixel accuracy for inferences of Hubble. Green represents satellite bus, red represents solar panels, yellow represents antenna, and black represents the background. (a) $\mathbf{F_1 = 0.85}$, $\mathbf{IoU = 0.80}$, $\mathbf{PA = 1.00}$, (b) $\mathbf{F_1 = 0.75}$, $\mathbf{IoU = 0.68}$, $\mathbf{PA = 0.99}$, (c) $\mathbf{F_1 = 0.65}$, $\mathbf{IoU = 0.58}$, $\mathbf{PA = 0.99}$, (d) $\mathbf{F_1 = 0.55}$, $\mathbf{IoU = 0.48}$, $\mathbf{PA = 0.98}$, (e) $\mathbf{F_1 = 0.45}$, $\mathbf{IoU = 0.37}$, $\mathbf{PA = 0.95}$, (f) $\mathbf{F_1 = 0.35}$, $\mathbf{IoU = 0.27}$, $\mathbf{PA = 0.86}$**

for all activation layers except the last. The last layer used softmax. Additionally, each kernel was initialized to He Normal [19].

For input image augmentation to both U-Nets, we applied random augmentation to each training image with the following specification: horizontal flips, crop to between 100% to $\frac{1}{3}$ the original width, crop to between 100% to $\frac{1}{3}$ the original height, and resized to the original image dimensions using Bilinear interpolation. Additionally, from the albumentations library [20], we applied randomly coarse dropout, multiplicative noise, image color inversion, hue saturation, additive Gaussian noise, or Gaussian Noise. Also from the albumentations library, we either applied contrast limited adaptive histogram equalization to the input image, sharpened

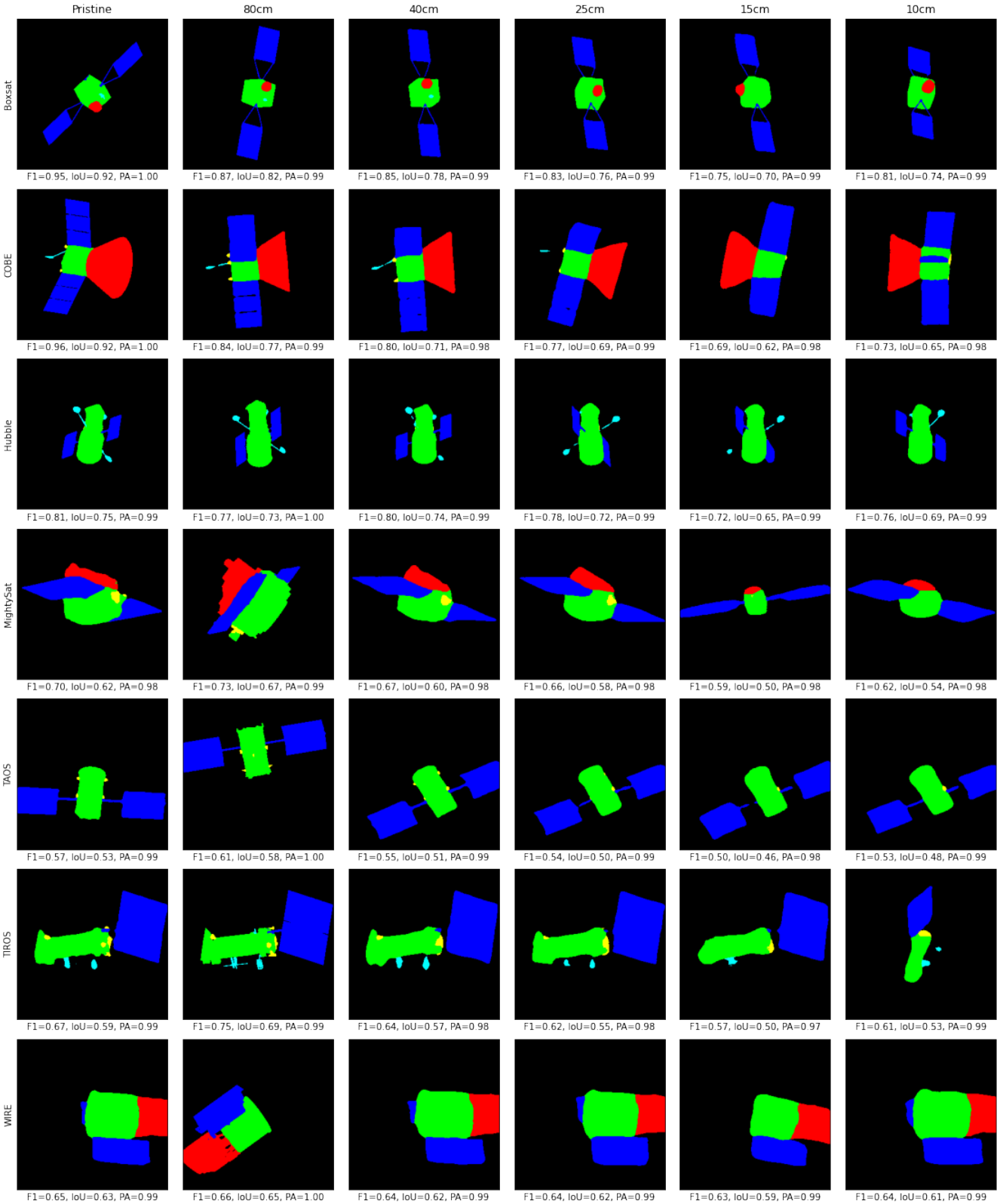
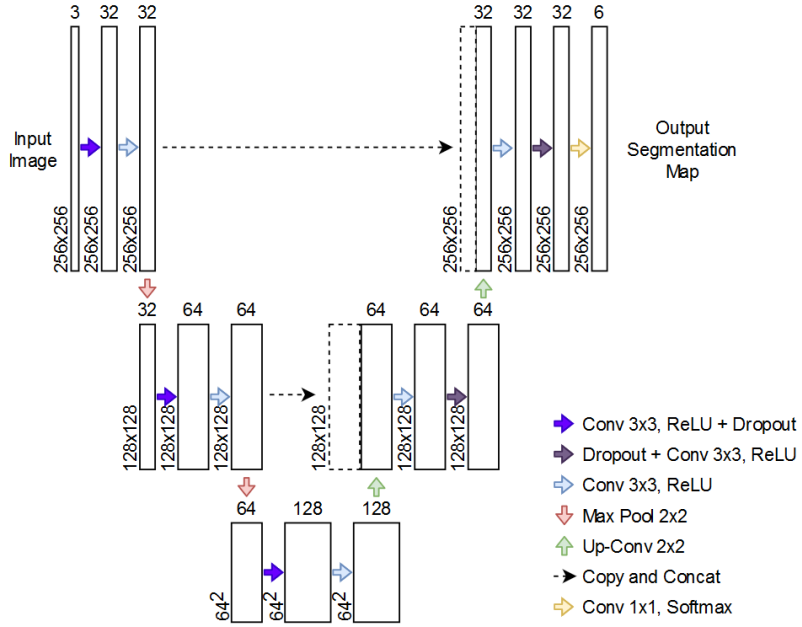
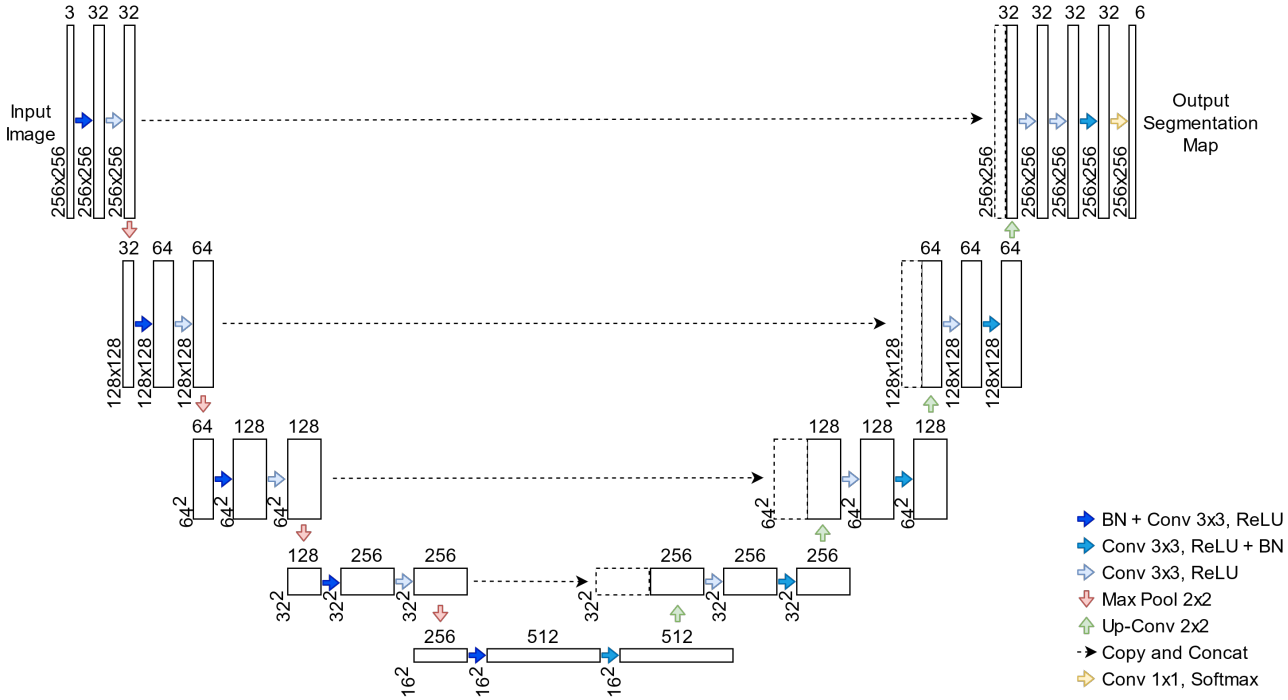


