EXPERIMENTS IN SEGMENTING MARS CRATERS USING CONVOLUTIONAL NEURAL NETWORKS

*D. M. DeLatte [1], S. T. Crites [2], N. Guttenberg [3], E. J. Tasker [2], T. Yairi [1]

[1] University of Tokyo, 7-3-1 Hongō, Bunkyō-ku, Tōkyō-to 113-8654, Japan, E-mail: delatte@ailab.t.u-tokyo.ac.jp

[2] Institute of Space and Astronautical Science / Japan Aerospace Exploration Agency, 3-1-1 Yoshinodai, Chuo-ku, Sagamihara City, Kanagawa Prefecture, 252-5210, Japan

[3] Araya, Toranomon 15 Mori Building 2F 2-8-10 Toranomon, Minato-ku Tokyo 105-0001, Japan

ABSTRACT

Planetary scientists use crater counting to age date regions of planetary bodies. Historically this task has been done by hand, but the publication of annotation datasets makes it possible to train algorithms to replicate this work. The authors present the series of experiments that led to the development of an algorithm that creates a segmentation map of the locations of craters. The algorithm is trained using annotations of Martian craters from 2-32 km and images from the THEMIS Daytime IR camera. Results from a test campaign comparing variations of a UNet-inspired neural network architecture are shared. While this is work in progress, the results so far are encouraging enough to suggest that with further optimization, neural nets may augment traditional approaches to the crater identification problem.

1 INTRODUCTION

Crater counting, a rite of passage for planetary geologists, is used to age date regions of planetary bodies. Surface ages are determined by counting the number of craters of various sizes in a region and comparing those counts to expected accumulation from a known production function based on expected meteorite impact rate. Radiometric dating of returned lunar samples anchor these chronologies to absolute ages [1]. Additional assumptions are needed to obtain absolute ages for Mars and other planets since no samples have been returned (to date). To support this method, legions of citizen scientists, graduate students, and experts have labeled – by hand – the characteristics of hundreds of thousands of craters.

Although researchers can glean insights by making direct observations while counting craters and general familiarity with the surface of the target celestial body is valuable, counting craters is a tedious, repetitive task that could be automated. While many methods for crater counting have been proposed [2] [3] [4], none of these methods has been definitive enough to replace hand labelling data. Moreover, [3] and [4] require information beyond image data, such as digital elevation.

With the recent publication of large crater counting datasets and increased accessibility of machine learning methods and computational hardware, this historically tedious task can potentially be automated. One challenge is that craters overlap and vary in size, depth, and visibility. A potential way to unpick this complex terrain is to use neural networks. Unlike other automated techniques which require a set of human-designed features, neural networks find their own, and complex pattern recognition criteria can be developed. In this study, the authors explore the potential of using a neural network to identify craters.

Here, the authors explore a machine learning technique called segmentation [5], where an image map is created of all the detected objects in the image. Unlike some of the methods mentioned above, segmentation relies only on visual images and a training set of existing annotations. Once trained, the machine learning algorithm detects craters in new images. The source of annotations, Robbins & Hynek's Mars Crater Dataset [6], is chosen because of the detail and completeness of the dataset. The dataset consists of the latitude, longitude, and diameters of all craters larger than 1 km on Mars. (Many other fields are included in the full dataset. The authors intend to explore the use of these in future work.) The source of image data is the Mars daytime infrared images from the NASA THEMIS mission [7]. These images were also used by [6].

These experiments begin a study of Convolutional Neural Network (CNN) architectures applied to the crater detection and extraction problem using segmentation. A series of tests systematically explores hyperparameter choices and target strategies. CNNs have been applied to a wide variety of machine learning tasks for images. Image segmentation, identifying a group of pixels as part of a region of interest, was explored in [8] using the UNet architecture, the design that motivated the network used in this work.

This work fits into a collection of efforts by crater counters and machine learning researchers to automate the detection of craters. Recent work

has explored how convolutional neural networks might aid in the classification or segmentation of craters. Some researchers have independently used a UNet to find lunar craters in digital elevation data [9] (preprint), but to the author's knowledge our work is the first use of a UNet to find Martian craters in THEMIS thermal infrared data.

