Deep Learning for Cislunar Object Detection

Luca Ghilardi
System and Industrial Engineering, University of Arizona
Roberto Furfaro
System and Industrial Engineering, University of Arizona

Vishnu Reddy
Lunar and Planetary Laboratory, University of Arizona

ABSTRACT

In this paper, we want to present a method-based deep learning for object identification in astronomical images of the cislunar space. Specifically, we explore the application of convolutional encoders used in UNet and vision transformers (ViT). Modern machine learning methods are powerful and reliable in many image processing applications. In recent years, the methods based on transformer encoders have quickly become state-of-the-art. However, they require large datasets. Convolutional encoders, on the other hand, have good performance with small datasets. The main challenge of this work is to create models that perform well with images with a low signal-to-noise ratio (SNR) for targets close to the Moon. The dataset is based on real observations of the area around the moon, and the targets are synthetically added to have a dataset as close as possible to a deployment scenario.

1. INTRODUCTION

Astronomical image processing for objects in the cislunar space can be very challenging due to the low signal-to-noise (SNR) ratio, given by the high brightness of the Moon, and the significant errors in the predicted position of such objects [FRCG21, CFR$^+$22, RBC$^+$21]. Furthermore, with the increasing amount of data gathered by spacecraft and telescopes, the need for efficient and accurate automated identification systems has become more urgent. Most astronomical image processing techniques for objects not in Near-Earth-Orbit(NEO) are based on taking multiple exposures at regular intervals and determining if the target exhibits consistent movement from one frame to another. The limitation of this technique is that it requires knowledge of the movement of the object in advance. Nevertheless, it is a valid approach for conducting follow-up observations of a known object.

In this paper, we want to test two neural network encoder architectures to identify objects orbiting the cislunar space autonomously. Another application to achieve the same goal might be background estimation and subtraction [BKM$^+$11, PS15, BR22]. However, for this method, more real observations are required. These observations are going to be collected in the future.

Our approach or object determination is a classification problem with semantic segmentation. Each pixel of the input image will be assigned a label. The training has been conducted in a supervised fashion. Therefore the networks learn by comparing the results on a validation dataset with the correct labels of the validation input provided by the user. These correct labels are called ground truth.

The architectures considered are both based on an encored/decoder model. The encoder has the role of extracting the features of the image. However, the original spatial information is lost during this process, so we also need a decoder to link the features learned to the original image. In this work, we consider two different encoder architectures. The first encoder is a convolutional encoder, characteristic of convolutional neural networks (CNNs). Specifically, we adopt the famous UNet. It was first proposed for biomedical image segmentation [RFB15], although it has also been successfully employed in other science fields. This architecture is well known for its flexibility and performance with small datasets. The UNet is already been studied in various vision-based space applications, from hazard detection. The authors have adapted the model developed for vision-based hazard detection applications [GDS$^+$21, SDG$^+$20].

The network has also been used for object detection in astronomical images [DVCDLM22] for LEO/GEO objects, but not in a high-noise environment.

The second method is based on vision transformers (ViT) made by Dosovitskiy et al. in their paper ”An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale.” This method is quickly becoming state-of-the-art in image processing tasks. The ViT is a deep neural network architecture that replaces the traditional convolutional layers with a series of self-attention layers introduced by Vaswani et al. in ”Attention Is All You Need.” It has shown superior performance in image classification tasks when large datasets are available. ViT’s key advantages are handling high-resolution images, variable-size inputs, and better generalization, maintaining a global receptive field throughout the encoder. The authors explored this architecture in a regression problem [GSF22] with results surpassing the UNet ones. The main advantage of the ViT architecture is the lack of information loss with downsampling operations in the encoder. However, the drawback of this model is the necessity of an extensive training set.

The first idea was to build a large one by simulating the optical observations using Blender’s open-source ray tracing software. However, the final goal of this research is to develop a deployable algorithm that we can use for real observations. Recent in-house tests showed us that the algorithms that perform well on synthetic data perform poorly on real ones. Therefore we decided to take real observations of the area around the moon and synthetically add a target using a 2D Gaussian distribution. Once the dataset has been created, the data get pre-processed before going into the network by stacking the 5 exposures and superimposing the relative labels. The model’s performance will also be evaluated with different target signal levels to explore the SNR limits for reliable object identification in the cislunar space.

2. METHOD

The architecture and dataset creation are described in this section. In Figure REF workflow diagram is presented.

3. DATASET

The dataset has been based on real images taken at different positions around the moon. Specifically at the angles of 30, 60, 90, and 120 degrees with respect to the center of the moon in all four cardinal directions. The images were taken on the 7th of February, 2023, from Tucson, AZ. The moon at that moment was 92.2% of the full moon.