Figure 3: **The 7 different satellites used as well as their best inference at each turbulence level. Green represents satellite bus, blue represents solar panels, cyan represents antenna, red represents payload, yellow represents thrusters, and black represents the background.**



(a) **Depth 3 U-Net used to segment pristine images**



(b) **Depth 5 U-Net used to segment turbulent images**

Figure 4: **U-Net Structures**

the input image and overlaid the result with the original image, embossed the input image and overlaid the result with the original image, randomly changed brightness and contrast, or equalized the image histogram. Each training scenario used a 70% training, 20% validation, and 10% test split. We used the Adam optimizer with learning rate set to $1e^{-3}$ [21]. Categorical crossentropy was used as the loss function. The CNN was trained for 200 epochs, and the best validation loss was saved for inference and prediction. Image augmentation was not used during inference.

Reproducibility

For reproduction, all neural networks were trained using Python 3 and TensorFlow 2.1. Operating system and hardware specifications include Ubuntu 18.04.5 LTS on an NVidia DGX Server with eight Tesla V100 GPUs with 32 GB of memory on each card. Because the goal of this work was a feasibility investigation, we did not tune or search for optimal hyperparameters.

Training Method

To test the feasibility of semantic segmentation of these images, we ran a series of experiments by training models to complete tasks starting from the simplest Scenario A, and most complex Scenario E described below.

Training Scenario A: Single Satellite, No Turbulence—We first established a baseline of performance of the model by training the network to segment images of a single satellite in the absence of turbulence. The training set consisted of images of Boxsat, chosen for its simplicity (i.e. clearly defined edges and features) using the depth 3 U-Net depicted in Figure 4a. We then measured the performance of the model when presented with images at the different turbulence levels.

Scenario B: Single Satellite, Single Turbulence Level—For the next level of complexity, we introduced single levels of turbulence during training and validation. Even with the advances in denoising of images, post-processing of telescope images are rarely if ever pristine. For semantic segmentation to be applied to real images, it is necessary to evaluate the performance of such models through turbulence. As such, we trained a model on Boxsat for each turbulence level for a total of 6 models. The U-Net with depth 3 would not converge when presented with turbulence, so we increased depth of the U-Net to the U-Net with depth 5 shown in Figure 4b.