This paper's primary contribution is introducing a Symmetric UNet (inspired by the UNet architecture [8]) and training that CNN-based network to detect craters in the THEMIS dataset [7] using Mars crater annotations [6]. Experimental results of a synthetic dataset and a crater dataset using two different types of targets (solid circles vs. outline circle targets) are presented. In Figure 1, the middle image is presented for the reader for comparison to the right Prediction image.

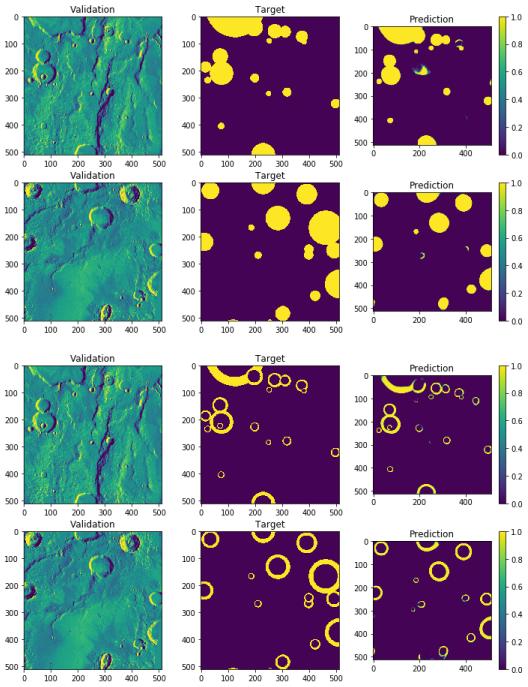

Figure 1: Example validation results with filed circle and outline targets

This paper discusses methods in Section 2, including data (2.1), segmentation network (2.2), and experiment hardware (2.3). Results from tests using the three datasets are discussed in Section 3, including a simple synthetic dataset (3.1), crater data with solid circle targets (3.2), and crater data with outline targets (3.3). The discussion and conclusion are in Section 4 and Section 5, respectively.

2 METHODS

This section describes the crater dataset used, details the segmentation UNet architecture, and comments on the hardware needed to run these experiments.

2.1 Data

A suitable annotation dataset is needed to train a convolutional neural network to recognize and extract craters. The Robbins & Hynek [6] set is chosen because it is the largest, most complete dataset available for Martian craters. Craters down to 1 km are identified, all counted by hand. Each crater example includes latitude, longitude, and diameter for each identified crater. For the purposes of this study, circular representations of craters are used and all crater annotations are included, regardless of crater degradation level. Generalizability to other crater annotation datasets will require further study. Since a single annotation set is used, this model finds craters specifically in the style of Robbins & Hynek.

THEMIS Daytime IR images (Figure 2) of Mars are chosen because of the dataset's high resolution and their use by [6] in creating the annotation database. Several other datasets considered are poorly documented or have projection issues. The THEMIS image tiles are each 30º per side and that uniform predictability made them very useful for this purpose. Unsurprisingly, when the Robbins annotations are applied to the images, there is excellent visual match between them.

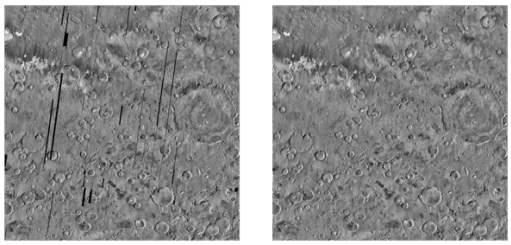

Figure 2: THEMIS image: thm_dir_N-30_030

The downside of using the THEMIS images is the absence of data in some areas, seen as the black streaks in Figure 2. Initially there was concern that this would be problematic for the network, particularly in cases where a streak cut through a crater. To mitigate the missing pixel issue, before training, all black pixels in the image are replaced with the average value of the tile. This preprocessing step keeps the missing data from becoming a hard edge and prevents drastic changes to the weights during training.