For this work, we discarded the images taken at 30 degrees from the moon since most of the pixels were saturated. Therefore, only small portions of those images were usable. The images are loaded in uint16 format. First, we choose a patch of size 384x384 pixels in the image, and we compute the median value of the pixels. This will be used as a base noise value for when we will add the signal of the synthetic target. In literature, computing the SNR of an image requires complete knowledge of the noises or more images with identical signals and uncorrelated noise [TSP01]. In this work, we refer to SNR as [FHDB21]:

$$SNR = \frac{S}{\sqrt{S+N}} \tag{1}$$

Where S and N are, respectively, the signal mean, and then the noise mean, where the noise is the value of the pixels in a window of 10% of the size of the original image around the target.

Once the patch location within the image has been defined, we randomize the object location within 60% of the patch size from the center. As mentioned, we add the signal of the synthetic target with a 2D Gaussian distribution around the object location selected. This simulates the effect of light of the target ”spilling” in the adjacent pixels.

For this work, we decided to collect 5 exposures of the object, keeping the telescope pointing in the same portion of the sky with respect to the moon. To simulate the target moving in the image, we selected a random direction and distance to move the target pixel. However, the rest of the background can not be constant between exposures. Therefore, we added a normal distribution noise to the whole image.

The relative labels are generated with a blank copy of the real images, and then we add the 2D Gaussian distribution of the target’s signal. The next step is to make the label image binary such that the pixels belonging to the target signal have a value of 1 and all the others zero, and then we crop it with the corresponding patch.

It is worth noticing that we tried different approaches to feed the images to the network. First, we tried to feed divide the image in non-overlapping patches of size 64x64 pixels. The problem with this approach is the class unbalancing. Only a few pixels belong to the target in the labels, and many do not. This unbalancing poisons the training process.

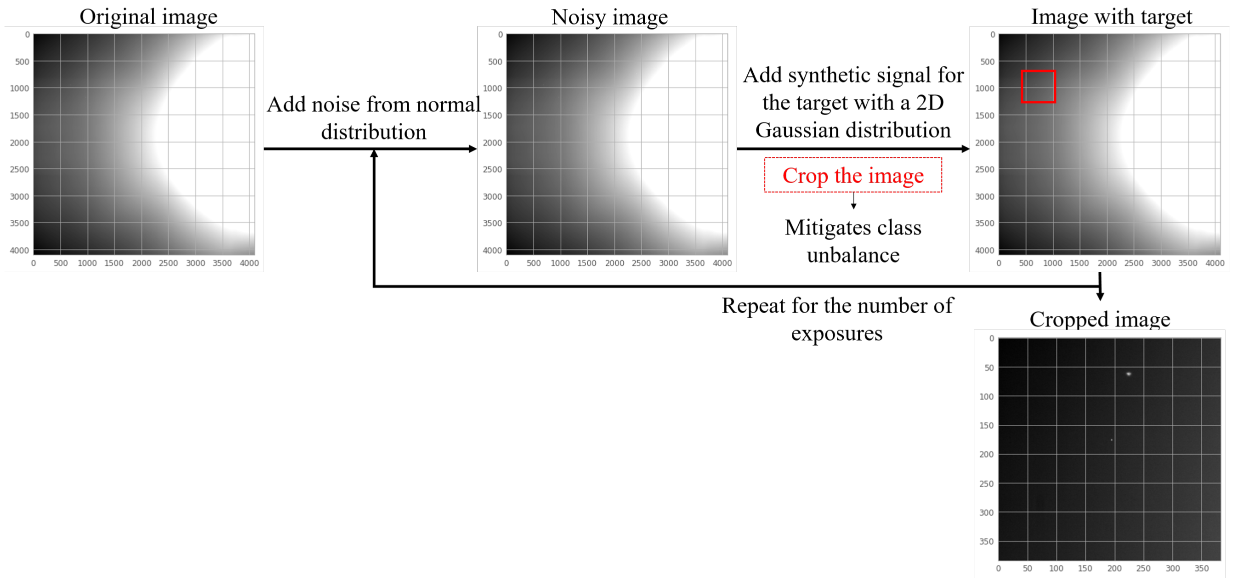

Original image
Add noise from normal distribution
Noisy image
Add synthetic signal for the target with a 2D Gaussian distribution
Crop the image
Mitigates class unbalance
Image with target
Repeat for the number of exposures
Cropped image
0
500
1000
1500
2000
2500
3000
3500
4000
0
50
100
150
200
250
300
350

Fig. 1: Procedure to add the synthetic signal to the real images.

By extracting a single patch of 384x384, the problem is not entirely solved, but it is mitigated. The networks' input has a size of *N_exposures* X *Patch_Width* X *Patch_Height*, which in our case are 5x384x384. The label, instead, is a single-channel binary image where the labels of the relative exposures are superimposed, Figure 2.

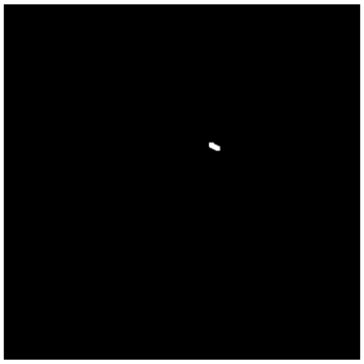


Fig. 2: Sample of the ground truth.

