**Using Machine Learning and Open Source Technology to Quantify Plant Species Cover1**

Arnav Dani2, Kelly McMillen3, Chris Monico4, Andrew Fagg5, Natasja van Gestel3, 6

2 The Harker School, San Jose, California 95128, USA

3 Department of Biological Sciences, Texas Tech University, Lubbock, Texas 79409-3131, USA

4 Department of Mathematics and Statistics, Texas Tech University, Lubbock, Texas 79409-1042, USA

5 School of Computer Science, University of Oklahoma, Norman, Oklahoma 73019, USA

6 Climate Center, Texas Tech University, Lubbock, Texas 79409-1005

Email addresses: AD: arnavdani@gmail.com

KM: kellymc9980@gmail.com

CM: c.monico@ttu.edu

AF: andrewhfagg@gmail.com

NV: Natasja.van-gestel@ttu.edu

Number of words: 3481

1 Manuscript received \_\_\_\_\_\_\_; revision accepted \_\_\_\_\_\_\_.

7 Author for correspondence: Natasja.van-gestel.@ttu.edu

Acknowledgements:

The authors are grateful to the Anson L. Clark Scholars Program for giving us the opportunity to pursue research, innovate, and explore new applications of technology. We would like to thank Dr. James Carr (Texas Tech University) and Ms. Wendoli Flores for coordinating and running the program. This research work was supported by the National Science Foundation (NSF 1643871).

**ABSTRACT**

Global warming is accelerating at a high rate, especially in the polar regions. In anticipation of higher temperatures, scientists are studying the impact of climate change in Antarctica, where there is a simple yet diverse plant ecosystem. When calculating carbon output over a plot consisting of multiple plants, scientists have been unable to get accurate results depicting the surface cover of different species of plant in a timely manner. Therefore, we propose taking a machine learning approach and using a set of Convolutional Neural Networks (CNNs) and computer vision models to analyze pictures of plots and classify segments of each image. The approach was 70% accurate in identifying features in random images. However, the neural network classified the individual features correctly as high as 94% of the time. Compared to traditional techniques, a neural network trained on various images can quickly identify features in photographs taken across different ambient conditions.

**Key words:** Antarctic Vegetation; Polar; Artificial Intelligence; Neural Networks; Plant Classification; Image Classification

**INTRODUCTION**

Quantifying the abundances of different plants in a landscape is a fundamental aspect of ecology. Explaining patterns of richness and diversity across the landscape (i.e., plant biogeography) has had a long and rich history, such as the pioneering research of von Humboldt and Bonpland in 1799 for elevational gradients or the latitudinal gradients as described by Alfred Russell Wallace in the 19th century. Since the quantification provides insight into how a community functions, it has also shown to be valuable in climate research. For instance, it can help to address questions related to how plant communities respond to global change or how the carbon cycle of a plant community might be affected. The response of high latitude ecosystems to global change have been of particular interest. That is because the large carbon stores in some polar ecosystems (Arctic and Antarctic) have been warming especially fast (Park et al., 2018). Traditionally, quantifying plant cover at these sites has been done using point intercept methods, visual cover estimation, and subplot frequency. However, these methods may be subjective, not good at capturing abundances of rare plants, not exact (Mamet et al. 2016), and time-consuming. Polar ecosystems that contain short-statured plants are ideally suited for a far more objective form of quantification, because the communities can be photographed and undergo image analyses (e.g., ArcGIS or QGIS). However, preprocessing each image takes hours, and the commercially available software is not accurate enough and/or is expensive (ArcGIS). ArcGIS has been used to solve a similar problem on a smaller scale (X. Sun et al., 2021), but the fact that the software is paid limits the benefit to scientists. Still, commercial software has limitations. For example, QGIS functions by having a user outline a shape around a feature. When doing this, QGIS assumes that all features of that type maintain the shape outlined by the user. This is obviously a poor assumption as two separate patches of grass are not the same. As a result, scientists using the software have to painstakingly outline every individual feature, often taking hours per image. As a result, data processing becomes the bottleneck during analysis.

