# Semantic Segmentation of Low Earth Orbit Satellites using Convolutional Neural Networks

**Julia Yang, Jacob Lucas, Trent Kyono, Michael Abercrombie**
*The Boeing Company*
**Justin Fletcher**
*Odyssey Systems Consulting*
**Ian McQuaid**
*Air Force Research Laboratory*

## ABSTRACT

Ground-based imaging of Low Earth Objects (LEO) is subject to perturbations by atmospheric turbulence, which makes it difficult to identify key features or components on the object of interest. Techniques for reconstructing images have been developed, but it is still up to a human to subjectively discern and identify truth features on a partially reconstructed image. In this paper, we present a neural network 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.

## 1. INTRODUCTION

Ground based imaging of Low Earth Orbit (LEO) satellites has been key to improved space domain awareness (SDA). However, image quality can be severely reduced by atmospheric turbulence. In such conditions, gathering information from the images and tasks such as labeling the different components of a satellite can become very difficult and increasingly susceptible to mistake when done by a human. However, this task of segmenting images into their individual components has seen revolutionary improvement through machine learning techniques [1, 4, 6, 14, 15, 17].

In this paper, we explore the feasibility of applying convolutional neural networks for semantic segmentation of ground based images of LEO satellites. We first created datasets of multiple satellites consisting of renders of the satellites both with and without atmospheric turbulence. Then, we designed and built the inference models used for the experiments. After the datasets and model were completed, we established 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 trained the model on single levels of turbulence, followed by training the model with multiple turbulence levels. Finally, we increased the complexity of the scenario by introducing multiple satellites to the model.

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, and 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 status quo for image classification, segmentation, and detection tasks. With the recent success of these algorithms in imaging competitions, such as Kaggle in [2, 11], ImageNet in [5, 18], 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 motivate our application of a customized neural network for semantic segmentation of satellites.

Our work draws on several related works demonstrating significant success of using CNNs for semantic segmentation [4, 6, 12, 15, 17]. There has also been significant improvements in performance for interpreting noisy and degraded images using CNNs [16, 20, 21]. Specifically, stacked denoising autoencoders were presented as early as 2010 by [19] which has been iterated on to develop new architectures, such as the U-Net, yielding state-of-the-art in many image segmentation tasks [17]. 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 where by $y_i \in Y$ we denote the $i^{th}$ 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 $x_i$ a 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.

### 4.1 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 3.6m AEOS (Advanced Electro-Optical System) telescope at the summit of Haleakala.

The satellites used were 6 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 6 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 for clarity.

The truth labels consisted of 6 different classes: bus, solar panels, thrusters, payloads, antenna, and background. Renders of these satellites as well as their class labels were produced using COAST/FIST with image properties reflecting that of diffraction limited images from the AEOS telescope.

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 [8] to 5 different turbulence levels as represented in Table 1. The turbulence levels were characterized by the Freid parameter ($r_0$) [7]. 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 Fig. 2. 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.

The best performing inference for each satellite at each turbulence level can be seen in Fig. 1.