4. ARCHITECTURE

This section describes the two networks' architecture adopted in this work.

4.1 Vision Transformer(ViT)

The vision transformer layers were first introduced by Dosovitskiy et al. [DBK$^+$20] as a method based on the original transformer of Vaswani et al. [VSP$^+$17] for image classification, Figure 3. Here, we adopted the architecture created by Belkar et al. [1], which is more customizable and easier to understand.

Before being input into the transformer encoder, the image $X \in \mathbb{R}^{h \times w \times c}$ is subdivided into non-overlapping patches $X_p \in \mathbb{R}^{N \times (p^2 \cdot c)}$ called tokens. h and w are the spatial image resolution, c is the number of channels, (p, p) represents

[1] https://github.com/antocad/FocusOnDepth

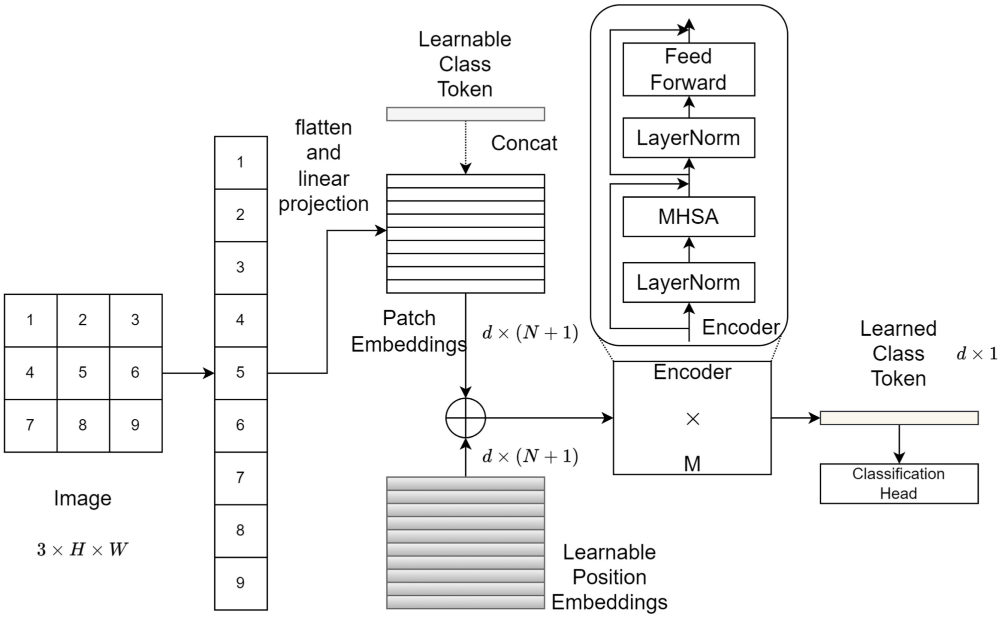


Fig. 3: The network architecture of ViT [JPLX22]

the patch size, and $N = hw/p^2$. Since the transformer uses constant latent vector size and is position invariant, the tokens are linearly embedded into a feature space D to identify the position and class of each token. This allows the model to weigh each patch's attention to its position.

$$\mathbf{z_0} = \left[\mathbf{x}_{class}; \mathbf{x}_p^1\mathbf{E}; \mathbf{x}_p^2\mathbf{E}; ...; \mathbf{x}_p^N\mathbf{E}\right] + \mathbf{E}_{pos} \qquad \text{Where } \mathbf{E} \in \mathbb{R}^{(p^2 \cdot c) \times D} \quad , \mathbf{E}_{pos} \in \mathbb{R}^{(N+1) \times D} \tag{2}$$

Once embedded, the patches are passed into the transformer encoder, which returns a value for each class token. As shown in Figure 3, the embedded patches get normalized and fed to the multi-head attention block. This block is based on the self-attention mechanism [KNH$^+$22].

The vectors derived from different inputs are packed in three matrices: queries $\mathbf{Q}$, keys $\mathbf{K}$, and values $\mathbf{V}$. Each query is compared with the keys to get scores/weights for the values. Each score/weight is, in short, the relevance between the query and each key. You reweight the values with the scores/weights and take the summation of the reweighted values. The attention function can be broken down into the following steps:

1 Compute scores $\mathbf{S} = \mathbf{Q} \cdot \mathbf{K}^T$, it measures the degree of attention of the surrounding image patches.