For testing purposes and to avoid projection issues in the poles of the THEMIS dataset, the tested imagery is constrained to ±30º latitude.

This subset of the THEMIS dataset contains 24 tiles of 30º by 30º. Each is 7680 x 7680 pixels. Also, a subset of the annotations available (craters sized 2-32 km) is chosen for these initial tests.

Of the 24 tiles available for testing, six are used in training and three for validation. The tiles used for training and validation are selected randomly (with no overlap). For training, the data tiles (training, validation) and generated target images are split into 512x512 pixel sub-images before being fed through the network.

2.2 Segmentation Network

Machine learning encompasses a wide range of techniques, including classification, segmentation, and localization. This work focuses on pixel-wise segmentation of craters. The goal is to pass an entire tile of the THEMIS dataset to the network and have the segmentation map returned.

Image classification problems may address the question "is the object in this region a X?" repeatedly. Classification algorithms can be binary, for example, "is this a cat?" or multi-modal: "Is this a cat? Is this a dog? Is this a bird?" This approach may require a preprocessing step that localizes to a region, for example by using a region proposal algorithm.

Segmentation differs from standard image classification in that its result effectively separates the objects of interest from the background. On the smallest scale (one pixel) and for a one-object detection, "segmentation" amounts to a binary classification of each pixel. Semantic segmentation does not distinguish between examples of the object while instance segmentation identifies each example object separately [5].

While both classification and segmentation have tremendous uses in image processing, for this application, object segmentation is most appropriate for automating crater counting. Humans who manually count craters consider many details when classifying a true crater. Also, segmentation methods are more similar, from a data processing perspective, to what humans would do: take a whole image, identify the craters, and determine the number and locations of those craters inside the region of interest.

The authors draw significant inspiration from [8] but want to preserve the input size in the final segmentation map. Experiments led to choosing a "default" Symmetric UNet (Figure 3) for the purposes of these experiments.

The "default" network has the following structure: ReLU activations on all layers, convolutional kernel size of 3x3, and filter values (nodes in each layer) from this array: [16, 24, 32, 48, 64, 96, 128, 128, 128, 96, 64, 48, 32, 24, 16]. It is coded using Python and Keras with a Tensorflow backend.

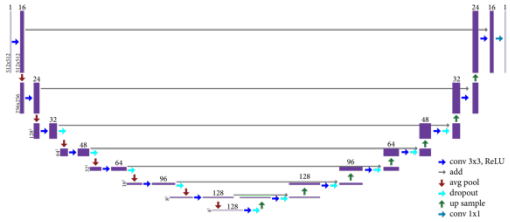

Figure 3: Symmetric UNet structure, visualized in the style of [8]

Each layer going "down" consists of seven rounds of these (names of the Keras layers are used below, see [10] for definitions):

- Conv2D layer with the appropriate filter value from the array above, the kernels size, "same" padding, and ReLU activation.
- Dropout (no dropout for the first of the "down" sections, but increasing from 0.2 to 0.5 at the bottom of the "U")
- AveragePooling2D with a pool size of 2

The bottom layer is Conv2D and Dropout (without AveragePooling2D).

Going "up" consists of:

- UpSampling2D, size 2
- Conv2D
- Dropout
- Add, adding the result of the previous dropout with the correspondingly sized Conv2D layer from going "down"

The final layer is another Conv2D with a kernel size of 1, "same" padding, and sigmoid activation.

2.3 Hardware

The two main datasets in this set of experiments are the simple synthetic dataset and the THEMIS dataset. The intention of the synthetic dataset is to rapidly test and debug the UNet architecture and determine what effects various hyperparameters have on the network. These tests could run on a MacBook Air laptop CPU in about 10 seconds per epoch (although run in less than a second on a Graphical Processing Units), making these experiments more portable.

The hardware used for the tests is a Deep Learning Box with four of these GPUs: 8GB

NVIDIA GeForce GTX 1080. Four tests can run simultaneously. Each test runs on one GPU.

3 RESULTS

This section describes experiments aimed at implementing the network then tuning it to improve performance. Promising initial progress is made for automatic crater detection.