Scenario C: Single Satellite, Multi Turbulence Level—Then, to eliminate this need of training a model for each turbulence level, we trained a model at the next level of complexity: one model trained on all turbulence levels of Boxsat. Additionally, we tested the ability of the model to segment images with turbulence levels it had not previously encountered by training models on all but one turbulence level and testing with every turbulence level.

Scenario D: Reproducibility with Hubble—To ensure that the results from Scenarios A-C could be replicated for any satellite, we trained models for these three Scenarios using Hubble.

Scenario E: Generalizing Poses—To make sure the model was learning to semantically segment the images rather than mapping to the most similar image from the training set, we trained a variation of the models for Scenarios A-C with the following change to the datasets for Hubble. For each model, we chose a *target test image* and identified the 200 most similar input images (i.e. the images with the 200 highest F_1 scores when comparing their truth labels with the target image's truth labels). These images were then held out while the remainder of the dataset was split for training, validation, and test.

Scenario F: Multiple Satellites—Finally we attempted to create a model which can generalize across satellites. We first trained the depth 5 U-Net with renders in the absence of turbulence for all satellites except Hubble which was excluded as a test set. We held out all Hubble renders to use as part of the test set to measure the model's ability to learn the features rather than the satellite - in other words, to measure its ability to segment satellites the model had not encountered before in training. We then trained models for each of the turbulence levels using the same method as in Scenario B, but combining the datasets of all satellites excluding Hubble.

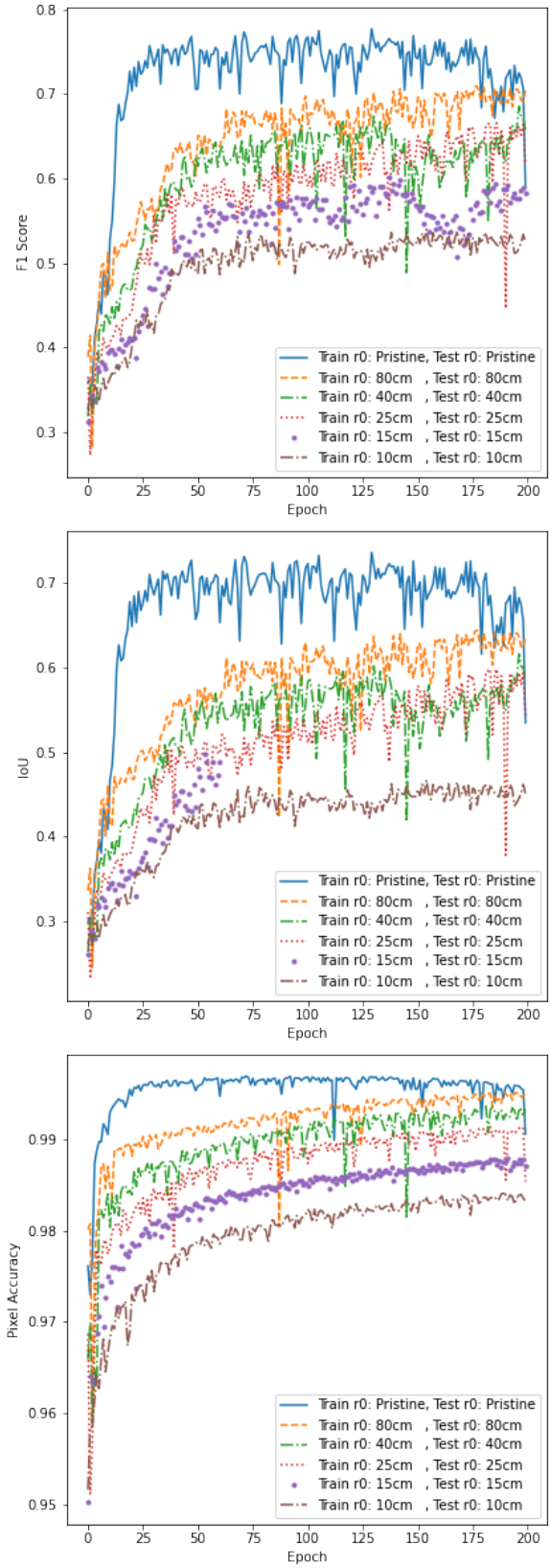


Figure 5: **Comparison of U-Net performance at different turbulence levels when trained and tested on a single turbulence level of Boxsat**

Results and Analysis

We report the results and analysis from the scenarios described in Section 4:

Scenario A: Single Satellite, No Turbulence—When measuring the baseline, the U-Net with depth 3 semantically segments pristine images of the test set at an F_1 score of 0.77, IoU of 0.73, and pixel accuracy of 1.00. When this baseline model which is only trained on pristine images is used to semantically segment images with turbulence, it consistently performed poorly with a mean F_1 score of 0.30, IoU of 0.26, and pixel accuracy of 0.71. These results summarized in the first column of Table 3 give us a point of comparison for the remaining scenarios.

Scenario B: Single Satellite, Single Turbulence Level—When attempting to train the model at the individual turbulence levels for Boxsat, the U-Net with depth 3 would not converge when presented with turbulence, so we increased depth of the U-Net to the U-Net with depth 5 shown in Figure 4b. The depth 5 U-Net converged and segmented through the noise. The performance of the resulting models when tested on images with the same turbulence level as that used in training can be seen in Figure 5. As expected, the performances of the models decrease with turbulence. However, the difference between the performance for segmenting turbulent images with the models trained in Scenario B when compared to Scenario A is improved by mean F_1 score of 0.31, IoU of 0.28, and pixel accuracy of 0.28.