2 Normalize the scores for the stability of the gradient $\mathbf{S}_n = \mathbf{S}/\sqrt{D}$

3 Translate the scores into a probability with the softmax function $\mathbf{P} = softmax(\mathbf{S}_n)$

4 Compute weighted value matrix $\mathbf{Z} = \mathbf{P} \cdot \mathbf{V}$

Multihead self-attention (MHSA) is an extension of self-attention to boost its performance, where we run k self-attention operations in parallel.

$$\text{MultiHead}(\mathbf{Q}', \mathbf{K}', \mathbf{V}') = \text{Concat}(head_1, head_2, ..., head_h)\mathbf{W}_0 \qquad \mathbf{W}_0 \in \mathbb{R}^{D \times D} \tag{3}$$

$$\text{where} \qquad head_i = \text{Attention}(\mathbf{Q}_i, \mathbf{K}_i, \mathbf{V}_i) \tag{4}$$

$\mathbf{Q}', \mathbf{K}'$ and $\mathbf{V}'$ are the concatenation of $\mathbf{Q}_i^m, \mathbf{K}_i^m, \mathbf{V}_i^m$ where m is the number of parallel MHSA layers. At last, $\mathbf{W}_0$ is the projection weight. Following the design of the original transformer, [VSP$^+$17], after the self-attention layer, a feed-forward network (FFN) is applied to introduce non-linearities with activation functions. Essential to model complex relationships between the elements.

In this work, the encoder is trained from scratch. The authors used linear learnable parameters for the class and positional encoding. We tested a high-frequency signal for positional encoding like in the original paper of Vaswani et al. [VSP+17]. However, the results were inferior.
Reducing the number of multi-heads as much as possible helps to reduce the learnable parameters and make the network more computationally efficient. After some attempts, the authors find that the lowest number of multi-head usable with good results is 8.

Unlike a convolutional encoder, the transformer encoder does not use down-sampling operations, which preserve all the image details. Into the encoder, multiple transformer layers are applied on the cascade. Once the encoder has processed the tokens, they must be reassembled to complete the dense prediction. The authors won't describe the decoder since its only role is to rebuild the image by connecting the features at the specific pixels. Dosovitskiy et al. in [DBK+20] have a more in-depth description of the decoder architecture.

4.2 UNet

In this section, we describe the architecture of the UNet. Figure 4 illustrates that the name comes from the characteristic shape. It is based on the encoder/decoder model. This architecture is much simpler conceptually.

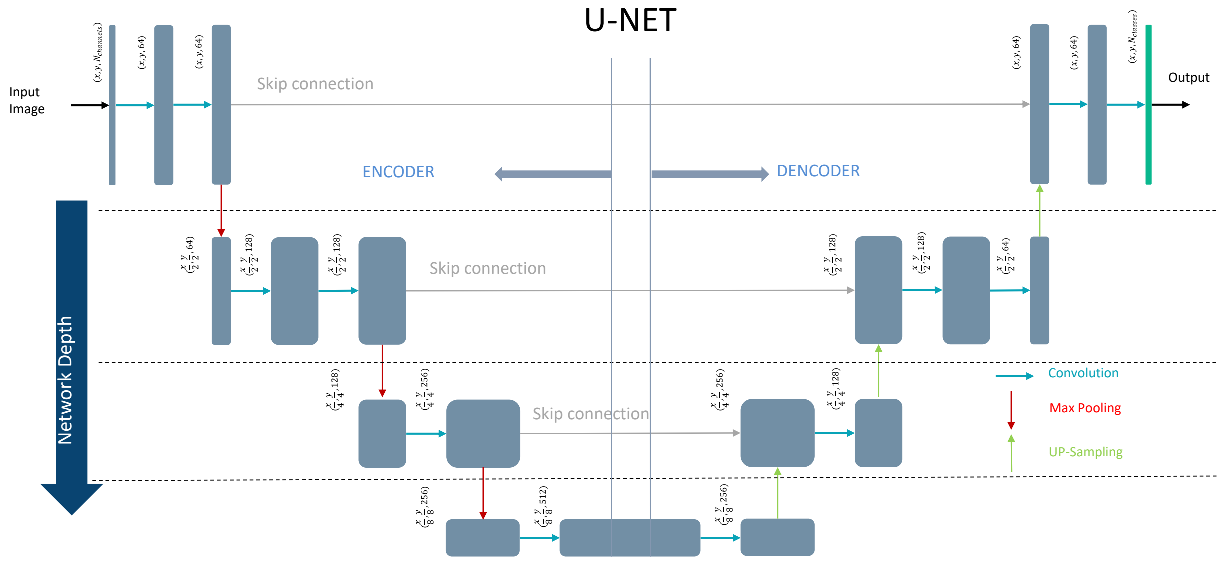
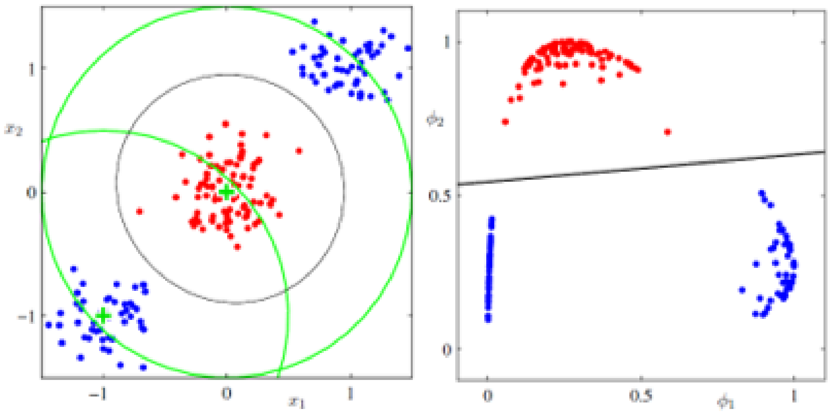

