HIGH SCHOOL EDITION
Journal of Student Research
Volume 14 Issue 1 (2025)

Reduced Image Classes in Modified U-Net for Mars Rover Navigation

Roy Qiu[1] and Victoria Lloyd[#]

[1]Fraser Heights Secondary, Canada
[#]Advisor

ABSTRACT

Rover navigation currently relies on algorithms to automatically determine their path. This is because the distance from Earth to Mars means that real time communication is impossible. Current navigation algorithms have difficulties in identifying terrain, causing problems such as becoming stuck in soft terrain. Furthermore, the available computation power and memory are limited on a rover. This paper presents both a modified U-Net model to identify parts of the terrain and combining multiple classes to have less output classes. The proposed method was to combine classes like soil and bedrock into more generalized classes, like traversable and untraversable, to reduce memory usage and needed computational power. Combining the classes shows that a model can be trained faster, and in some cases even improve. Testing this method on low resolution images has shown improved results in testing. After training, a three-class model is able to yield a higher mIoU of 0.4583 on a test set compared to the full five-class model, which achieved 0.3451. This method is non-specific to U-Net and can be applied to many different models. Combining this method with other models and larger datasets during training could be an option of improving the accuracy of models running on less processing power, allowing for use on platforms such as Mars rovers.

Introduction

Exploration on other planets is difficult to perform with humans due to the vast distances causing real-time communication to be impossible. NASA's Mars rovers, for example, must have programs to detect when its path is unsafe. The conditions of a pre-planned route might be unknown until the rover is able to analyze its path, or conditions change. A full plan can only be uploaded to the rover approximately every one in three days. Therefore, onboard algorithms are needed for exploration and to supply the rover with autonomy during its exploration. Rovers also have limited processing power immediately available. [2] Much of the terrain needs to be assessed by operators, and encountering unexpected terrain is a large issue as it can immobilize a rover entirely. [3] There exist many similar projects on Earth. Self-driving cars utilize image segmentation to identify different elements of the road, such as painted lines and signs. [4] Multiple datasets and benchmarks have been made available for this purpose. [5] NASA has released a similar dataset for Martian images. The AI4Mars dataset contains images from the Spirit, Opportunity, and Curiosity rovers, labeled by a crowdsourcing effort. [1] The purpose of this database is to provide data for classification of different terrain to determine whether it is safe to drive on.

This paper presents a modified U-Net architecture for image segmentation. U-Net was originally a biomedical image segmentation model, but it has high success rates in other image segmentation applications and is very well-known. [6] This makes it a good starting point to test various methods. The AI4Mars dataset contains grayscale images of Martian terrain, along with labeled masks. The dataset contains five categories for classification of Martian terrain. [1] This model aims to accurately predict the category of terrain for each image and create an appropriate mask.

Background

Some previous methods have already been used on the dataset. When publishing the AI4Mars dataset, Michael Swan et al. used a DeepLabv3 model pre-trained on ImageNet for experimentation, as it was well-maintained, and its codebase is relatively mature. Their model was trained using machines with two NVIDIA GeForce GTX TITAN X or two Tesla P100 GPUs and done over the MSL labels (labels made by the Mars Science Laboratory). Naively using this approach achieves high accuracy, around 94.97% on an unmodified random label set. This method also can misclassify big rocks as soil, which may cause the rover to attempt to drive over undrivable terrain and use relatively larger image sizes than the model presented in this paper. [1]

Lihang Feng et al. proposes a different model called Mobile-DeepRFB. The model was based on DeepLab3+, the same model that Michael Swan et al. used in their naive approach. However, they modified the model to use the backbone from MobileNetV3 to reduce the number of parameters that the model needed, thereby also reducing processing power. They also added a module called the Receptive field Block, which enhances features that are extracted by their model. Their model also takes 512x512 sized images. The model mIoU reaches 71.10%. [7][8]

Steven Kay et al. compares two different models and their metrics in dealing with this task, namely U-Net and DeepLab3+. Each model had two different configurations tested. They also attempted to use SemanticStyleGAN to produce additional images, which was also added to the training dataset in two separate experiments to test how this would affect the performance. In the end, DeepLab3+ did end up outperforming the U-Net model. Notably, augmentations by GaN increased the model mIoU as well. Our model presents a different method, which could be used in models like this one to improve results. [9]

Dataset

Table 1. Makeup of the used data.