To solve these problems, we propose using free and open-source computer vision and neural networks software to process the images and classify features in the images. Here, we use images from experimental plots in Antarctica that have undergone a field warming treatment. The images were taken for an upscaling modeling approach to tie lab-obtained photosynthetic rates of individual species of plants at different environmental conditions to the ecosystem-level photosynthetic rates from entire plant communities. Hence, it is important to have subjective, high-quality quantification of plant cover in the experimental plots. To our knowledge, this marks the first time artificial intelligence (AI) has been applied for plant classification for natural communities in the polar region.

**METHODS AND RESULTS**

1. **The Dataset**

The raw dataset is made up of images of 40 unique plots in the Western Antarctic Peninsula. Our hypothesis is that we can use this dataset to train and validate a neural network model. The plot images need to be pre-processed and normalized, such that they can provide a consistent dataset that can be sampled to generate the training dataset. The first step is to crop the images so that the resultant images show only the objects that need to be classified. For this particular image set, this means limiting the image to everything inside the elliptical metal ring. We used the OpenCV software program to cleanly and automatically crop the image as shown in Figure 1. The metal boundary was modeled as an ellipse. Given that the plots are not circular and that the photos were not taken at a perfect right angle to the plot, we had to develop a custom function that would analyze the contours, identify an ellipse, and then trace the boundary around the plot. Everything outside the boundary could then be discarded. The inputs to the custom function are a contour, a coordinate as a center point, and a maximum deviation from the orthogonal. Using these inputs, the custom function discarded a multitude of contours that were not applicable. A contour was discarded if its center was not near the center of the image, if the contour did not have a circular or elliptical boundary, or if the area within the contour was too small compared to the size of the image. The details of this approach are provided in the supplemental materials.

Each input image contains various types of objects that need to be classified. So, the next step in the creation of the training dataset was to decompose each input image into smaller segments, where each segment contains only one type of target object, such as a rock, grass, soil etc. This was done manually using visual inspection and clipping software. Each image was segmented as shown in Figure 1. The output of this process was a set of 1,600 images, where each image is composed primarily of one type of target object. Out of the 1,600 images, 635 were rocks, 151 were moribund moss, 149 were sanionia moss, 146 were hairgrass, 143 were soil, 101 were chorisodontium aciphyllum, 90 were polytrichum strictum, 65 were white lichen, 86 were in the random category, and 34 were algae. These comprise the 10 target features that the neural network has to learn to identify. Below is the procedure to create the complete training dataset

For every image of a plot

{

Crop it using previously written code and save it

Open the image in an image viewer

Visually identify all types of objects such as rocks, moss, algae, grasses etc for every type of object in the image

For object in image

{

Use an image snipping tool to draw a rectangular bounding box outside the object and save it as a separate image (.jpg) file

Visually identify the type of feature in the image and tag the image as rock, soil etc.