U-NET
Input Image
$(x, y, N_{channels})$
$(x, y, 64)$
$(x, y, 64)$
Skip connection
ENCODER
DENCODER
$(x, y, 64)$
$(x, y, 64)$
$(x, y, N_{classes})$
Output
Network Depth
$(\frac{x}{2}, \frac{y}{2}, 64)$
$(\frac{x}{2}, \frac{y}{2}, 128)$
$(\frac{x}{2}, \frac{y}{2}, 128)$
Skip connection
$(\frac{x}{2}, \frac{y}{2}, 128)$
$(\frac{x}{2}, \frac{y}{2}, 128)$
$(\frac{x}{2}, \frac{y}{2}, 64)$
$(\frac{x}{4}, \frac{y}{4}, 128)$
$(\frac{x}{4}, \frac{y}{4}, 256)$
Skip connection
$(\frac{x}{4}, \frac{y}{4}, 256)$
$(\frac{x}{4}, \frac{y}{4}, 128)$
Convolution
Max Pooling
UP-Sampling
$(\frac{x}{8}, \frac{y}{8}, 256)$
$(\frac{x}{8}, \frac{y}{8}, 512)$
$(\frac{x}{8}, \frac{y}{8}, 256)$

Fig. 4: Representation of UNet architecture

In the encoder, the image will pass through a series of 3x3 (unpadded) convolutional layers, each followed by a rectified linear unit (ReLU) and a 2x2 max-pooling layer to extract the image's features. At each downsampling step, the number of feature channels doubles. This process can be seen as non-linear transformations that project the image's features from the input space into the feature space, as illustrated in Figure 5. The main advantage of this transformation is to linearly classify the different classes using a score function that maps the raw data to class scores and a loss function that quantifies the agreement between the predicted scores and the ground truth labels. However, these non-linear operations also distort the image and change its resolution, with consequent loss of spatial information. After the encoder, the data goes through the decoder that recombines the information to obtain the original size image through successive upscaling convolutional layers [RFB15]. The features are assigned to the corresponding pixels, which a final softmax layer will classify. This process produces a labeled pixel-wise image with different values associated with safe and unsafe areas. Since this architecture is known to behave well with small datasets, we decided to train it from scratch. In Table 1, a more accurate description of the architecture adopted is shown.



(a) Input Space

(b) Feature space

Fig. 5: The left plot represents the input space and data samples from two classes (red and blue). The right plot shows the same data points but in the feature space after a non-linear transformation. The linear decision boundary is the black line, a curve in the input space [BN06].

Blocks	**Channels**
Double Conv.	3 → 64
Down1	64 →128
Down2	128 → 256
Down3	256 → 512
Down4	512 → 1024
Up1	1024 → 512
Up2	512 → 256
Up3	256 → 128
Up4	128 → 64
Classification Head	64 → 2

Blocks	**Layers**
Double Conv.	Conv2D BatchNorm2D ReLU Conv2D BatchNorm2D ReLU
Down	MaxPool2D Double Conv.
Up	Bilinear Up-sampling Double Conv.
Classification Head	Conv2D

Table 1: UNet architecture

5. PARAMETERS

In this section, we describe the parameters used to train the networks. For the ViT, we used a cross-entropy loss function with modified class weight to compensate for the unbalanced dataset. We noticed that the weight values significantly affect this model's performance; a good balance has been found after many training runs. The learning rate has been set to 3e-7, and the optimizer adopter is 'adam' [Zha18]. The batch size selected was 5, and the training lasted 200 epochs.

For the UNet, we continued to use the cross-entropy loss function. However, we noticed better results without class balancing. The learning rate was set to 1e-3, and the optimizer adopted is still 'adam'. The batch size selected was 5, and the training lasted 200 epochs.

6. METRICS

The following methods and terminology are employed to evaluate the test data:

- TP = true positive, pixels predicted as a target that is the target.
- TN = true negative, pixels predicted as background that is part of the background.

- FP = false positive, pixels predicted as a target that is part of the background.
- FN = false negative, pixels predicted as background that is the target.