Category	Soil	Bedrock	Sand	Big Rock	NULL
Percent (approx.)	20.10%	27.05%	6.27%	0.49%	43.20%

Table 2. Makeup of the data with reduced classes.

Category	Traversable	Untraversable	NULL
Percent (approx.)	47.15%	5.76%	43.20%

The AI4Mars dataset contains some different data. First, it contains two separate sets for training and testing. The training set contains cleaned images that had some pre-processing to add masks. The testing labels were made from volunteers. There are different labels made from a single person, two people agreeing, or three people agreeing. The labels had five categories: soil, bedrock, sand, big rocks, and a null class. Finally, each label has separate masks for range and to indicate which parts of the image were of the rover itself, and not the terrain. Each input image was taken from Curiosity, Spirit, and Opportunity rovers, with Curiosity operated by MSL and Spirit and Opportunity operated by MER. Each of the images were taken from the NAVCAM mounted on the rover, along with additional data from other instruments. [1] This provides greyscale images that are 1024x1024 pixels in size. For this paper, only NAVCAM data from the MSL was used due to constraints discussed later.

It should be noted two experiments were carried out. One was with the full five classes provided by the labels, and the other was with three classes: Traversable, Un-traversable, and the null class. The reduction of classes was done to test reducing computation and increasing accuracy.

Preprocessing for the images was already completed in the training images, so little additional preprocessing was done. The images were resized to 256x256 for training to reduce computational power.

For the experiment with three classes, the soil and bedrock were classified as “traversable,” while the sand and big rocks were considered “untraversable.” This was because of multiple instances where the Mars rovers were stuck in pits of sand and would become trapped. [3] The general composition of each image differs greatly, with some images only containing null class, while others would be completely dominated by different classes. The total composition of the training images is shown in the above table, where some classes are more prominent than others. Notably, most of the images are composed of null class, while big rock accounts for less than 1% of the dataset. The dominance of null is expected, since each image consists of sky or faraway terrain, neither of which is actually defined and therefore falls under the null category. The lack of big rock examples, while not exactly unexpected, may have reduced the accuracy of models trained using this dataset.

Methodology

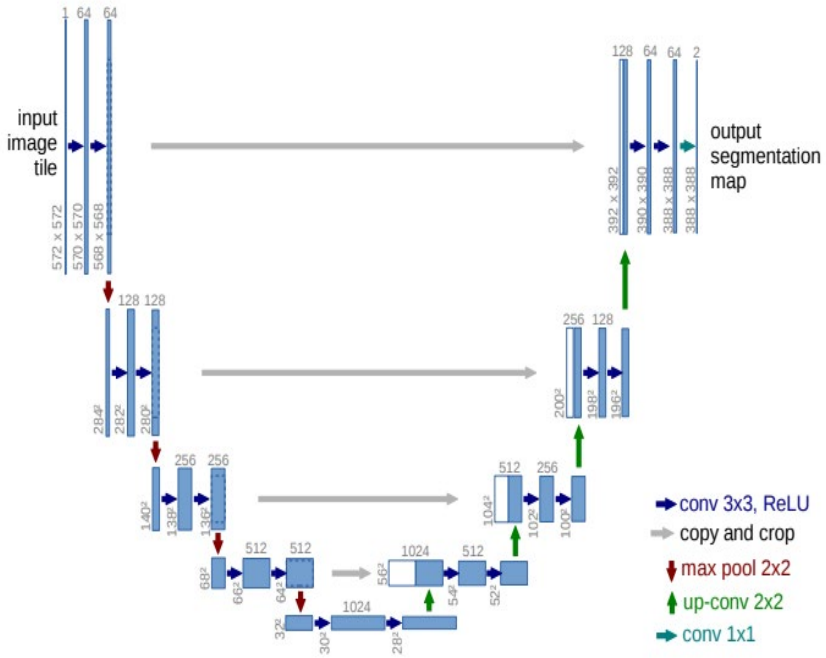

Figure 1. A picture of the unmodified U-Net architecture [5]

The overarching structure resembles the original U-Net. [5] This model also contains expanding and contracting layers for encoding and decoding. Skip connections are also preserved, where images from the encoder sections are concatenated with the decoder input to preserve details. Each convolutional layer also has 3x3 kernels, with strides of size 2, with two notable exceptions. All encoders feed into max-pooling layers, and decoders feed into upsampling layers.