Initial segmentation results (Figure 4) were not distinctive and consistent enough to declare victory on the crater counting problem, so a series of network tests using synthetic and real crater data improved understanding of the strengths and weaknesses of the model. Through these diagnostic tests, an error in implementation was discovered and results post that discovery improved significantly. (Initially the Keras default linear activation was used between layers instead of ReLU.) The experiments in Section 3.1, 3.2, and 3.3 are all from tuning the network after the implementation errors were diagnosed.

In the early results (Figure 4), the rough location of many craters was identified, but their shape and coverage were not clearly marked in the prediction map.

Note: for all results reported in this paper, the "Validation" image is an example that was unseen by the network during training, although comparison to the Target is calculated in the Validation loss or accuracy (i.e., Figure 7). The middle "Target" image is included for human comparison with the "Prediction" image that the network produces when given the Validation image.

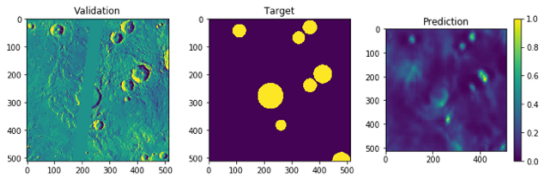

Figure 4: Early crater detection results were poor

Considerations such as overlapping craters and craters within craters make the problem challenging. Additionally, missing data in the THEMIS images (shown as empty pixels) reduced the effectiveness and accuracy of the results. To pick apart the problem and identify the major contributing factors, a series of experiments with synthetic data help determine the major causes of inaccuracies in the initial segmentation results.

The following sections detail results from each of the test campaigns. Each campaign includes the exact same set of architecture tests. Section 3.1 contains a detailed description of the architectures undergoing tests and details the synthetic data tests (each test is labeled with S-#). The dataset includes randomly generated gray dots (various shades) on a black background. (This is shown as a purple to yellow scale in the images but represents a single grayscale channel.) The target set is generated to be solid filled circles in the same places. Section 3.2 and 3.3 are tests on real crater data tiles. The same training data is used for each of the tests. The difference is in the "target" or goal prediction. Section 3.2 includes tests using a filled circle while Section 3.3 targets are an outline.

Note: For all tests in this paper, all network architecture numbers are the same between datasets. For example, tests S-3, C-3, and O-3 have the exact same network architecture (same convolutional kernel size, filters, activation, layer types, etc.). The letters S, C, O refer to the different data-target pairs each was trained on (synthetic, craters and filled circle targets, craters and outline targets, respectively).

Accuracy is the number of correctly predicted pixels divided by the total number of pixels (per the annotations). Loss is an aggregated measure of uncertainty in those predictions. The reported loss uses the binary cross-entropy function.

3.1 Network Testing – Synthetic Data

For this sequence of tests, some tests investigated repeatability of a single architecture while others tested the impact of various architecture changes.

For each of these tests, the activation type is a rectified linear unit (ReLU). Convolutional kernel sizes of 3x3, 7x7, and 11x11 were tested. Using the default Symmetric UNet as a baseline, the filter numbers were varied as either half the original value or double the original value. (Other multiples were tested, but caused a resource exhausted error on the test hardware.) A filter is a node in a layer of the neural network. The number of filters can vary by layer. Including more filters means that more parameters need to be trained in the network, which will increase training time, but more filters can also let the network learn more complex features.

In *Table 1*, the value in the "Kernel" column is the kernel size of the convolution. The "Filter multiple" column value refers to the multiple of the default Symmetric UNet filter values. The defaults can be found as the value at the top of each layer's rectangle in Figure 3. and refer to the number of nodes in each of those layers. For reference, this array is repeated here: [16, 24, 32, 48, 64, 96, 128, 128, 128, 96, 64, 48, 32, 24, 16]

Table 1: Synthetic Test Comparison

Test	Loss at 50 epochs	Accuracy at 50	Kernel	Filter multiple
S-1	0.0502	0.9871	3x3	1
S-2	0.0305	0.9894	3x3	1
S-3	0.0802	0.9851	3x3	1
S-4	0.0164	0.9939	7x7	1
S-5	0.1835	0.8627	3x3	½
S-6	0.0144	0.9948	3x3	2
S-7	0.0185	0.9928	11x11	1
S-8	0.0130	0.9947	7x7	2

These synthetic data tests were each conducted through 50 epochs to create a snapshot. Training in most cases could be improved with further epochs, but the accuracy is already extremely high in all but test S-5 (Figure 5).