To further test the generalization of the models trained on each r_0 value, we measured the performance of each model at the 6 different turbulence levels of Boxsat. The performance of the models as shown in Table 3 generally decreases as the difference in r_0 value for the train and test sets increases. For models trained and tested *on the same turbulence level*, performance clearly worsened as turbulence levels increased; as evidenced in the results for the pristine (F_1 = 0.77, IoU = 0.73, PA = 1.00) and 10cm (F_1 = 0.53, IoU = 0.46, PA = 0.98) cases. However, this performance was better than that of models trained and tested on *different turbulence levels*, where the model trained to infer using pristine images received an F_1 score of 0.30, IoU of 0.27, and pixel accuracy of 0.71 when tested on 10cm images. There is one anomaly to this generalization: of the models that were tested on $r_0 = 40cm$, the model that was trained on $r_0 = 25cm$ marginally outperforms the one trained on $r_0 = 40cm$. We posit that this anomaly is caused because the $r_0 = 40cm$ test set renders do not share any poses with the $r_0 = 40cm$ training set renders whereas they do share poses with the $r_0 = 25cm$ training set. Additionally, these are adjacent degradation levels, meaning that the difference between them is relatively small. However, based on the overall performance of all the models trained on single turbulence levels, the models overfit to its training turbulence level, and thus in order to have optimal performance at a specific r_0 when only training on one r_0, a model trained for that specific r_0 is needed.

Scenario C: Single Satellite, Multi Turbulence Level—The performance of the Scenario C model trained with all turbulence levels when tested at each turbulence level can be seen Figure 6. Additionally, as shown in Table 4, the model trained on all turbulence levels consistently performs better than the model trained at single turbulence levels when segmenting images from every turbulence level. This suggests that when testing on a turbulent image, the model leverages features from images of other turbulence levels to learn the features of the test image's turbulence level. Furthermore, when

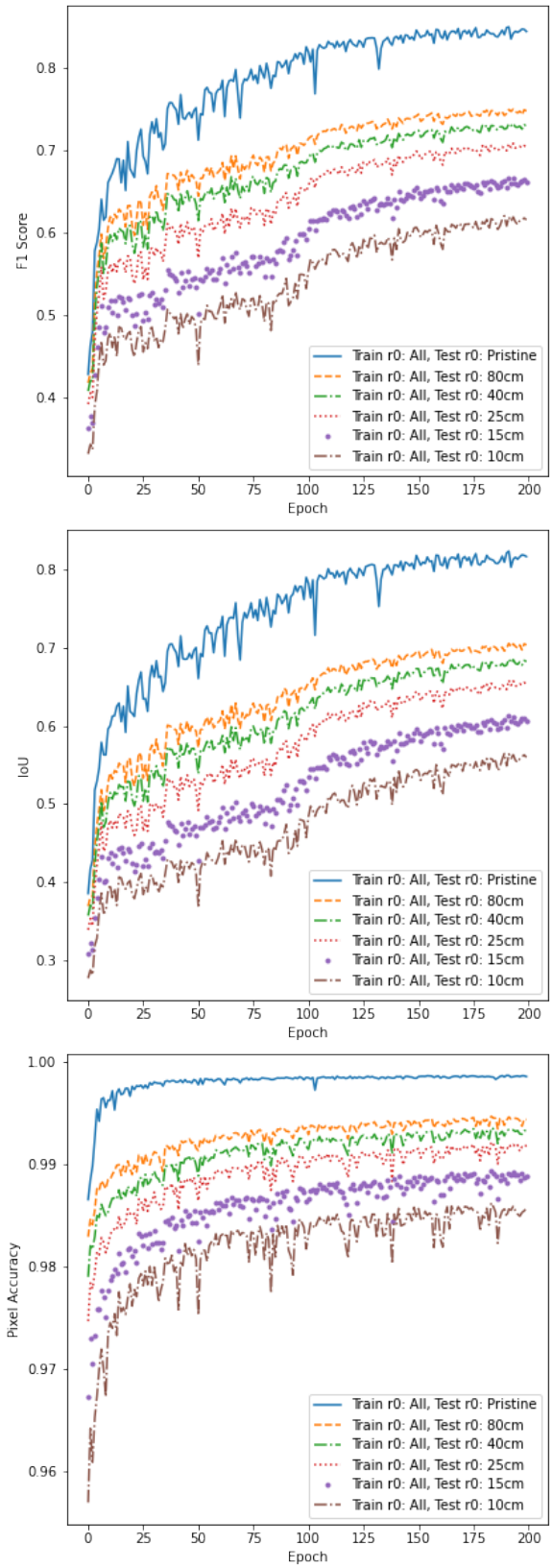


Figure 6: Comparison of U-Net performance at different turbulence levels of Boxsat when trained on multiple turbulence levels of Boxsat.