}

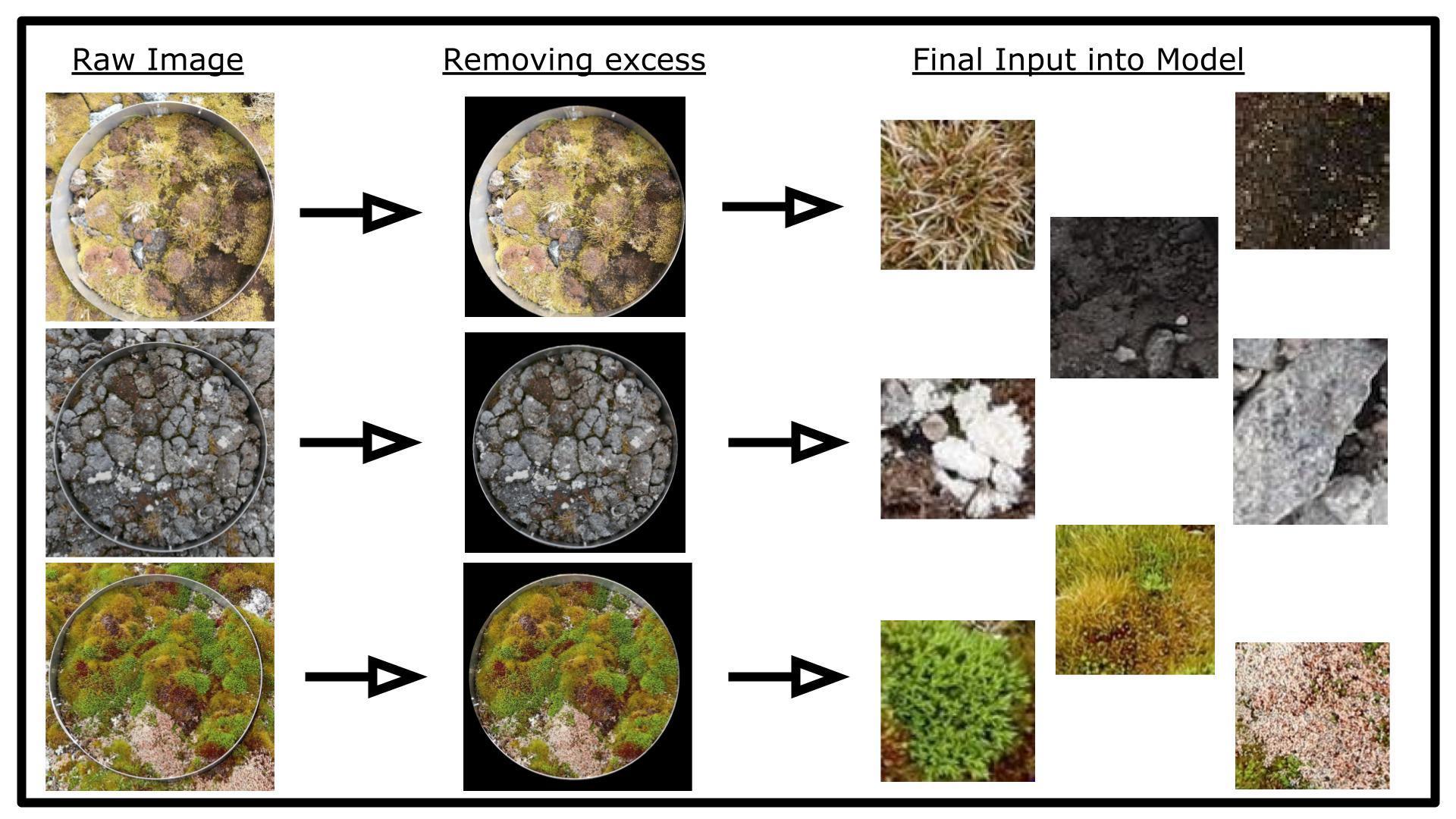


Figure 1 Processing of raw input images as sub-images to be used as input to train the neural network model.