At first, we consider using more traditional metrics for semantic segmentation problems. However, since we use full-resolution images for the target, we have, on average, less than 150 pixels per image, while in an entire image, the background pixels are more than 16 million. Metrics like intersection over union and precision vs. recall do not give insight into the model's performance since all the metrics are close to zero due to the large number of TN. Therefore, we used a confusion matrix, Figure 6. However, since showing the number of pixels belonging to each category would not be beneficial due to the large number of pixels involved, we decided to divide the TP and FN for the total number of pixels belonging to the target signal in the ground truth. Instead, the FP and TN will be divided by the total number of pixels belonging to the background. The values are presented as percentages.

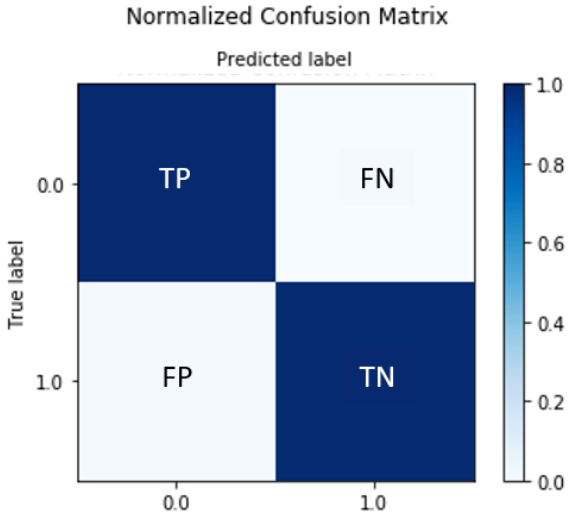


Fig. 6: Normalized confusion matrix.

7. RESULTS

The test dataset is made of full-resolution images, 4096x4096 pixels. The rest of the procedure is the same as the training set minus the cropping. This makes it harder to handle such large files, but it is closer to a realistic scenario since the user shouldn't know where the target is within the image. We create a test set at 3 levels of SNR following the equation 1, going from 1 to 4. Each SNR level has a test set of 240 images.

Figure 7 shows the confusion matrices for both models. We can notice that the UNet performs overall better than the ViT model. Interestingly, both models have very low FP, which means that very few pixels that belong to the background have been identified as targets. The high values of FN mean that pixels that should belong to the target have been considered as part of the background. Therefore, the noise covered the target signal. However, this can also be attributed to the lack of segmentation accuracy. For example, in the confusion matrix at SNR 2 for the UNet Figure 7, TP is 42% and FN is 58%. However, we must consider that the total number of target pixels normalizes those values. Therefore, 42% of the overall target pixels have been correctly classified, and 58% did not. We do not notice any significant change in performance in the SNR range between 2 and 4. However, both models can not classify the targets with SNR 1. To give some more perspective, the target signal level in pixel value at SNR 1 is around 260, where the average noise variance is around 42000. While the target signal level in pixel value at SNR 4 is 850.

To establish how common the FP are in the images, we computed the maximum distance error between the target location and the FP pixels, Figure 8. Unfortunately, for both models, the distribution is uniform, which means that the FP cannot be at any distance from the target. This can be caused by stars or other bright objects in the images, which suggests that the networks did not learn to identify only the object that moves between exposures.