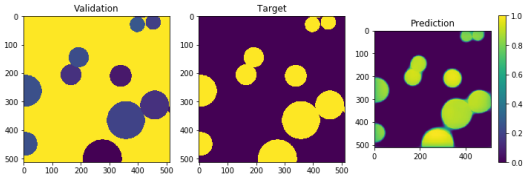

Figure 5: Example test S-3, results after 50 epochs

From the table, a few types of tests emerge:

- Test of repeatability (1-3)
- Variations of kernel sizes (1, 4, 7)
- Variations in number of filters (1, 5, 6)
- Combination of kernel and filter effects (8, combination of 4 and 6)

For the most part, these tests have strikingly similar results. The only outlier is S-5, which plateaued at a lower accuracy value. However, more training may let this architecture get to similar high accuracy.

3.2 Network Testing – Crater Data with Filled in Circle Targets

This section details the results from the experiments using crater data with filled circle targets. An example of a target tile, corresponding to one of the THEMIS 30º tiles is in Figure 6. The test numbers match the architecture from the previous section. Kernel and filter numbers are not shown.

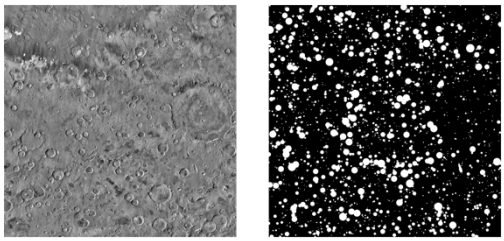

Figure 6: Filled circle target for thm_dir_N-30_030, shown: 2-32 km craters

These targets are generated using Python Image Library (PIL). A circle of the appropriate radius is added to the image using the "white" color (set pixel value to 1), leaving all remaining pixels black. This method falls under semantic segmentation because it does not distinguish between different examples of craters.

The architecture tests detailed in *Table 1* are repeated on the dataset-target pair in Table 2.

Table 2: Crater Data Test Comparison

Test	Loss at 50	Acc at 50	Loss at 500	Acc at 500	Sec per epoch
C-1	0.147	0.9568	0.271	0.9618	35 s
C-2	0.152	0.9553	0.261	0.9616	36 s
C-3	0.160	0.9536	0.288	0.9617	36 s
C-4	0.248	0.9618	0.398	0.9658	75 s
C-5	0.166	0.9485	0.182	0.9582	29 s
C-6	0.169	0.9621	0.374	0.9652	57 s
C-7	0.272	0.9626	0.416	0.9658	137 s
C-8	0.345	0.9644	0.446	0.9662	130 s

Interestingly, while the C-5 test's final accuracy is lower than the other tests, the accuracy limit is not as significantly lower compared to the other tests as the S-5 test is compared to its relative tests (Figure 7). This speaks to the importance of trying the same architecture with different data.

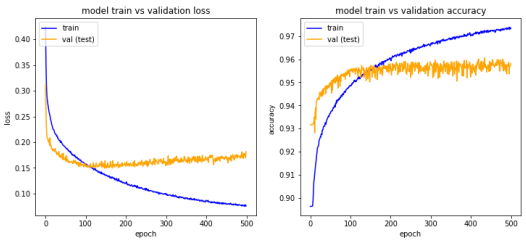

Figure 7: Loss and accuracy plots of C-5 through 500 epochs

First, this C-5 accuracy does reach similar levels

to other experiments in this set, unlike the equivalent architecture in S-5, which plateaus at a noticeably smaller accuracy compared to other S tests. Also, despite achieving the similar level of accuracy and less overfitting compared to other tests in this series, visually inspecting the results shows missing craters in C-5.