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 |

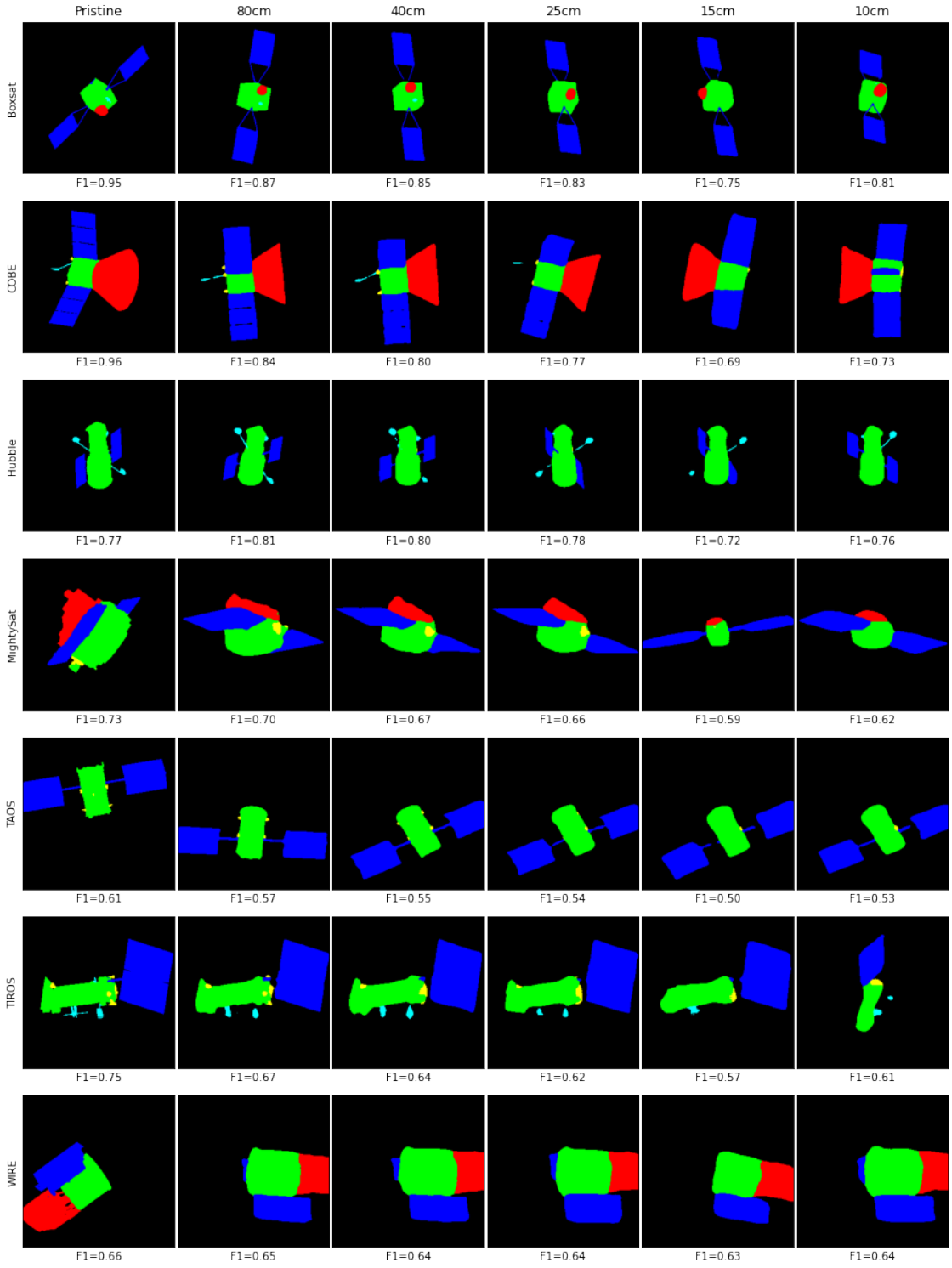
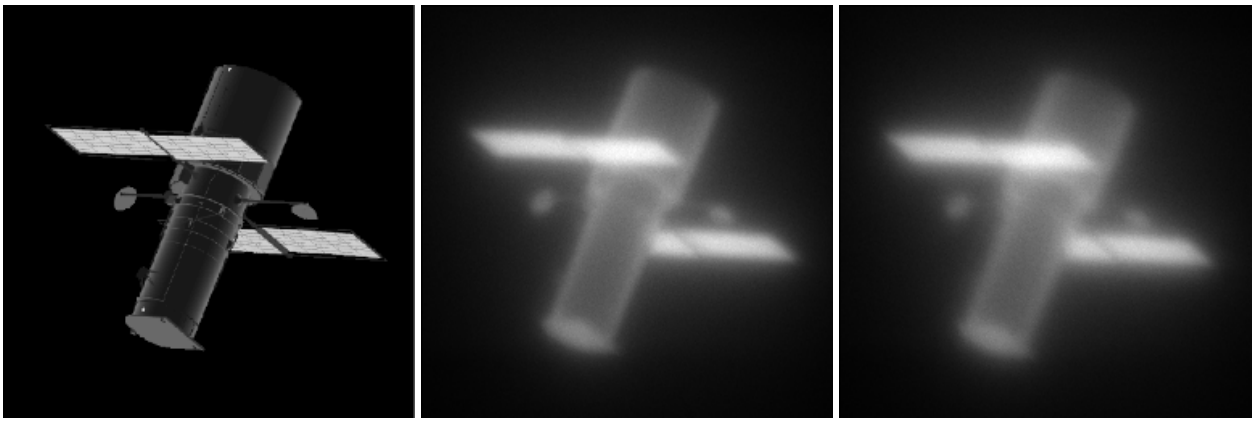
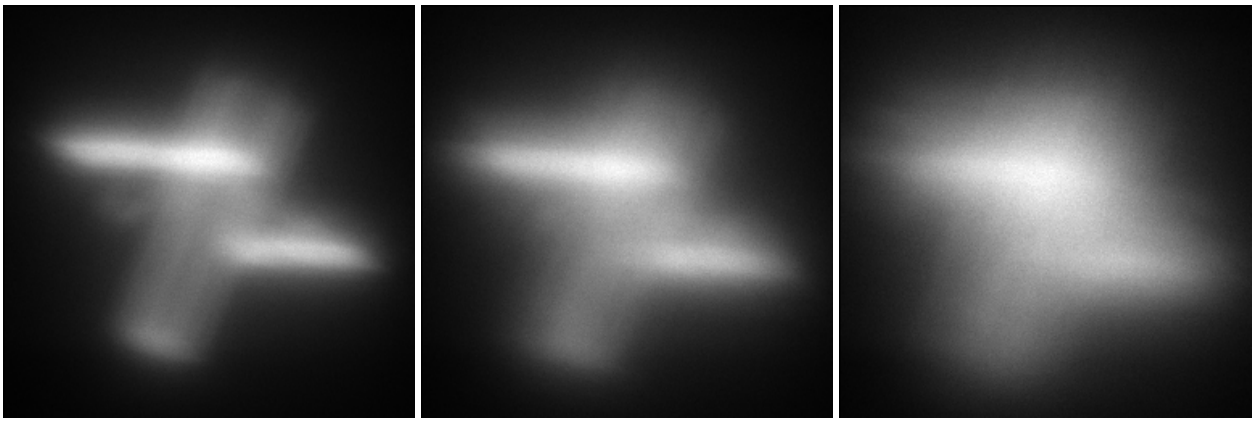