1. **Procedure**

Our first attempt was to use a model that would take in a full image of the plot, and produce as output, a list of all the target objects that are present in the image and the percentage of area that each target object covers in that image. This approach required a deep neural network that could perform multi-object classification. Deep networks have been applied to plant classification (Ghazi et al., 2017), but our dataset of 40 plots is far too small compared to the tens of thousands of images necessary for results. So, we discarded this approach. We next decided to follow the segmentation approach, where there is one feature in each sub-image that the model can identify out of ten possible features. This is the approach common in the plant classification literature and has been used with moss and lichen (Galanty et al., 2021; Ise et al., 2017; Y. Sun et al., 2017). However, our training dataset of 1,600 images was too small for this approach, especially since the number of occurrences of each feature is not evenly distributed. These works are also focused on only classifying a single type of plant, not a wide variety. Therefore, we decided to use a binary object classification approach. With the binary classification approach, we use models to provide a Yes/No answer for particular types of target objects. We developed a neural network model to classify an image as a rock or not, and another model to classify an image as grass or not and so on. In total we built 10 neural network models, one for each target feature. Each model was then trained using a sample of data from the input dataset. We curated the training set to ensure a balanced mix of positive and negative outputs. The neural network for all the 10 models is the same; each model is trained on different target values. The sub-images undergo the following transformations to prepare the images before they are provided as an input to a model: resizing to 244x244 pixels, random horizontal flips, random rotations within 10 degrees, random changes to the brightness, saturation, and contrast, and normalization to the mean. With the binary classification approach, 10 neural networks needed to be trained, one for each target feature. The models are trained to output either 0 or 1, where 0 implies the feature is missing and 1 implies the feature is present. An output closer to 1 denotes a higher probability of the feature being in the image. The trained models took a random segmented image as input and classified it as a specific feature type. This was accomplished by running the segmented image through all models and returning the target type name where the model’s output was the highest. By taking the maximum value among the outputs, we limited the impact of errors in some of the models. The following procedures describe the steps:

Procedure to classify a new segmented image

model\_output = 0.0

threshold\_value = 0.5

image\_type = Unknown

For every current\_model in {rock model, soil model, ….}

{

current\_model\_output = Result of running the image through the current\_model

If (current\_model\_output > threshold\_value and current\_model\_output > model\_output)

{

model\_output = current\_model\_output

image\_type = current\_model type (rock, soil, hairgrass etc.)