Kernel size variations are significant for detection of some large faded craters. Figure 8 shows one example of how the large crater at the bottom left of each image is only detected when the network's convolutions are set with the larger kernel values.

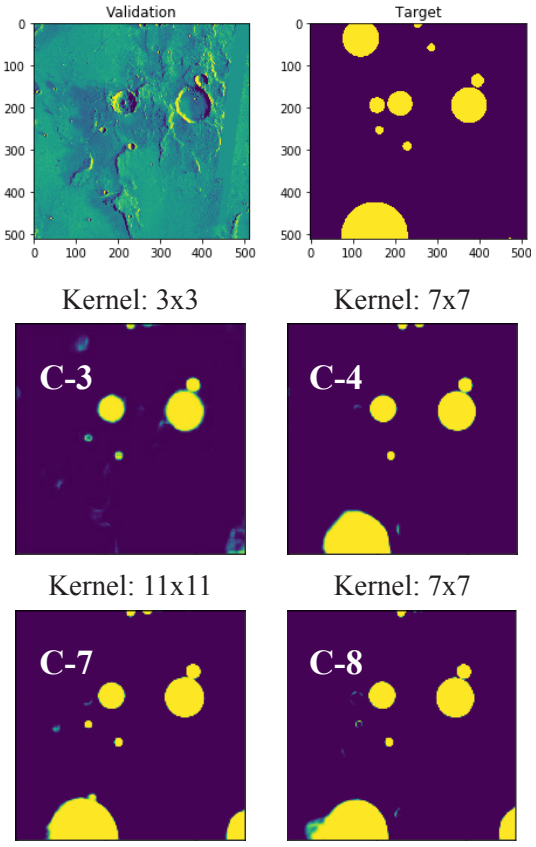

Figure 8: Effect of kernel size

Although each of the models improves in accuracy even through 500 epochs, the binary cross entropy loss went up dramatically in some cases, which would usually indicate the model is overfitting. Modifying the regularization strategy could help and will be explored in future work.

3.3 Network Testing – Crater Data with Outline Targets

The purpose of the target testing is to see how a different type of target, specifically, an outline target that varied based on the radius, would perform compared to the solid target used in the previous section. Outline targets are interesting because they allow craters within craters to be detected. An example target image can be found in Figure 9.

These targets were also created using Python Image Library (PIL). First a circle of the appropriate radius is added to the image using the "white" color (set pixel value to 1), then a smaller circle with the "black" color (pixel value 0) is inscribed at 70% of the radius. This could be tuned but for this test campaign was determined experimentally. Larger craters are added to the target tile first, which allows craters within craters to be visible.

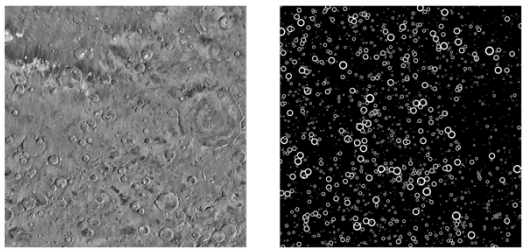

Figure 9: Outline target for thm_dir_N-30_030, shown: 2-32 km craters

Table 3: Crater Data Test Comparison

Test	Loss at 50	Acc at 50	Loss at 500	Acc at 500	Sec. per epoch
O-1	0.098	0.9721	0.1453	0.9754	33 s
O-2	0.098	0.9718	0.1342	0.9746	34 s
O-3	0.111	0.9668	0.1315	0.9742	35 s
O-4	0.142	0.9761	0.2192	0.9777	67 s
O-5	0.114	0.9669	0.0958	0.9735	29 s
O-6	0.097	0.9743	0.2030	0.9768	52 s
O-7	0.166	0.9740	0.2360	0.9782	129 s
O-8	0.184	0.9775	0.2399	0.9785	127 s

One noticeable difference in the results shown in Table 3 is the higher overall accuracy values compared to the solid targets. While this would be encouraging, it is worth noting that this is partially because there are fewer pixels to identify and the impact of large faded craters is reduced.