The modifications made to the model are changes to padding in the layers, the number of filters, and an extra layer at the start to act essentially as a “pre-processing” layer. The original U-Net had around four encoders and four decoders. Each encoder and decoder were specified to have zero padding, which caused each encoder/decoder to have

an output which is 4 pixels smaller in both width and height than the input. For this model, each convolutional layer was given padding which allows the input and output of the model to have the exact same shape. The model contains four encoders, each ending in a 2x2 pooling layer. This means four decoders, and there are two convolutional layers with no pooling between the encoders and decoders. The last layer of the model was a 1x1 convolutional layer. Each convolutional layer had ReLU activation, except for the last layer which used Softmax to optimize for classification. Two dropout layers were added between the second and third encoders, and the second and third decoders to prevent overfitting of the model. [10] Both models were trained on a NVIDIA Tesla T4 GPU.

To reduce resource usage, instead of removing layers from the model, it was decided to try and remove extra classes from the output. The reduction of classes from five to three was done by a find and replace, and this was done as preprocessing of the dataset. It should be noted that only 1000 images from the training dataset were used for training due to limitations in available memory. This method is actually non-specific to this model and can be applied to different models. By targeting the specific goal of the rover, the data can be processed faster while hopefully achieving the same goal.

Results

Table 3. The final model metrics from both training and validation. Includes both three-class and five-class models.

Metrics	mIoU	Dice	Precision	Recall	Accuracy
Three-label (training)	0.6020	0.2350	0.8142	0.8068	0.8097
Three-label (test)	0.4583	0.3750	0.6629	0.6520	0.6663
Five-label (training)	0.6777	0.1843	0.9134	0.8879	0.8948
Five-label (testing)	0.3451	0.3967	0.6629	0.6385	0.6567

Comparing both the models, they do perform similarly, but there are key differences. The full five-class model required much more epochs (around 200 epochs) and much more tuning to reach 80% accuracy whereas the smaller three-class model only needed 100 epochs to reach higher accuracy on the test set. Also notably, performance for each model drops in the training set compared to the testing set. The final three-class model was trained with 100 epochs while the final five-label model was trained with 200 epochs.

For the full five-class model, notably, IoU is fairly low, and there was some mischaracterization of elements like soil and bedrock. It also has a high tendency to group large areas as one category. In Figure 3, the true mask lists most of the image as NULL, but the model will classify a huge portion as different classes. The most likely cause for this is the limited information that the model was given, namely the low resolution of the images.

There were further concerns due to the small size of the dataset used to train it. Because of constant issues with aforementioned memory, the model's output was reduced to three-classes to see if it would help increase overall accuracy.

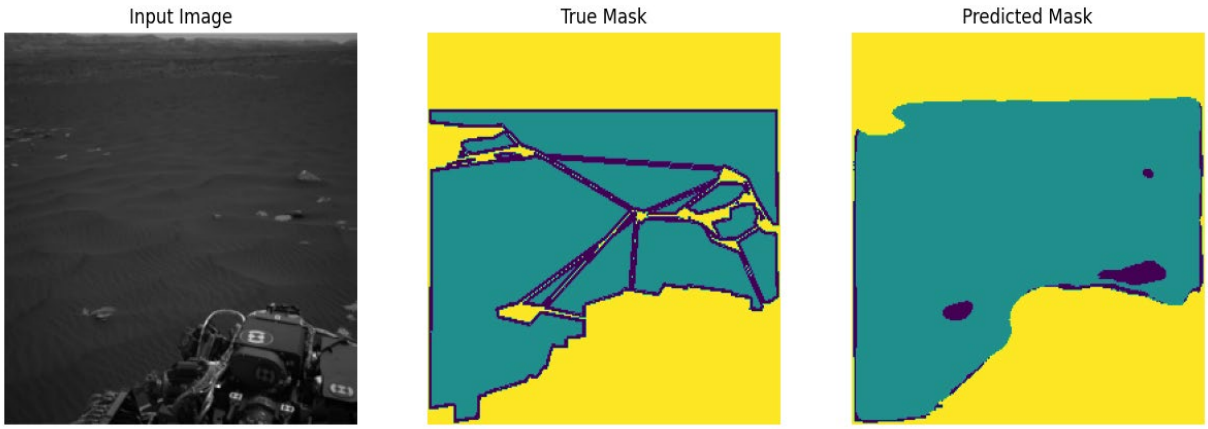

Figure 2. Example of three-class model output.