}

}

Return image\_type

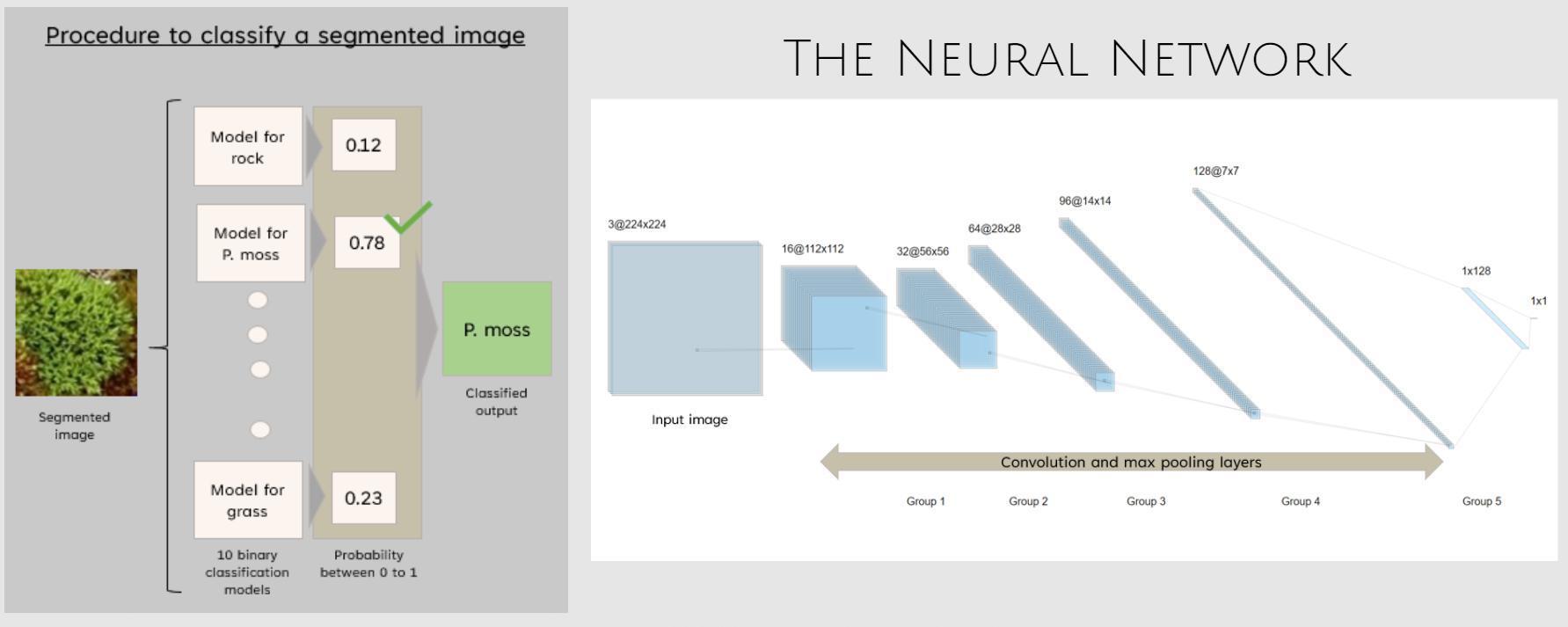


Figure 2: A visual representation of the approach and the neural network

1. **The Neural Network**

All 10 neural network “models” are variations of the same foundational model with the same layers and parameters. We decided to build our own neural network because many of the popular pre-trained models suffer from two limitations that make them unsuitable for our domain. First, they were not designed for binary object classification and are more suited for identifying a single object out of many potential options. Second, these pre-trained models are deep neural networks with hundreds of millions of parameters, meaning that our small dataset of 1,600 images would barely put a dent in their weights. Therefore, we built our own custom model with a relatively small number of 1.08 million parameters. The custom model consists of 5 groups of layers, with each group containing two convolutional layers and a max pooling layer as shown in Figure 2. The first group takes in a 3 channel input. All convolutions were conducted with a kernel size of 3x3, a stride of 1x1, and a zero padding of 1. At the end of every group of layers, there was a max pooling layer to minimize the number of parameters and a RELU activation function. Each max pooling layer had a stride of 2x2. The first convolutional layer takes in 3 inputs and has 10 outputs, and the second layer has 10 inputs with 16 outputs. The 2 convolutions were followed by the max pooling and activation. The first convolution in the second group had an input of 16 and an output of 24 and the second convolution had an input of 24 and an output of 36. After the pooling and activation, the third group had convolutions with outputs of 44 and 64, and the fourth group had convolutions with outputs of 81 and 96. Both the third and fourth group had pooling and RELU activation at the end. The 5th group consisted of a single convolution with an input of 96 and 128 and was followed by a max pooling layer with a stride of 2x2. Finally, two linear layers were added at the end of the 5th group; the first had an input of 6,272 and an output of 128 and the second had an input of 128 and an output of 1. The output is intended to be always between 0 and 1. This final output would be the determiner of the binary classification. If the output was greater than 0.5, the sub-image was classified as the model target type. If the output was less than 0.5, it was marked as not being the model target type. A visual depiction of the approach is in Figure 2. The custom model was trained on the images as described above. To train each of the ten models, we used the mean square error loss function. We chose the mean square loss function because other loss functions were more tuned for multi classification models with multiple outputs. After experimenting with binary cross entropy and sigmoid loss functions, we concluded that the simplicity of mean square error made it most applicable to our scenario. Each model used the Stochastic Gradient Descent Optimizer with a momentum of 0.9. Because of the large variety and the small size of each dataset, each model had a slightly different learning rate that was determined through experimentation. Similar to the learning rate, the number of epochs varied for each model. During experimentation, we saw that some models flatten faster than others due to the difference in the frequency of occurrence of the features in the training set. In addition, all models had a scheduler with a gamma 0.1 that would decrease the learning rate after an experimentally obtained step size. Rocks were trained with a learning rate of 0.01 and a step size of 15; soil was trained with learning rate of 0.014 and a step size of 15; P. strictum moss was trained with a learning rate of 0.01 and a step size of 14. More details about individual models are available in the supplemental materials. The next step is to validate or test the custom models. To validate the models, we used a separate validation set. We needed to exclude images that were present in the training dataset and ensure that the validation set had a mix of images - images that contained the target object and images that did not. The output of the neural network is a single floating point value. We applied a simple thresholding function to convert the floating point value to a binary (0 or 1) value. It is important to note that the thresholding function is not a part of the neural network, and was used solely to “digitize” the output. The model itself did not have any visibility into the output of the thresholding function. Adjustments to the parameters of the neural network model were done on the basis of the loss function and optimizer function.