In the training process, pairs of images like Figure 10 are passed to the network for training or validation. While some craters are very clear to human eye in this dataset, some, like the large crater on the right edge, are very difficult to spot. Many such examples of faded craters exist in the dataset, and detecting them is difficult. In Figure

11, none of the tests could detect that particular crater, although the distinct craters are detected well.

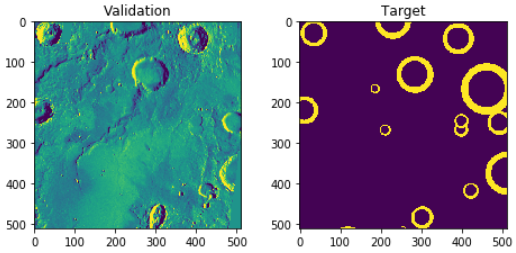

Figure 10: Example of an untrained image ("Validation") and the goal image ("Target")

Another observable effect is that for some tests, the prediction image would fit a circular outline and for others it would fit the edge outline. (Note the small elliptical crater at the bottom of the left image near 300 pixels in Figure 10 and the corresponding regions in Figure 11.) This was particularly noticeable in O-1 to O-3 vs. O-4 and O-7. (See O-3 in Figure 11.)

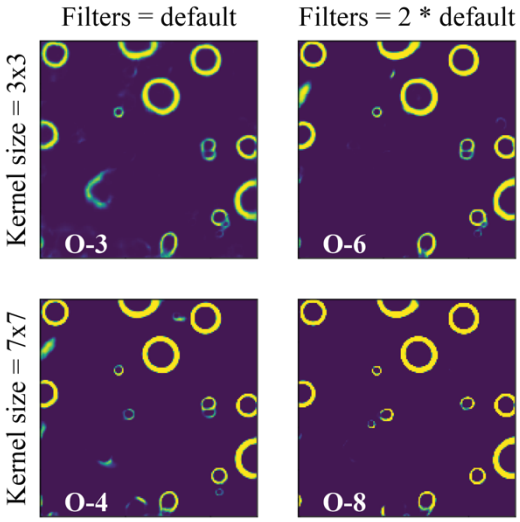

Figure 11: Comparison of effects of filters and kernel size

In Figure 11, one example output is shown across four architectures. The top left image is O-3, top right is O-6, bottom left is O-4, and bottom right is O-8. In this example, the larger kernel size predicts more circular craters. Different architectures also are better at finding craters with various characteristics. Manual inspection of results reveal how some very faint craters are detected or not detected in the final prediction image. Some craters are distinct in a subset of the O-tests and other faint craters are distinct in a different subset. This may indicate that the final crater counter will need to be a combination of trained models to identify different groups of faint craters.

The speed at which the networks train is significant as well. Gains of 0.003 in accuracy between O-1 and O-8 may not be worth taking four times as long to train, especially since O-8 was overfitting.

3.4 Other Tests

Other smaller scale tests were conducted looking at changing the batch size, using contrast, increasing the amount of data being used for training, and varying layer types.

Contrast helps in the first few epochs of training but was less helpful as training progressed. This is consistent with expectations of how the non-linear activation, ReLU is expected to work.

4 DISCUSSION

Evaluating differences between the models is difficult using traditional machine learning metrics such as accuracy. For example, all outline target tests performed better than the solid target tests by accuracy measure, but some outline predictions would be more difficult to identify by an edge finding algorithm. The pixel by pixel loss and accuracy functions that arise naturally in the segmentation approach, while valuable, are not the only relevant functions and are probably not the loss and accuracy functions that a human annotator would design.

There is a trade-off between sensitivity to detect craters in the training data and generalizability to other data. C-5 & O-5 tests showed the least overfitting, but only slightly lower accuracy. Visual inspection showed that craters are missed in these tests despite the better-looking loss and accuracy plots. Another accuracy measure that specifically targets this discrepancy must be developed for the final crater counting product.