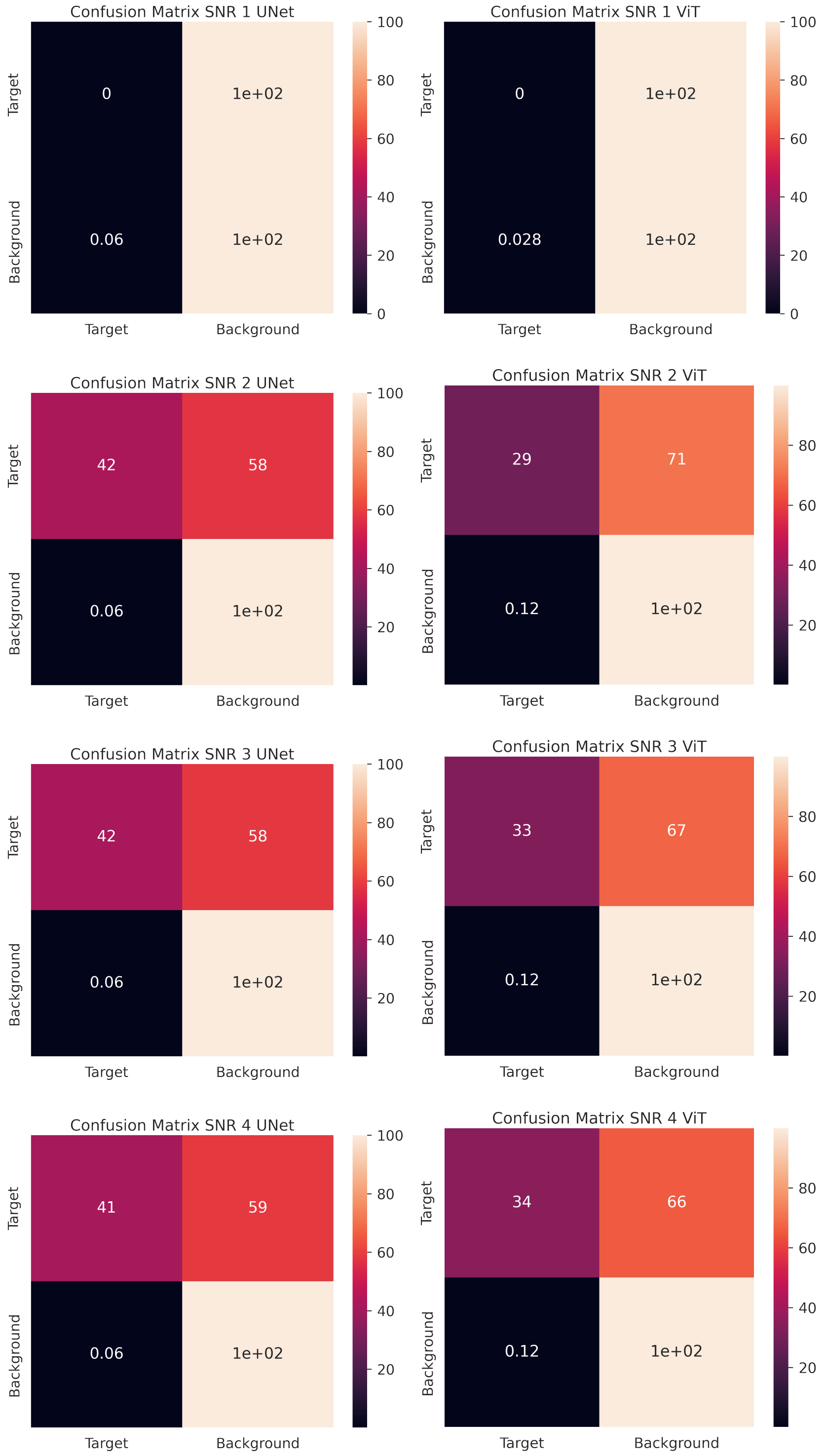


Fig. 7: In the first column are the confusion matrices for the UNet at different SNR levels. The second column contains the confusion matrices for the ViT at different SNR levels.

Even if the computational speed is not a priority for this kind of application and therefore the models are not optimized for it. It is interesting that the UNet, a much smaller network with fewer learning parameters, performs much faster than the ViT. The UNet can process a full-resolution image in ∼7 seconds, while the ViT takes ∼60 seconds for an

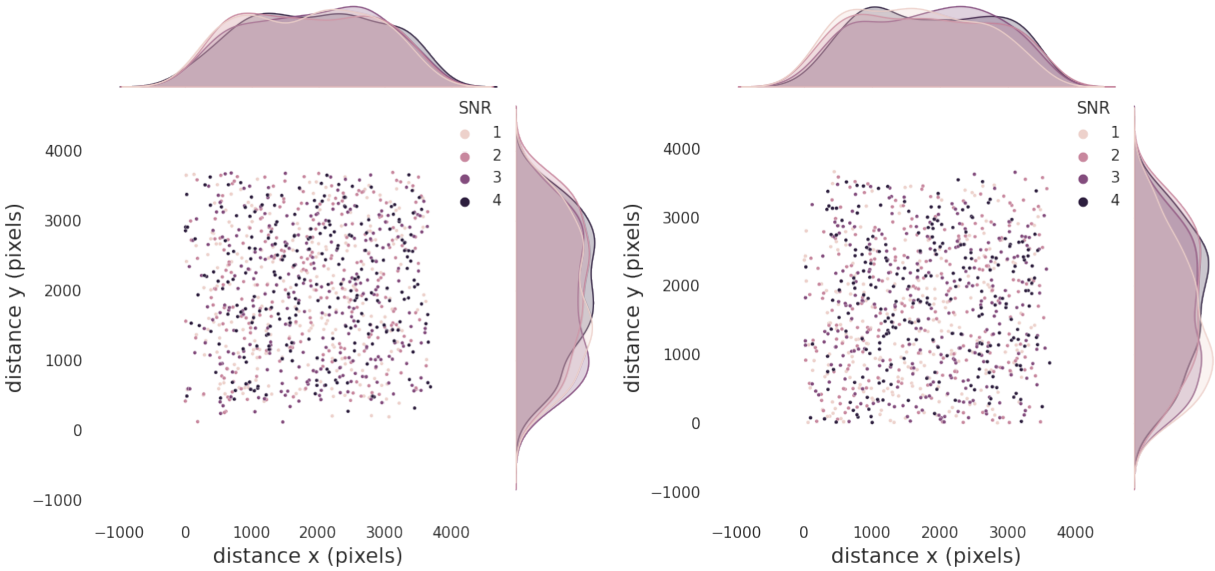


Fig. 8: On the left, the distribution of maximum distances of the FP with the target for the UNet is shown. The ViT model is on the right.

image.