Fig. 1: The 7 different satellites used as well as their best inference at each turbulence level. Green represents satellite bus, blue represents solar panels, cyan represents antenna, red represents payload, yellow represents thrusters, and black represents the background.



(a) Pristine (b) $r_0 = 80cm$ (c) $r_0 = 40cm$



(d) $r_0 = 25cm$ (e) $r_0 = 15cm$ (f) $r_0 = 10cm$

Fig. 2: Examples of renders of Hubble at the various turbulence levels

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

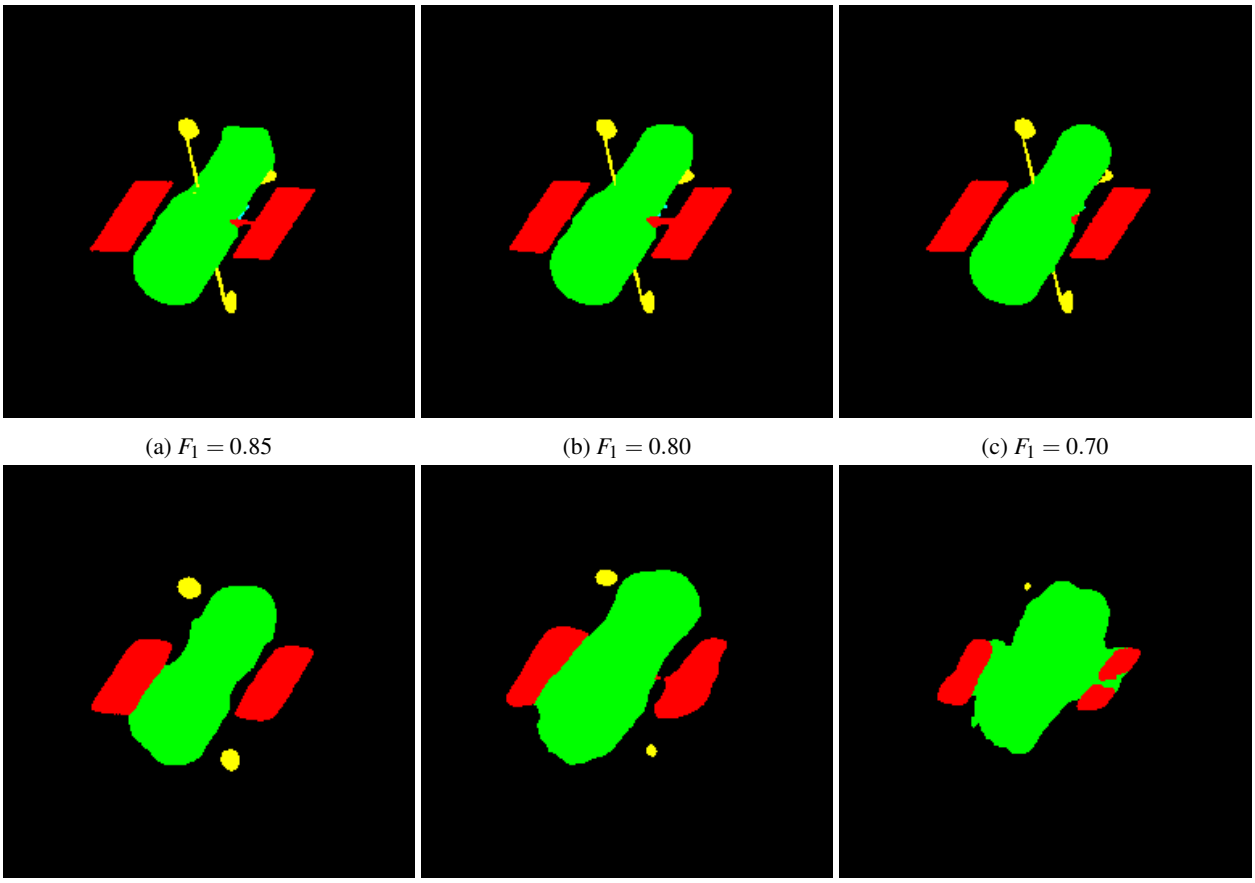
| Parameter | Value | Unit |
|---|---|---|
| Image Size | 256x256 | pixels |
| IFOV | 100 | nrad |
| PSF Generation Rate | 800 | Hz |
| Frame Rate | 40 | Hz |
| Waveband | Bessel I | - |
| Turbulence Profile | MK50P | - |
| Transmission Noise Coefficient | 0.7 | - |
| Quantum Efficiency | 0.9 | - |
|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