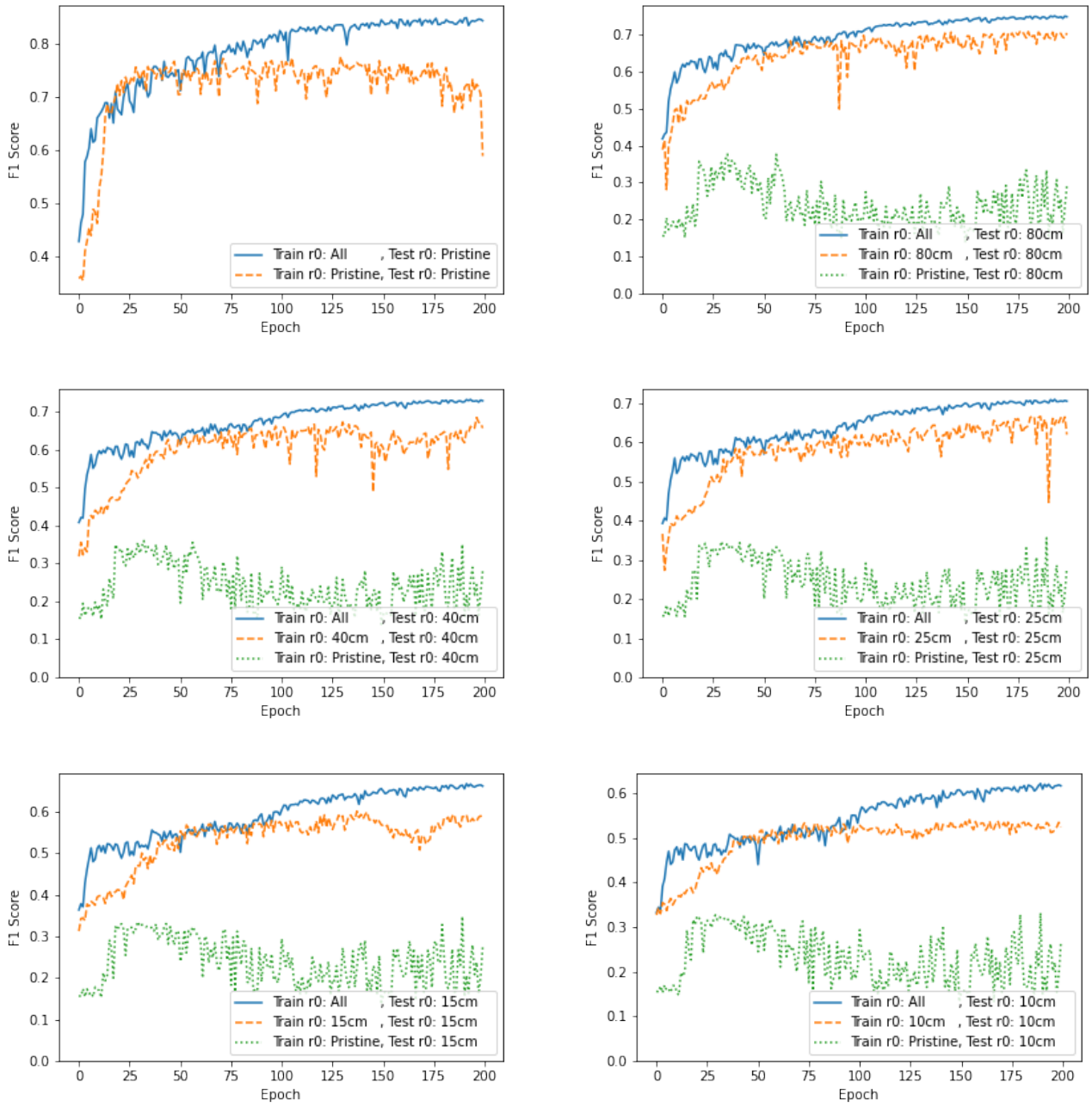


Figure 7: **Comparison of model trained on single turbulence levels of Boxsat and model trained on all turbulence levels of Boxsat**

Table 3: **Performance of U-Net trained on Boxsat at single turbulence levels. Values are given as F_1 score, IoU, pixel accuracy. Bolded values are best performance for each training r_0. Italicized value is the anomaly discussed in section 4 for Scenario B.**

		Training r_0 (cm)					
		Pristine	80	40	25	15	10
Test r_0 (cm)	Pristine	**0.77, 0.73, 1.00**	0.38, 0.31, 0.96	0.32, 0.26, 0.95	0.33, 0.28, 0.94	0.23, 0.20, 0.93	0.29, 0.26, 0.93
	80	0.28, 0.23, 0.70	**0.69, 0.62, 0.99**	0.59, 0.51, 0.98	0.58, 0.50, 0.98	0.35, 0.28, 0.95	0.30, 0.24, 0.94
	40	0.31, 0.26, 0.70	0.59, 0.51, 0.99	**0.62, 0.55, 0.99**	*0.64, 0.56, 0.98*	0.43, 0.35, 0.96	0.33, 0.26, 0.94
	25	0.31, 0.28, 0.71	0.47, 0.40, 0.97	0.54, 0.47, 0.98	**0.66, 0.60, 0.99**	0.51, 0.43, 0.97	0.38, 0.30, 0.95
	15	0.31, 0.28, 0.72	0.36, 0.30, 0.95	0.39, 0.33, 0.96	0.53, 0.44, 0.97	**0.57, 0.50, 0.99**	0.48, 0.40, 0.97
	10	0.30, 0.27, 0.71	0.30, 0.24, 0.93	0.31, 0.25, 0.92	0.39, 0.31, 0.94	0.44, 0.37, 0.96	**0.53, 0.46, 0.98**

Table 4: **Performance of models trained on single turbulence levels of Boxsat compared to that of models trained on all turbulence levels of Boxsat. Values are given in the order of F_1 score, IoU, pixel accuracy.**

Test r_0	Trained on Same Turbulence as Test	Trained on All Turbulence Levels	Δ
Pristine	0.77, 0.73, 1.00	0.84, 0.81, 1.00	0.07, 0.08, 0.00
80cm	0.69, 0.62, 0.99	0.74, 0.70, 0.99	0.05, 0.07, 0.00
40cm	0.62, 0.55, 0.99	0.72, 0.67, 0.99	0.10, 0.13, 0.00
25cm	0.66, 0.60, 0.99	0.70, 0.64, 0.99	0.04, 0.05, 0.00
15cm	0.57, 0.50, 0.99	0.65, 0.60, 0.99	0.08, 0.10, 0.00
10cm	0.53, 0.46, 0.99	0.60, 0.54, 0.99	0.07, 0.09, 0.00

Table 5: **Performance of models trained on all turbulence levels of Boxsat and that of models trained on all but one turbulence level of Boxsat. Values are given in the order of F_1 score, IoU, pixel accuracy.**

Test r_0	Trained on All Turbulence Levels	Trained on All Turbulence Levels Excluding $r_0 = 25cm$	Δ
Pristine	0.83, 0.81, 1.00	0.84, 0.81, 1.00	0.01, 0.01, 0.00
80cm	0.74, 0.70, 0.99	0.74, 0.70, 0.99	0.00, 0.00, 0.00
40cm	0.72, 0.67, 0.99	0.72, 0.67, 0.99	0.00, 0.00, 0.00
25cm	0.68, 0.63, 0.99	0.70, 0.64, 0.99	0.01, 0.01, 0.00
15cm	0.65, 0.60, 0.99	0.65, 0.60, 0.99	0.01, 0.00, 0.00
10cm	0.61, 0.56, 0.98	0.60, 0.54, 0.98	-0.01, -0.01, 0.00

measuring the performance of the models epoch by epoch as in Figure 7, the models trained on multiple turbulence levels continued to learn even at the later epochs while the models trained on single turbulence levels leveled off towards the latter half of training. This indicates that the model trained on all turbulence levels could have seen even better performance if it had been trained with more epochs or a higher learning rate. On the other hand, the model trained on single turbulence levels could have also benefited from further regularization.