Another consideration is how to record craters that are not in the original annotation dataset. Some of these may be unlabeled craters, craters above or below the target threshold, or objects of interest worth investigating further. Ultimately, these lessons need to be synthesized into a crater counter that may include multiple networks. There is a large subset of craters that appeared in every prediction, but a few craters showed up in only a few. The larger kernel size, for example, seems to help detect some particularly faded craters.

Future work consists of two immediate steps: (1) continue architecture tests and investigate other hyperparameter values; (2) complete the crater counting pipeline by using a circle finder or edge detector to generate a list of crater locations and sizes from the prediction images generated by the

network. Other potential directions include modifying the targets to either be more simple (same size outline for each crater) or to be a hybrid of outline targets for large craters and solid circles for smaller craters. Overfitting is an issue, particularly in the solid target experiments. Changing the decay on the learning rate, training with more data, and changing the dropout rate could help.

5 CONCLUSION

The authors detail a successful implementation of a UNet-inspired convolutional neural network and report results from testing on three dataset-target pairs. A variety of architecture experiments are conducted. Filled circles and outlines are considered as targets, and the outline targets are shown to have higher overall accuracy in the network as well as display other advantages like being able to see craters within craters.

This work demonstrates that convolutional neural networks offer an advantageous approach to the challenging and labor intensive task of analyzing space image data. The potential exists to automate a significant portion of the crater identification workflow. Realizing this potential may be possible through further exploration of machine learning techniques and through leveraging extensive previous annotation work of human crater counters. General approaches to image processing enabled by deep neural networks provide a promising start toward automation. The full realization will require continued collaboration with space scientists to design tailored algorithms that fully incorporate deep expertise derived from well-vetted human-derived annotations.

Acknowledgement

The authors acknowledge and thank their colleagues at the University of Tokyo, JAXA, and Araya. DeLatte is funded by the School of Engineering, University of Tokyo (SEUT). Funding for Crites is through the JAXA ITYF.

References

[1] Crater Analysis Techniques Working Group (1979) Standard Techniques for Presentation and Analysis of Crater Size-Frequency Data. *ICARUS* 37: 467–474.

[2] Cohen JP (2016) *Automated Crater Detection Using Machine Learning*. Doctoral Thesis, University of Massachusetts, Boston, USA.

[3] Yamamoto S, Matsunaga T, Nakamura R, Sekine Y, Hirata N and Yamaguchi Y (2017) An Automated Method for Crater Counting Using Rotational Pixel Swapping Method. In: *proceedings of IEEE Trans. Geosci. Remote Sensing*, pp. 1–14.

[4] Salih AL, Mühlbauer M, Grumpe A, Pasckert JH, Wöhler C, and Hiesinger H (2016) Automatic Age Map Construction for the Floor of Lunar Crater Tsiolkovsky. In: *proceedings of 47th Lunar Planetary Science Conference*, Houston, Texas, USA pp.1–2.

[5] Garcia-Garcia A, Orts-Escolano A, Oprea SO, Villena-Martinez V, and Garcia-Rodriguez J (2017) A Review on Deep Learning Techniques Applied to Semantic Segmentation, *arXiv*. pp. 1–23.

[6] Robbins SJ and Hynek BM (2012) A new global database of Mars impact craters ≥1 km: 1. Database creation, properties, and parameters. *Journal of Geophysical Research: Planets* 117(E05004).

[7] Mars Space Flight Facility. THEMIS Day IR Global Mosaic. Available at: http://www.mars.asu.edu/data/thm_dir/

[8] Ronneberger O, Fischer P, and Brox T (2015) U-Net: Convolutional Networks for Biomedical Image Segmentation. *Medical Image Computing and Computer-Assisted Intervention – MICCAI*. 9351(28):234–241.

[9] Silburt A, Ali-Dib M, Zhu C, Jackson A, Valencia D, Kissin Y, Tamayo D, and Menou K (2018) Lunar Crater Identification via Deep Learning. *arXiv*. pp.1–44.

[10] Keras Documentation. About Keras Layers. Available at: https://keras.io/layers/about-keras-layers/