1. **Results**

After training and validating all 10 custom models separately, we arrived at the following results. The custom model for rocks was able to identify a rock correctly 91% of the time. Similarly, the custom model for the soil identified soil correctly 88% of the time. Along the same lines, the identification accuracy was 94% for white lichen, 94% for moribund moss, 80% for the sanionia moss, 75% for hairgrass, 87.5% for Polytrichum strictum, 73% for chorisodontium aciphyllum, 97% for algae, and 86% for the bryum spo./dead moss/random category. However, given the small dataset for white lichen, sanionia moss, hairgrass, C. aciphyllum, and algae, it is likely that models for these targets have some inaccuracy. While training, we noticed more unexpected movements and progressions with white lichen, sanionia moss, hairgrass, c. aciphyllum, and algae compared to the other targets such as rock, soil, p. strictum and moribund moss, which followed normal conventions. The accuracies are listed in Table 1. For rock, soil, p. strictum, and moribund moss, we observed that the training loss and accuracy were improving at a rate that was expected. This is not a surprise given that these four objects were not only abundant in the training dataset, but also relatively easy to classify compared to the other features. In Table 2, we show the progression of the training of moribund moss through the various epochs. The graph flattens after a period of improvement. Finally, we wanted to test whether a random sub-image could be evaluated correctly. To do so, we passed that image through all 10 models and took the highest output. While we did not test a large number of images because most images were used for training, we observed that the combination of models identified the features correctly 70% of the time.

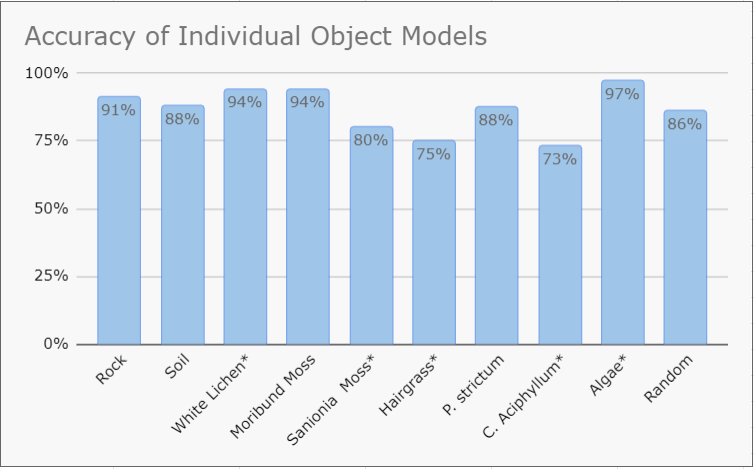


Table 1. Validation accuracy of all the separate models after they were trained

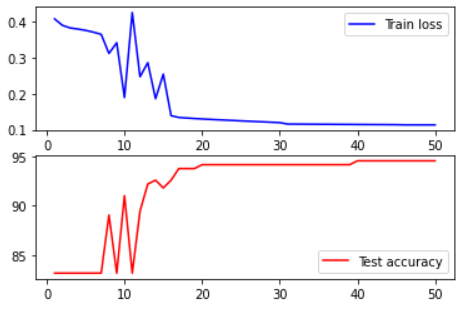


Table 2. The training diagram of the moribund moss model

1. **Errors and Future Developments**

After experimentation, we came up with a few reasons for why the models were not highly accurate. First, the small size of the training dataset and the skewed distribution of the target features in the training dataset are the biggest impediments to getting a highly accurate output from this approach. Next, the segmentation and ground truth classification of the images were done manually and there is a small chance that human error may have resulted in imperfections in the dataset, leading to inaccuracies in training. In addition, we noticed some patterns in the results. For example, both moribund moss and soil have a dark color, leading the model to sometimes incorrectly interpret a dark patch as either. In addition, chorisodontium aciphyllum and hairgrass have similar visual textures, leading to a similar misidentification. Given that c. aciphyllum and grass do not have a distinctive shape, the model may misidentify one for the other. Due to these potential errors and the nature of manual segmentation, an AI based approach needs a much larger training dataset to completely replace the current methods. There are parts of our work that can be used right away in all scenarios. For example, the procedure written to crop the raw image to only the relevant sections is useful to scientists who currently have to manually edit the images.