Figure 3. Example of five-class model output.

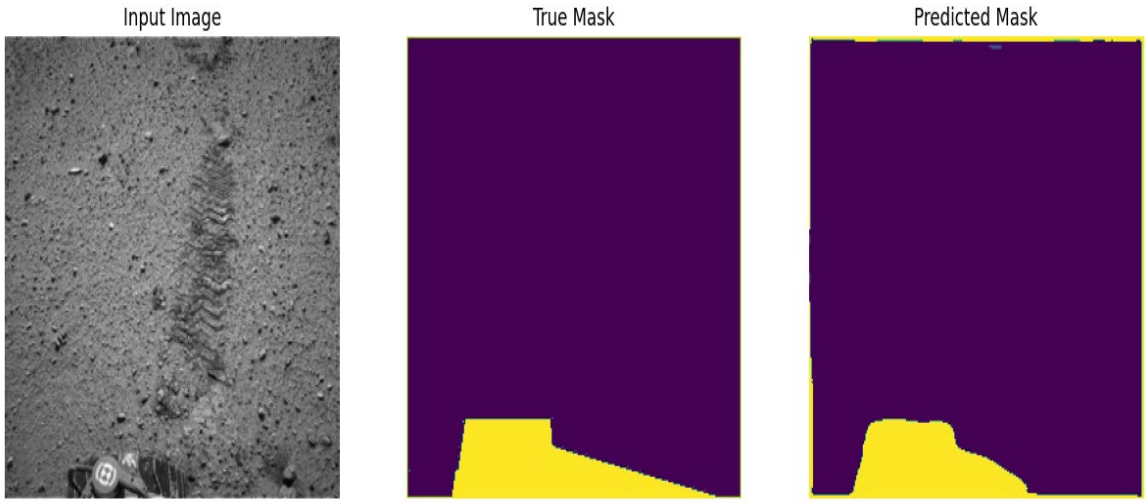

Figure 4. Example of improved five-class model output.

The model was trained using the MSL provided training dataset, while all results were verified with the unprocessed volunteer dataset. Around 300 images were used for verification, with the labels requiring three volunteers to agree on the label. The number of images used for training and verification are the same.

The three-class model is more consistent between training and testing, and it also seems to perform better overall for traversable and untraversable terrain during testing. It should be noted that the full five-class model seems to achieve higher precision and recall during training.

Table 4. Class-specific IoU for the five-class model.

Category	Soil	Bedrock	Sand	Big Rock	NULL
Training IoU	0.6993	0.8825	0.5632	0.3808	0.8626
Testing IoU	0.4655	0.2870	0.3916	0.0035	0.5709

Table 5. Class-specific IoU for the three-class model.

Category	Traversable	Untraversable	NULL
Training IoU	0.7106	0.3986	0.6968
Testing IoU	0.4392	0.3694	0.5663

Discussion

The results show that the reduction of classes seems to be effective. As mentioned, the three-class model is able to achieve either similar or better results from half the number of epochs. Both models were limited by the relatively small dataset used to train them, and the relatively low resolution of the images (256x256). There are some results that should be noted. For instance, the IoU for classes like Big Rock are significantly lower compared to NULL. This could be accounted for by the fact that the NULL class makes up a much more significant percentage of the dataset overall, which causes the model to be able to recognize those classes better. Coincidentally, by reducing the number of sets, this actually decreases this effect since smaller classes are grouped up with larger classes, thus combining the effective amount of training data.