### 4.2 Metrics

For this study we measure performance using the $F_1$ score, also known as the Sørensen–Dice coefficient. The $F_1$ score measures the ability of the models to properly segment the image and is defined by Eq. 1.

$$F_1 = \frac{2|A \cap B|}{|A| + |B|} \tag{1}$$

The metric is calculated for each batch of images and then averaged across the entire test set. A visual representation of $F_1$ score is shown in Fig. 3.



(a) $F_1 = 0.85$ (b) $F_1 = 0.80$ (c) $F_1 = 0.70$

(d) $F_1 = 0.55$ (e) $F_1 = 0.45$ (f) $F_1 = 0.35$

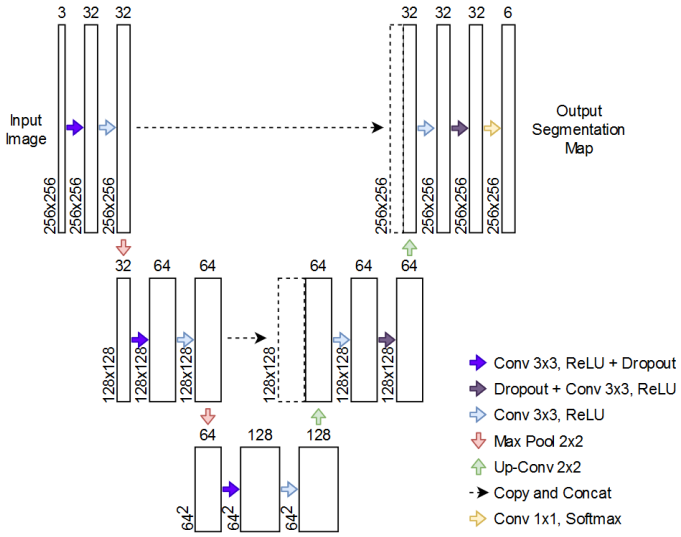
Fig. 3: Examples of $F_1$ scored inferences of Hubble. Green represents satellite bus, red represents solar panels, yellow represents antenna, and black represents the background.

### 4.3 Architecture

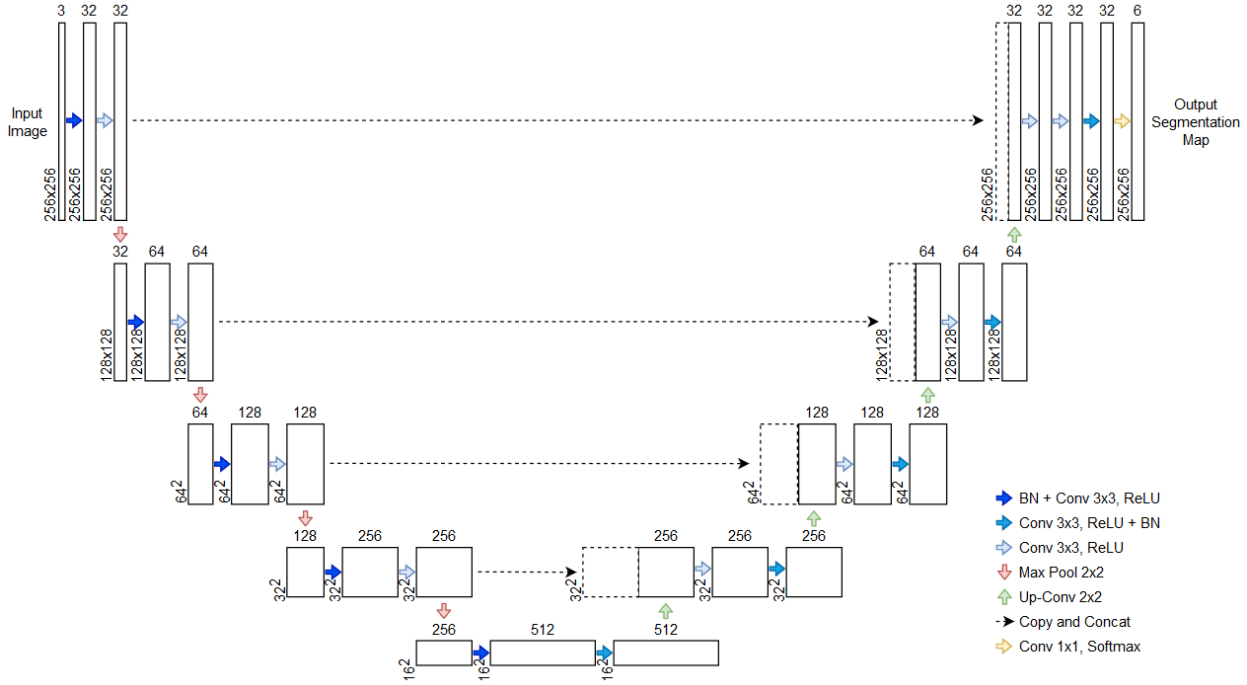
In this work, we use a U-Net network based on previous success of U-Net for semantic segmentation [4]. We used both a depth 3 U-Net depicted in Fig. 4a and a depth 5 U-Net depicted in Fig. 4b.