For the second part of Scenario C, training a model on all but one turbulence level, the resulting model was able to segment across the full range of turbulence levels, even at the levels not included in training. As shown in Table 5, the performance on the excluded level was nearly identical to the model trained on all the turbulence levels. This further reinforces that the model can glean features of any turbulence level from features at other turbulence levels. As a result, the models trained in Scenario C, though only trained on a small set of discrete turbulence levels, can be used to segment images on a continuous spectrum of turbulence levels.

Scenario D: Reproducibility with Hubble— The results of Scenarios A-C were also replicated with the Hubble dataset. When training with Hubble renders for Scenario A, F_1 was measured at 0.66, IoU at 0.59, and pixel accuracy at 0.99. Table 6 shows the F_1 with relation to turbulence for Scenario B, and Table 7 shows the turbulence for Scenario C.

Scenario E: Generalizing Poses— The models trained for Scenario E as described in section 4 were then tested using the 200 most similar images previously identified and the regular test set. The results for both are shown in Figure 8 for each turbulence level. As shown, the performance between the two sets is very similar. This suggests that the U-Net is capable of learning to semantically segment the images of Hubble rather than mapping the images to the most similar input from the training set.

Scenario F: Multiple Satellites—When we expanded training to multiple satellites at pristine levels, the resulting model was able to segment the images for the satellites within the training set. However, F_1 score, IoU and pixel accuracy for segmenting the satellites within the training set is 0.67, 0.64, and 1.00 respectively compared to 0.45, 0.39, and 0.97 respectively for Hubble which was excluded from the training set. The difference in performance indicates that the model is overfitting to our limited training dataset of 6 satellites. This trend also continues as we introduce turbulence as shown by the performance in Table 8.

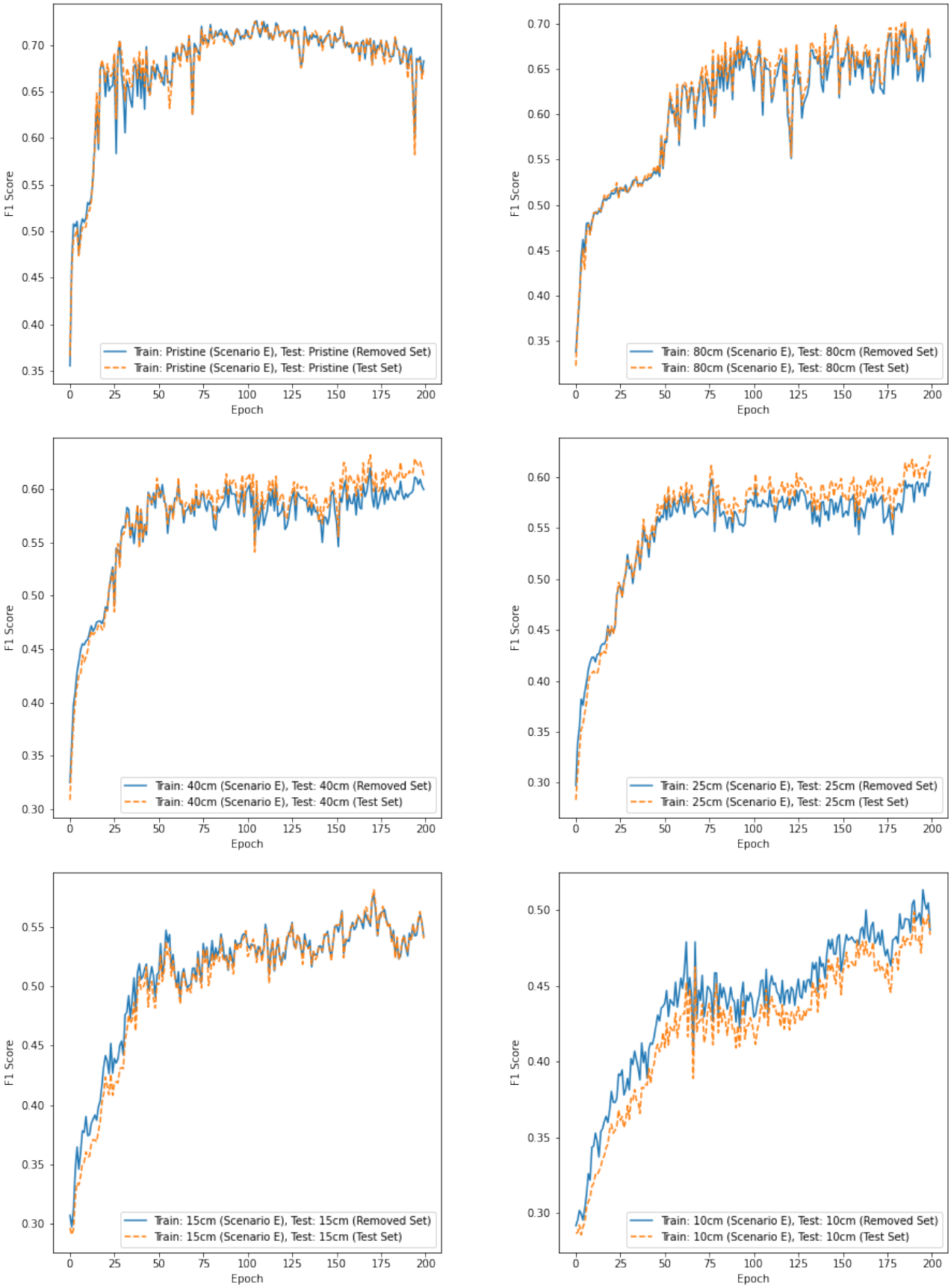


Figure 8: **Comparison of performance of model trained for Scenario E when tested on images removed from the dataset and images within the test dataset.**

Table 6: **Performance of U-Net trained on Hubble at single turbulence levels. Values are given in the order of F_1 score, IoU, pixel accuracy. Bolded values are best performance for each training r_0.**