Most of the errors within the models can be solved with a larger, more representative training dataset. A system that uses tiling algorithms alongside computer vision to automatically locate and extract features would be able to eliminate the errors introduced by a manual segmentation approach. Once image segmentation is automatic and the accuracy of the binary classification is high enough, the process can be fully automated to take in a raw image and produce a distribution of features present in the input image. Finally once the dataset is large enough the binary classification neural network model can be replaced by a multi-classifier which would eliminate the requirement to run the input image through 10 different models.

**CONCLUSIONS**

We conclude that it is worth applying machine learning and AI techniques to improve the process of classifying plant species cover. We demonstrated that open-source software packages such as OpenCV and Pytorch are viable solutions. By using segmentation, a computer will be able to classify the different species. Compared to current software, a user can get within 70% accuracy in 75% less time. The approach described in this paper can augment the existing processes. Some more exploration is needed before it can completely replace it. The process of image segmentation, use of neural networks, and binary classification of an image has promise. We need a larger training dataset to tune the models to the desired level of accuracy. With a larger and more diverse dataset, we can achieve a high degree of prediction accuracy for all the models. We can also use more sophisticated contouring methods to automate the process of segmenting an image. Finally the neural network model structure can be evolved to a true multi-classification mode, which may make the classification process run much faster. The combination of image cropping, automated segmentation, and accurate neural network models can greatly simplify and automate the process of determining the plant species cover for any input image. Since this will be built using open-source software, this solution will also be free for anyone to use. Finally, we would like to contribute our dataset and our model to the open source community so that others can benefit from it and build on it.

**LITERATURE CITED**

Mamet, S. D., Young, N., Chun, K. P., and Johnstone, J. F. (2016). What is the most efficient and effective method for long-term monitoring of alpine tundra vegetation? *Arct. Sci.* 2, 127–141. doi:10.1139/as-2015-0020

Galanty, A., Danel, T., Węgrzyn, M., Podolak, I., & Podolak, I. (2021). Deep convolutional neural network for preliminary in-field classification of lichen species. *Biosystems Engineering*, *204*, 15–25. https://doi.org/10.1016/j.biosystemseng.2021.01.004

Ise, T., Minagawa, M., & Onishi, M. (2017). *Identifying 3 moss species by deep learning, using the “chopped picture” method*. https://doi.org/10.4236/oje.2018.83011

Mehdipour Ghazi, M., Yanikoglu, B., & Aptoula, E. (2017). Plant identification using deep neural networks via optimization of transfer learning parameters. *Neurocomputing*, *235*, 228–235. https://doi.org/10.1016/j.neucom.2017.01.018

Park, K., Kang, S. M., Kim, D., Stuecker, M. F., & Jin, F. F. (2018). Contrasting local and remote impacts of surface heating on polar warming and amplification. *Journal of Climate*, *31*(8), 3155–3166. https://doi.org/10.1175/JCLI-D-17-0600.1

Sun, X., Wu, W., Li, X., Xu, X., & Li, J. (2021). Vegetation abundance and health mapping over southwestern Antarctica based on worldview-2 data and a modified spectral mixture analysis. *Remote Sensing*, *13*(2), 1–26. https://doi.org/10.3390/rs13020166

Sun, Y., Liu, Y., Wang, G., & Zhang, H. (2017). Deep Learning for Plant Identification in the Natural Environment. *Computational Intelligence and Neuroscience*, *2017*. <https://doi.org/10.1155/2017/7361>