All convolutional layers of the depth 3 U-Net used a ReLU activation function except the last. We initialized each kernel to Glorot Uniform [9]. The depth 5 U-Net also used ReLU for all activation layers except the last. The last layer used softmax. Additionally, each kernel was initialized to He Normal [10].

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 [3], 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 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 $1e^{-3}$ [13]. Categorical crossentropy was used as the loss function. The CNN was trained for 200 epochs, and the best validation loss was saved for inference/prediction. Image augmentation was not used during inference.




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




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

Fig. 4: U-Net Structures

### 4.4 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.

### 4.5 Training Method and Performance

We begin the training by establishing a baseline of performance of the model trained on the pristine Boxsat dataset. Then we evaluate the model's ability to segment when the dataset is subject to a a single turbulence level when trained

on single turbulence levels. Then we expand to training a single model on Boxsat at multiple turbulence levels. Finally, we attempt to expand to multiple satellites.

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 Fig. 4a. At this baseline, the U-Net semantically segments pristine images of the test set at an $F_1$ score of 0.77. When this baseline model is used to semantically segment images with turbulence, it consistently performed poorly with a mean $F_1$ score of 0.30.

However, 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 today's images, it is necessary to evaluate the performance of such models through turbulence. As such, we started by training a model on Boxsat for each turbulence level representing the different conditions as specified in Table 1. The depth 3 U-Net would not converge when presented with turbulence, so we increased depth of the U-Net to depth 5, removed the dropout layers, and included batch normalization layers. This U-Net can be seen in Fig. 4b. The depth 5 U-Net converged and segmented through noise. The performance of the depth 5 U-Net when tested on images with the same turbulence level as used in training can be seen in Fig 5. As shown, performance does decrease with turbulence.

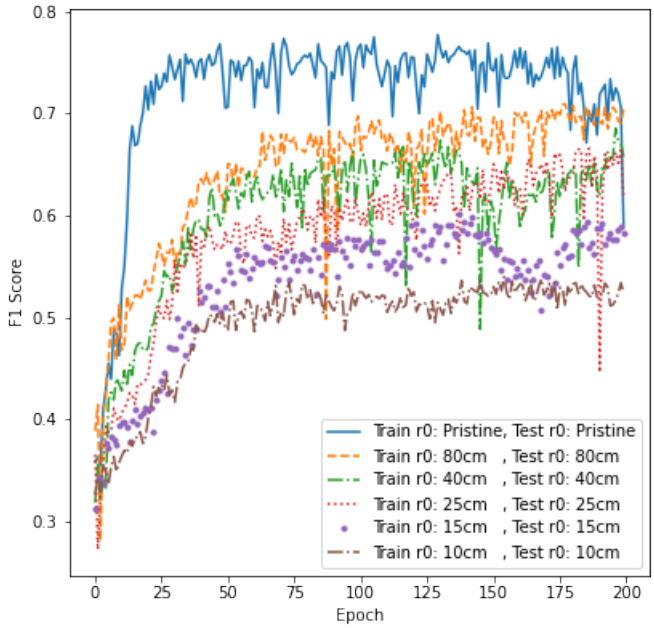



Fig. 5: Comparison of U-Net performance at different turbulence levels when trained and tested on a single turbulence level of Boxsat

To further test the generalizability of the models trained on each $r_0$ value, we measured the performance of each model at all turbulence levels of Boxsat. The performance of the models as shown in Table 3 generally decreases as the difference increases in $r_0$ value for the train and test sets. 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 ($F_1 = 0.77$) and 10cm ($F_1 = 0.53$) 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 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, in order to have optimal performance at a specific $r_0$, a model trained for that $r_0$ is needed.