	Training r_0	Pristine	80	40	25	15	10
Test r_0 (cm)	Pristine	**0.66, 0.59, 0.99**	0.46, 0.39, 0.97	0.30, 0.24, 0.93	0.33, 0.26, 0.92	0.21, 0.18, 0.90	0.24, 0.20, 0.90
	80	0.28, 0.22, 0.87	**0.66, 0.57, 0.99**	0.60, 0.50, 0.98	0.53, 0.44, 0.97	0.29, 0.24, 0.92	0.25, 0.21, 0.90
	40	0.25, 0.20, 0.86	0.60, 0.51, 0.98	**0.63, 0.55, 0.99**	0.61, 0.52, 0.98	0.35, 0.29, 0.94	0.25, 0.21, 0.90
	25	0.23, 0.19, 0.86	0.48, 0.40, 0.96	0.59, 0.50, 0.98	0**.63, 0.55, 0.98**	0.44, 0.36, 0.96	0.30, 0.24, 0.92
	15	0.20, 0.17, 0.86	0.34, 0.27, 0.90	0.41, 0.33, 0.93	0.53, 0.44, 0.96	**0.52, 0.44, 0.98**	0.44, 0.35, 0.95
	10	0.18, 0.15, 0.85	0.27, 0.21, 0.85	0.31, 0.25, 0.87	0.39, 0.31, 0.90	0.42, 0.34, 0.95	**0.50, 0.42, 0.97**

Table 7: **Performance of models trained on single turbulence levels of Hubble compared to that of models trained on all turbulence levels of Hubble. Values are given in the order of F_1 score, IoU, pixel accuracy.**

Test r_0 (cm)	Trained on Same Turbulence as Test	Trained on All Turbulence Levels	Δ
Pristine	0.66, 0.59, 0.99	0.67, 0.60, 0.98	0.02, 0.00, -0.01
80cm	0.66, 0.57, 0.99	0.73, 0.67, 0.99	0.08, 0.10, 0.00
40cm	0.63, 0.55, 0.99	0.72, 0.66, 0.99	0.09, 0.11, 0.00
25cm	0.63, 0.55, 0.98	0.70, 0.63, 0.98	0.06, 0.09, 0.00
15cm	0.52, 0.44, 0.98	0.66, 0.59, 0.98	0.13, 0.15, 0.00
10cm	0.50, 0.42, 0.97	0.60, 0.53, 0.97	0.10, 0.11, 0.00

Table 8: **Performance of U-Net trained on all satellites except Hubble at single turbulence levels. Values are given in the order of F_1 score, IoU, pixel accuracy. Test set either included Boxsat, COBE, MightySat, TAOS, TIROS, and WIRE as indicated by "All", or just Hubble as indicated by "Hub."**

		Test r_0 (cm)					
		Pristine		80		40	
		All	Hub.	All	Hub.	All	Hub.
Training r_0	Pristine	0.67, 0.64, 1.00	0.45, 0.39, 0.97	0.11, 0.08, 0.36	0.10, 0.07, 0.40	0.10, 0.07, 0.34	0.09, 0.07, 0.37
	80cm	0.36, 0.30, 0.93	0.27, 0.23, 0.93	0.60, 0.55, 0.99	0.41, 0.36, 0.96	0.54, 0.48, 0.97	0.39, 0.33, 0.95
	40cm	0.28, 0.22, 0.85	0.22, 0.18, 0.87	0.55, 0.50, 0.98	0.37, 0.32, 0.96	0.58, 0.53, 0.99	0.39, 0.33, 0.96
	25cm	0.27, 0.22, 0.89	0.18, 0.16, 0.90	0.45, 0.39, 0.95	0.31, 0.26, 0.94	0.52, 0.46, 0.97	0.35, 0.30, 0.95
	15cm	0.20, 0.16, 0.84	0.17, 0.15, 0.89	0.26, 0.21, 0.89	0.19, 0.17, 0.90	0.37, 0.30, 0.93	0.24, 0.21, 0.92
	10cm	0.21, 0.18, 0.87	0.18, 0.16, 0.90	0.19, 0.16, 0.87	0.16, 0.15, 0.90	0.22, 0.18, 0.88	0.17, 0.16, 0.90
		25		15		10	
		All	Hub.	All	Hub.	All	Hub.
Training r_0	Pristine	0.10, 0.06, 0.31	0.09, 0.06, 0.33	0.08, 0.05, 0.25	0.07, 0.04, 0.22	0.07, 0.04, 0.16	0.05, 0.03, 0.11
	80cm	0.44, 0.38, 0.94	0.32, 0.27, 0.91	0.29, 0.23, 0.83	0.25, 0.21, 0.85	0.21, 0.17, 0.77	0.21, 0.18, 0.83
	40cm	0.52, 0.46, 0.97	0.36, 0.30, 0.94	0.36, 0.30, 0.89	0.29, 0.24, 0.89	0.27, 0.21, 0.81	0.25, 0.20, 0.84
	25cm	0.55, 0.50, 0.98	0.37, 0.31, 0.95	0.45, 0.39, 0.94	0.32, 0.27, 0.92	0.31, 0.25, 0.84	0.27, 0.22, 0.86
	15cm	0.45, 0.39, 0.95	0.30, 0.25, 0.93	0.52, 0.47, 0.98	0.32, 0.27, 0.94	0.41, 0.34, 0.92	0.29, 0.24, 0.90
	10cm	0.28, 0.23, 0.90	0.19, 0.17, 0.90	0.41, 0.34, 0.94	0.25, 0.21, 0.92	0.49, 0.44, 0.97	0.29, 0.24, 0.92

Despite overfitting to the satellites within the training set, the overall performance of these experiments has shown that semantic segmentation of ground-based images of LEO satellites through turbulence is viable.

5. CONCLUSION

In this work, we have provided a convolutional neural network approach for segmenting ground-based images of LEO satellites. We have shown that a U-Net approach for this semantic segmentation task can segment images through a wide range of atmospheric turbulence levels. We have also shown that the network can segment multiple different satellites. For future work, we plan to incorporate procedural satellite generation for better regularization and avoid overfitting to just the satellites within the training dataset. Additionally, we plan to explore performance with satellite images with unique conditions such as glints, smear, and jitter. We wish to also explore the network's ability to segment satellites when taking into account the class imbalance present.