There is also a great difference between the training and testing metrics, implying overfitting. However, this should have been countered by the Dropout layers included in both models. This problem is likely another complication from the small dataset. There are many future improvements that could be applied to the model. First, training a model with a larger and more diverse dataset will likely give a better result. Second, increasing the resolution for the model input would also likely improve the result. Finally, being able to test a trained model on rover-specific hardware would also likely give a better idea of how this model would perform in the real world.

Compared to methods like the one proposed by Lihang Feng et al., this model can reach similar mIoU metrics on the training dataset while working with images of a lower resolution. Furthermore, reducing the number of classes can be applied to a variety of models, and can be combined with the lightweight model they have proposed.

This method does have a drawback, which is the lack of details. Reducing the number of classes means that the rover cannot immediately discern between different types of terrain. The previous proposed methods, such as the lightweight model proposed by Feng et al. and Swan et al., all have results at higher resolution and with much more

information. However, for applications like pathfinding, being able to discern potentially traversable areas is useful data for pathfinding algorithms. The rover is unable to use high power servers, so this method could still prove helpful. Applying this method to other models outlined in previous papers could help create an even more effective model that can be used on the mars rover.

Conclusion

This paper proposed combining a modified U-Net model and reducing the number of output classes for AI4Mars classification. Using grayscale images from rover's NAVCAM, the model can generally give an indicator whether an area is traversable or untraversable. By reducing the number of classes, it helps reduce the number of epochs needed to train a model and helps it with smaller and lower resolution datasets. The results presented were limited by the available memory, resulting in a less accurate model. By training this model on a larger dataset or combining this method with other models may allow for useful results in navigation. The drawback to this method is the lack of other information, like what type of terrain that a certain area is. But knowing traversability still can be useful information for navigation.

Acknowledgments

This paper was written as part of the Inspiritai Mentors program, and much support was given by them for this paper.

References

Swan, M. R., Atha, D., Leopold, H. A., Gildner, M., Oij, S., Chiu, C., & Ono, M. (2021). AI4MARS: A dataset for terrain-aware autonomous driving on Mars. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. https://doi.org/10.1109/cvprw53098.2021.00226

Verma, V., Maimone, M. W., Gaines, D. M., Francis, R., Estlin, T. A., Kuhn, S. R., Rabideau, G. R., Chien, S. A., McHenry, M. M., Graser, E. J., Rankin, A. L., & Thiel, E. R. (2023). Autonomous robotics is driving Perseverance rover's progress on Mars. Science robotics, 8(80), eadi3099. https://doi.org/10.1126/scirobotics.adi3099

Rankin, A., Maimone, M., Biesiadecki, J., Patel, N., Levine, D., & Toupet, O. (2020). Driving curiosity: Mars rover mobility trends during the first seven years. *2020 IEEE Aerospace Conference*. https://doi.org/10.1109/aero47225.2020.9172469

Lee, D.-H., & Liu, J.-L. (2023). Multi-task UNet architecture for end-to-end autonomous driving. arXiv [Cs.LG]. Retrieved from http://arxiv.org/abs/2112.08967

Geiger, A., Lenz, P., & Urtasun, R. (2012). Are we ready for autonomous driving? The Kitti Vision Benchmark Suite. *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 3354–3361. https://doi.org/10.1109/cvpr.2012.6248074

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. *Lecture Notes in Computer Science*, 234–241. https://doi.org/10.1007/978-3-319-24574-4_28

Feng, L., Wang, S., Wang, D., Xiong, P., Xie, J., Hu, Y., Zhang, M., Wu, E. Q., & Song, A. (2023). Mobile-deeprfb: A lightweight terrain classifier for automatic Mars Rover Navigation. *IEEE Transactions on Automation Science and Engineering*, 1–10. https://doi.org/10.1109/tase.2023.3340190

Liu, S., Huang, D., & Wang, Y. (2018). Receptive field block net for accurate and fast object detection. *Lecture Notes in Computer Science*, 404–419. https://doi.org/10.1007/978-3-030-01252-6_24

Kay, S., Quoos, M., Field, R., Prokopczyk, M. J., De Benedetti, M., Mohammad, F., ... & Ntagiou, E. V. (2023). AI-enabled Computer Vision Framework for Automated Knowledge Extraction in Planetary Rover Operations. Submitted to ASTRA.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research, 15(1), 1929-1958.