To eliminate this need of training a model for each turbulence level, we trained a model using the full range of

Table 3: $F_1$ scores for U-Net trained on Boxsat at single turbulence levels. Bolded values are best performance for each training $r_0$. Italicized value is the anomaly discussed in section 4.5.

| | | Training $r_0$ (cm) | | | | | |
|---|---|---|---|---|---|---|---|
| | | Pristine | 80 | 40 | 25 | 15 | 10 |
| Test $r_0$ (cm) | Pristine | **0.77** | 0.38 | 0.32 | 0.33 | 0.23 | 0.29 |
| | 80cm | 0.28 | **0.69** | 0.59 | 0.58 | 0.35 | 0.30 |
| | 40cm | 0.31 | 0.59 | **0.62** | *0.64* | 0.43 | 0.33 |
| | 25cm | 0.31 | 0.47 | 0.54 | **0.66** | 0.51 | 0.38 |
| | 15cm | 0.31 | 0.36 | 0.39 | 0.53 | **0.57** | 0.48 |
| | 10cm | 0.30 | 0.30 | 0.31 | 0.39 | 0.44 | **0.53** |

turbulence levels. The resulting model segments accurately and consistently across all turbulence levels. The resulting performance of the model when tested at every turbulence level can be seen Fig. 6. Additionally, as shown in Table 4, the model trained on all turbulence levels consistently performs better than the model trained at single turbulence level. Furthermore, when measuring the performance epoch by epoch as in Fig. 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. The model trained on single turbulence levels could have also benefited from further regularization.

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. 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 Fig. 5, the performance on the excluded level was nearly identical to the model trained on all the turbulence levels.

These results were also replicated with the Hubble dataset. When training with Hubble renders in the absence of turbulence, $F_1$ was measured at 0.66. Table 6 shows the $F_1$ with relation to turbulence when training with single turbulence levels, and Table 7 shows the turbulence when trained on a multiple turbulence levels.

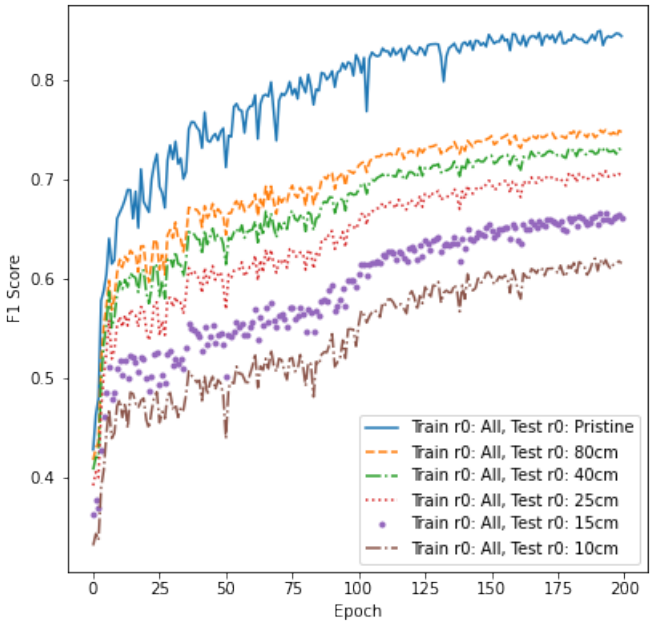



Fig. 6: Comparison of U-Net performance at different turbulence levels of Boxsat when trained on multiple turbulence levels of Boxsat.

Table 4: $F_1$ scores of models trained on single turbulence levels of Boxsat compared to that of models trained on all turbulence levels of Boxsat.

| Test $r_0$ | Trained on Same Turbulence as Test | Trained on All Turbulence Levels | $\Delta F_1$ |
|---|---|---|---|
| Pristine | 0.77 | 0.84 | 0.07 |
| 80cm | 0.69 | 0.74 | 0.05 |
| 40cm | 0.62 | 0.72 | 0.10 |
| 25cm | 0.66 | 0.70 | 0.04 |
| 15cm | 0.57 | 0.65 | 0.08 |
| 10cm | 0.53 | 0.60 | 0.07 |

Table 5: $F_1$ scores of models trained on all turbulence levels of Boxsat and that of models trained on all but one turbulence level of Boxsat.

| Test $r_0$ | Trained on All Turbulence Levels | Trained on All Turbulence Levels Excluding $r_0 = 25cm$ | $\Delta F_1$ |
|---|---|---|---|
| Pristine | 0.83 | 0.84 | 0.01 |
| 80cm | 0.74 | 0.74 | 0.00 |
| 40cm | 0.72 | 0.72 | 0.00 |
| 25cm | 0.68 | 0.70 | 0.02 |
| 15cm | 0.65 | 0.65 | 0.00 |
| 10cm | 0.61 | 0.60 | -0.01 |