8. CONCLUSIONS

In this work, we design two deep neural network architectures for object detection with low SNR images. The authors collected real astronomical images of the area around the moon and added a synthetic signal to simulate a target in the cislunar space. The network architectures adopted are based on two different encoder designs. One exploits the vision transformers, while the other uses a convolutional encoder. Both models were trained in a supervised fashion and were tested on full-resolution images.

From the results, we can see that the UNet performs better than the ViT. As mentioned in the manuscript, transformer-based models are powerful when trained on large datasets. On top of that, hyperparameter tuning affected the ViT performance much more than the UNet. However, we need to consider the usability of such tools. Both models proved good results in highlighting the area of the image where the target is, which satisfies the task required. Even considering the presence of FP, the post-processing method can filter out the objects that are stationary or have a motion that does not correlate with the desired targets.

We plan to optimize further and test the algorithm with realistic objects in cislunar space. This requires precise ephemerides and powerful telescopes. The Space4 center that we are part of has both the skills and the tools for such a campaign. The final goal of this work is to create a deep-learning-based tool that can be used in the real world to track these faint objects.

REFERENCES

[BKM+11] Michael R Blanton, Eyal Kazin, Demitri Muna, Benjamin A Weaver, and Adrian Price-Whelan. Improved background subtraction for the sloan digital sky survey images. *The Astronomical Journal*, 142(1):31, 2011.

[BN06] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

[BR22] Fateme Bahri and Nilanjan Ray. Dynamic background subtraction by generative neural networks. In *2022 18th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–8. IEEE, 2022.

[CFR+22] Tanner Campbell, Roberto Furfaro, Vishnu Reddy, Adam Battle, Peter Birtwhistle, Tyler Linder, Scott Tucker, and Neil Pearson. Bayesian approach to light curve inversion of 2020 so. *The Journal of the Astronautical Sciences*, 69(1):95–119, 2022.

[DBK+20] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[DVCDLM22] Andrea De Vittori, Riccardo Cipollone, Pierluigi Di Lizia, and Mauro Massari. Real-time space object tracklet extraction from telescope survey images with machine learning. *Astrodynamics*, 6(2):205–218, 2022.

[FHDB21] Carolin Frueh, Kathleen Howell, Kyle J DeMars, and Surabhi Bhadauria. Cislunar space situational awareness. In *31st AIAA/AAS Space Flight Mechanics Meeting*, pages 6–7, 2021.

[FRCG21] Roberto Furfaro, Vishnu Reddy, Tanner Campbell, and Bill Gray. Tracking objects in cislunar space: the chang'e 5 case. In *AMOS Conf. Proc*, 2021.

[GDS+21] Luca Ghilardi, Andrea D'Ambrosio, Andrea Scorsoglio, Roberto Furfaro, and Fabio Curti. Image-based lunar landing hazard detection via deep learning. In *Proceedings of the 31st AAS/AIAA Space Flight Mechanics Meeting, Virtual*, pages 1–4, 2021.

[GSF22] Luca Ghilardi, Andrea Scorsoglio, and Roberto Furfaro. Iss monocular depth estimation via vision transformer. In *International Conference on Applied Intelligence and Informatics*, pages 167–181. Springer, 2022.

[JPLX22] Kai Jiang, Peng Peng, Youzao Lian, and Weisheng Xu. The encoding method of position embeddings in vision transformer. *Journal of Visual Communication and Image Representation*, 89:103664, 2022.

[KNH+22] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey. *ACM computing surveys (CSUR)*, 54(10s):1–41, 2022.

[PS15] Adam Popowicz and Bogdan Smolka. A method of complex background estimation in astronomical images. *Monthly Notices of the Royal Astronomical Society*, 452(1):809–823, 2015.

[RBC+21] Vishnu Reddy, Adam Battle, Tanner Campbell, Paul Chodas, Al Conrad, Dan Engelhart, James Frith, Roberto Furfaro, Davide Farnocchia, Ryan Hoffmann, et al. Spectral characterization of 2020 so. In *Advanced Maui Optical and Space Surveillance Technologies Conference (AMOS)*, 2021.

[RFB15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.

[SDG+20] Andrea Scorsoglio, Andrea D'Ambrosio, Luca Ghilardi, Roberto Furfaro, Brian Gaudet, Richard Linares, and Fabio Curti. Safe lunar landing via images: A reinforcement meta-learning application to autonomous hazard avoidance and landing. In *Proceedings of the 2020 AAS/AIAA Astrodynamics Specialist Conference, Virtual*, pages 9–12, 2020.

[TSP01] JTL Thong, KS Sim, and JCH Phang. Single-image signal-to-noise ratio estimation. *Scanning*, 23(5):328–336, 2001.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[Zha18] Zijun Zhang. Improved adam optimizer for deep neural networks. In *2018 IEEE/ACM 26th international symposium on quality of service (IWQoS)*, pages 1–2. Ieee, 2018.