ACKNOWLEDGMENTS

This work was supported by the Air Force Research Laboratory and is approved for public release under case number AFRL-2021-3547.

REFERENCES

[1] B. Artacho and A. Savakis, "Waterfall atrous spatial pooling architecture for efficient semantic segmentation," *Sensors*, vol. 19, no. 24, 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/24/5361

[2] J. C. Caicedo, J. Roth, A. Goodman, T. Becker, K. W. Karhohs, C. McQuin, S. Singh, and A. E. Carpenter, "Evaluation of deep learning strategies for nucleus segmentation in fluorescence images," *bioRxiv*, 2018. [Online]. Available: https://www.biorxiv.org/content/early/2018/05/31/335216

[3] M. Drozdzal, E. Vorontsov, G. Chartrand, S. Kadoury, and C. Pal, "The importance of skip connections in biomedical image segmentation," 2016.

[4] F. Lateef and Y. Ruichek, "Survey on semantic segmentation using deep learning techniques," *Neurocomputing*, vol. 338, pp. 321–348, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S092523121930181X

[5] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," 2015.

[6] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015.

[7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR09*, 2009.

[8] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge," 2015.

[9] C. Liu, B. Zoph, J. Shlens, W. Hua, L. Li, L. Fei-Fei, A. L. Yuille, J. Huang, and K. Murphy, "Progressive neural architecture search," *CoRR*, vol. abs/1712.00559, 2017. [Online]. Available: http://arxiv.org/abs/1712.00559

[10] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le, "Regularized evolution for image classifier architecture search," *CoRR*, vol. abs/1802.01548, 2018. [Online]. Available: http://arxiv.org/abs/1802.01548

[11] S. Jégou, M. Drozdzal, D. Vazquez, A. Romero, and Y. Bengio, "The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation," 2017.

[12] T. Remez, O. Litany, R. Giryes, and A. M. Bronstein, "Deep convolutional denoising of low-light images," 2017.

[13] J. Xie, L. Xu, and E. Chen, "Image denoising and inpainting with deep neural networks," in *Advances in neural information processing systems*, 2012, pp. 341–349.

[14] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising," *IEEE transactions on image processing*, vol. 26, no. 7, pp. 3142–3155, 2017.

[15] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, and L. Bottou, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion." *Journal of machine learning research*, vol. 11, no. 12, 2010.

[16] N. Gagnier, J. Lucas, T. Kyono, M. Werth, I. McQuaid, and J. Fletcher, "Silo: A machine learning dataset of synthetic ground-based observations of leo satellites," 03 2020, pp. 1–8.

[17] D. L. Fried, "Optical resolution through a randomly inhomogeneous medium for very long and very short exposures," *J. Opt. Soc. Am.*, vol. 56, no. 10, pp. 1372–1379, Oct 1966. [Online]. Available: http://www.osapublishing.org/abstract.cfm?URI=josa-56-10-1372

[18] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, Y. W. Teh and M. Titterington, Eds., vol. 9. Chia Laguna Resort, Sardinia, Italy: PMLR, 13–15 May 2010, pp. 249–256. [Online]. Available: http://proceedings.mlr.press/v9/glorot10a.html

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *2015 IEEE International Conference on Computer Vision (ICCV)*. Los Alamitos, CA, USA: IEEE Computer Society, dec 2015, pp. 1026–1034. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/ICCV.2015.123

[20] A. Buslaeve, A. Parinov, E. Khvedcheny, V. I. Iglovikov, and A. A. Kalinin, "Albumentations: fast and flexible image augmentations," *ArXiv e-prints*, 2018.

[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.

BIOGRAPHY

Julia Yang *is a machine learning engineer with Boeing at the Air Force Maui Optical and Supercomputing (AMOS) site and works on several programs. Her work primarily focuses on machine learning and computer vision for applications in the space domain. Julia received a B.S. in Electrical Engineering from the California Institute of Technology in 2019.*

Jacob Lucas *is a physicist and machine learning engineer at the Air Force Maui Optical and Supercomputing (AMOS) site. He is primarily involved in new applications of machine learning and computer vision, with interests including Space Situational Awareness, active imaging, novel sensor design, and imaging through turbulence. He received an M.S. in Physics from the University of Hawaii at Manoa in 2009.*

Trent Kyono *is a machine learning engineer and researcher working for Boeing on several programs related to space situational awareness, autonomous aerial refueling, and machine learning assurances for autonomous aircraft. His areas of interest include applied deep networks, robust machine learning, computer vision, and causal modelling. He received an M.S. in Computer Science from UCLA in 2010 under Turing Award Winner Judea Pearl. He is currently finishing his Ph.D. at UCLA researching novel applications of machine learning in healthcare.*

Michael Abercrombie *Michael Abercrombie is a systems engineer and member of Boeing's Science and Analysis team at the Air Force Maui Optical and Supercomputing Site. His primary research interests include photometry and non-resolved object characterization as it applies to the SSA domain. Michael received his PhD in Physics from Washington University in St. Louis for work on laboratory tests of General Relativity in 2016, and BS degrees in Astrophysics and Mathematics from Florida Institute of Technology in 2011.*

Andrew Vanden Berg *is a Captain in the US Air Force, and he serves as a research engineer for the Air Force Research Laboratory, Detachment 15, located at the Air Force Maui Optical & Supercomputing Site. His work focuses on the development and deployment of machine learning models and autonomous systems for operations in the space domain. Andrew received his BS in Applied Mathematics from the US Air Force Academy in 2016 and his SM in Operations Research from the Massachusetts Institute of Technology in 2018.*

Justin Fletcher *received the M.S. degree in computer science from the Air Force Institute of Technology, Dayton, OH, USA, in 2016, with a concentration in machine learning. Mr. Fletcher is an advisor and subject matter expert on the enterprise adoption of deep learning technologies for the United States Space Force, Space Systems Command, and serves as a Major in the United States Air Force reserves. His current research interests include applications of computer vision to space object sensing, reinforcement learning for optical system actuation, deep learning framework optimization on high performance computing systems, and low size, weight, and power deep learning applications.*