Table 6: $F_1$ scores for U-Net trained on Hubble at single turbulence levels. 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** | 0.46 | 0.30 | 0.33 | 0.21 | 0.24 |
| | 80cm | 0.28 | **0.66** | 0.60 | 0.53 | 0.29 | 0.25 |
| | 40cm | 0.25 | 0.60 | **0.63** | 0.61 | 0.35 | 0.25 |
| | 25cm | 0.23 | 0.48 | 0.59 | **0.63** | 0.44 | 0.30 |
| | 15cm | 0.20 | 0.34 | 0.41 | 0.53 | **0.52** | 0.44 |
| | 10cm | 0.18 | 0.27 | 0.31 | 0.39 | 0.42 | **0.50** |

Table 7: Performance of models trained on single turbulence levels of Hubble compared to that of models trained on all turbulence levels of Hubble.

| Test $r_0$ (cm) | Trained on Same Turbulence as Test | Trained on All Turbulence Levels | $\Delta F_1$ |
|---|---|---|---|
| Pristine | 0.66 | 0.67 | 0.02 |
| 80cm | 0.66 | 0.73 | 0.08 |
| 40cm | 0.63 | 0.72 | 0.09 |
| 25cm | 0.63 | 0.70 | 0.06 |
| 15cm | 0.52 | 0.66 | 0.13 |
| 10cm | 0.50 | 0.60 | 0.10 |

Finally we attempted to create a model which can generalize across satellites. We 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. The

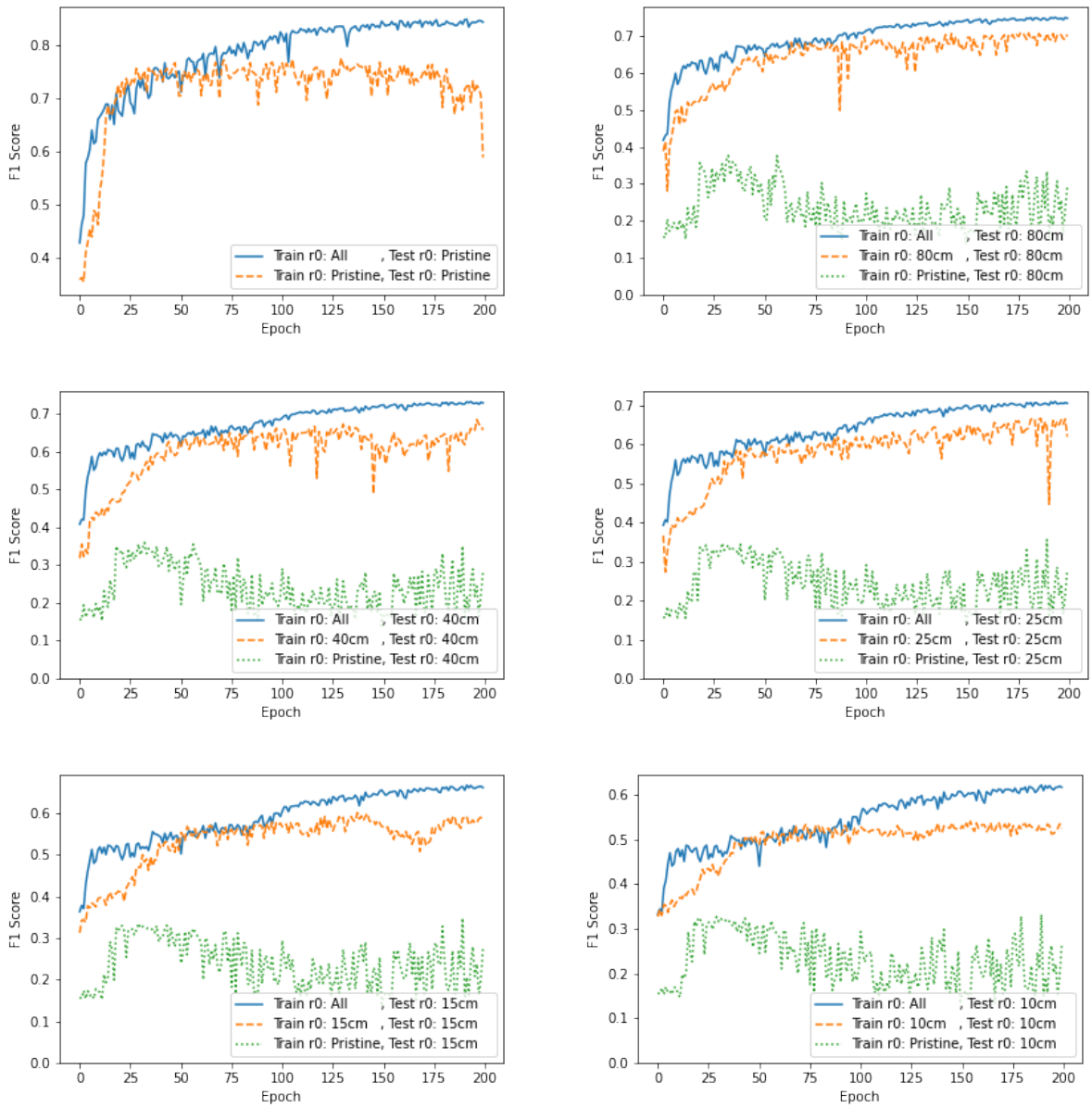



Fig. 7: Comparison of model trained on single turbulence levels of Boxsat and model trained on all turbulence levels of Boxsat

resulting model was able to segment the images for the satellites within the training set. However, $F_1$ for segmenting the satellites within the training set is 0.67 compared to 0.45 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 continues as we introduce turbulence as well (Table 8).

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.

Table 8: $F_1$ score for U-Net trained on all satellites except Hubble at single turbulence levels. 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 | | 25 | | 15 | | 10 | |
| | | All | Hub. | All | Hub. | All | Hub. | All | Hub. | All | Hub. | All | Hub. |
| Training $r_0$ (cm) | Pristine | 0.67 | 0.45 | 0.11 | 0.10 | 0.10 | 0.09 | 0.10 | 0.09 | 0.08 | 0.07 | 0.07 | 0.05 |
| | 80cm | 0.36 | 0.27 | 0.60 | 0.41 | 0.54 | 0.39 | 0.44 | 0.32 | 0.29 | 0.25 | 0.21 | 0.21 |
| | 40cm | 0.28 | 0.22 | 0.55 | 0.37 | 0.58 | 0.39 | 0.52 | 0.36 | 0.36 | 0.29 | 0.27 | 0.25 |
| | 25cm | 0.27 | 0.18 | 0.45 | 0.31 | 0.52 | 0.35 | 0.55 | 0.37 | 0.45 | 0.32 | 0.31 | 0.27 |
| | 15cm | 0.20 | 0.17 | 0.26 | 0.19 | 0.37 | 0.24 | 0.45 | 0.30 | 0.52 | 0.32 | 0.41 | 0.29 |
| | 10cm | 0.21 | 0.18 | 0.19 | 0.16 | 0.22 | 0.17 | 0.28 | 0.19 | 0.41 | 0.25 | 0.49 | 0.29 |

## 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 trained on a limited range of satellite poses.

## 6. REFERENCES


[1] Bruno Artacho and Andreas Savakis. Waterfall atrous spatial pooling architecture for efficient semantic segmentation. *Sensors*, 19(24), 2019.

[2] Casper Solheim Bojer and Jens Peder Meldgaard. Kaggle forecasting competitions: An overlooked learning opportunity. *International Journal of Forecasting*, 37(2):587–603, Apr 2021.

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

[4] Juan C. Caicedo, Jonathan Roth, Allen Goodman, Tim Becker, Kyle W. Karhohs, Claire McQuin, Shantanu Singh, and Anne E. Carpenter. Evaluation of deep learning strategies for nucleus segmentation in fluorescence images. *bioRxiv*, 2018.

[5] 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.

[6] Michal Drozdzal, Eugene Vorontsov, Gabriel Chartrand, Samuel Kadoury, and Chris Pal. The importance of skip connections in biomedical image segmentation, 2016.

[7] D. L. Fried. Optical resolution through a randomly inhomogeneous medium for very long and very short exposures. *J. Opt. Soc. Am.*, 56(10):1372–1379, Oct 1966.

[8] Nicole Gagnier, Jacob Lucas, Trent Kyono, Michael Werth, Ian McQuaid, and Justin Fletcher. Silo: A machine learning dataset of synthetic ground-based observations of leo satellites. pages 1–8, 03 2020.

[9] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In Yee Whye Teh and Mike Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR.

[10] 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)*, pages 1026–1034, Los Alamitos, CA, USA, dec 2015. IEEE Computer Society.

[11] Vladimir Iglovikov, Sergey Mushinskiy, and Vladimir Osin. Satellite imagery feature detection using deep convolutional neural network: A kaggle competition, 2017.

[12] Simon Jégou, Michal Drozdzal, David Vazquez, Adriana Romero, and Yoshua Bengio. The one hundred layers

tiramisu: Fully convolutional densenets for semantic segmentation, 2017.
[13] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.
[14] Fahad Lateef and Yassine Ruichek. Survey on semantic segmentation using deep learning techniques. *Neurocomputing*, 338:321–348, 2019.
[15] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation, 2015.
[16] Tal Remez, Or Litany, Raja Giryes, and Alex M. Bronstein. Deep convolutional denoising of low-light images, 2017.
[17] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.
[18] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge, 2015.
[19] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(12), 2010.
[20] Junyuan Xie, Linli Xu, and Enhong Chen. Image denoising and inpainting with deep neural networks. In *Advances in neural information processing systems*, pages 341–349, 2012.
[21] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE transactions on image processing*, 26(7):3142–3155, 2017.