

NASA/TM-20260000672



MicroNet Assistant Tool User's Manual v1.0

*Brandon L. Hearley, Joshua Stuckner, Evan J. Pineda, and Laura Wilson
Glenn Research Center, Cleveland, Ohio*

NASA STI Program Report Series

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA scientific and technical information (STI) program plays a key part in helping NASA maintain this important role.

The NASA STI program operates under the auspices of the Agency Chief Information Officer. It collects, organizes, provides for archiving, and disseminates NASA's STI. The NASA STI program provides access to the NTRS Registered and its public interface, the NASA Technical Reports Server, thus providing one of the largest collections of aeronautical and space science STI in the world. Results are published in both non-NASA channels and by NASA in the NASA STI Report Series, which includes the following report types:

- **TECHNICAL PUBLICATION.**
Reports of completed research or a major significant phase of research that present the results of NASA programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA counterpart of peer-reviewed formal professional papers but has less stringent limitations on manuscript length and extent of graphic presentations.
- **TECHNICAL MEMORANDUM.**
Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain

minimal annotation. Does not contain extensive analysis.

- **CONTRACTOR REPORT.**
Scientific and technical findings by NASA-sponsored contractors and grantees.
- **CONFERENCE PUBLICATION.**
Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or cosponsored by NASA.
- **SPECIAL PUBLICATION.**
Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.
- **TECHNICAL TRANSLATION.**
English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services also include organizing and publishing research results, distributing specialized research announcements and feeds, providing information desk and personal search support, and enabling data exchange services.

For more information about the NASA STI program, see the following:

- Access the NASA STI program home page at <http://www.sti.nasa.gov>



MicroNet Assistant Tool User's Manual v1.0

*Brandon L. Hearley, Joshua Stuckner, Evan J. Pineda, and Laura Wilson
Glenn Research Center, Cleveland, Ohio*

National Aeronautics and
Space Administration

Glenn Research Center
Cleveland, Ohio 44135

Trade names and trademarks are used in this report for identification only. Their usage does not constitute an official endorsement, either expressed or implied, by the National Aeronautics and Space Administration.

Level of Review: This material has been technically reviewed by technical management.

This report is available in electronic form at <https://www.sti.nasa.gov/> and <https://ntrs.nasa.gov/>

NASA STI Program/Mail Stop 050
NASA Langley Research Center
Hampton, VA 23681-2199

Contents

Abstract.....	1
1.0 Overview	1
2.0 Home	2
3.0 Model Creator.....	3
3.1 File Selection	3
3.2 Resize Images	4
3.3 Crop Images.....	5
3.4 Segment Images.....	7
3.5 Segment Anything (SAM).....	7
3.5.1 Manual	9
3.5.2 Adding Multiple Classes	9
3.6 Export	10
3.7 Data Definition	10
3.8 Train Model	10
3.8.1 Model Definition.....	12
3.8.2 Labels.....	12
3.8.3 Image Augmentation.....	12
3.8.4 Training Settings.....	14
3.8.5 Training the Model.....	14
4.0 Model Inference.....	15
4.1 Segmenting an Image.....	15
4.2 Post-Processing Toolsets	17
4.2.1 Circular Fiber Idealization	17
4.2.2 TGO Measurement.....	20
Appendix A.—MicroNet Model Settings	23
A.1 Available Architectures	23
A.2 Available Encoders	23
Appendix B.—Index.....	25
References.....	26

MicroNet Assistant Tool User's Manual v1.0

Brandon L. Hearley, Joshua Stuckner, Evan J. Pineda, and Laura Wilson
National Aeronautics and Space Administration
Glenn Research Center
Cleveland, Ohio 44135

Abstract

The MicroNet Assistant Tool allows users to upload, pre-process, and annotate images for training microscopy segmentation models using MicroNet. The tool also trains the segmentation model and enables use of trained models for segmentation of additional images. Finally, it generates voxelated geometric models of the microstructure for use in micromechanics analysis toolsets. The tool provides an intuitive graphical user interface (GUI) that enables allows users to resize images, crop images to develop a training data set, rapidly label classes using either the Segment Anything Model (SAM) or built-in manual drawing tools, and train a MicroNet segmentation model. Additionally, it enables users to perform model inference using a previously trained MicroNet model, and contains various post-processing toolsets to develop geometric models for analysis tools and perform measurements on segmented images. The MicroNet Assistant Tool thus significantly reduces the time to train and use a microscopy segmentation model and reduces the barrier-to-entry scripting knowledge and skillset needed to use advanced segmentation modeling capabilities.

1.0 Overview

MicroNet, a software tool for building deep learning microscopy segmentation and analysis models with less training data, has previously shown its ability in rapidly segmenting microscopy images necessary for material characterization with high accuracy at a fraction of the cost manual annotation (Ref. 1). In its current state, however, it suffers from two main drawbacks that limit model training efficiency and inhibit user implementation: 1) Training the model requires manual labeling of images, which can be tedious and require subject matter expertise and 2) training a model requires the user to have extensive knowledge of Python scripting, limiting those who can use the tool. Classic computer vision techniques for automatic segmentation, such as image thresholding and morphology operations, address the first drawback by offering repeatable, rapid segmentations, but they are often not robust enough for complex microscopy segmentations (Ref. 2). Recent developments in other machine learning segmentation models, such as the Segment Anything Model (SAM) from Meta (Ref. 3), have looked to address these short-coming by offering graphical user interfaces (GUIs) for image segmentation and generalizing model inference such that a machine learning model does not need to be trained for specific use cases, rather relying on user prompts to find features in an image. When applied to microscopy images, however, manually prompting each individual segment (e.g., fibers in a composite) can be as time consuming as manual labeling, and artifacts often found in the microscopy imaging techniques can lead to false predictions of classes, thus requiring manual editing of the model results.

The MicroNet Assistant Tool looks to bridge the gap between the superior segmentation capabilities offered by MicroNet and rapid implementation of general segmentation models like SAM while providing users with an interactive GUI for training and implementing a segmentation model. Through the GUI, the MicroNet Assistant Tool enables users to upload images for training a segmentation model, preprocess the images for improved MicroNet performance, and rapidly annotate images for training using SAM and built

in manual editing tools. Once images have been labeled, users can easily define training, validation, and test data sets, define the model architecture, encoder, and pretrained weights available in MicroNet, define settings for image augmentation to improve the model performance for variation in images, and train a MicroNet model. Additionally, users can perform model inference from trained MicroNet models through the GUI to segment new images and use built-in post-processing toolsets to develop geometric models for analysis codes or perform automatic measurements. Thus, the MicroNet Assistant Tool provides users with little Python experience an efficient and effective methodology to rapidly segment microscopy images, with the goal of using the segmented results in materials analysis.

2.0 Home

The MicroNet Assistant Tool is launched by executing the main.py script, which displays the Home page (Figure 1).

The Home page allows users to create a new segmentation model, edit an existing model, or segment an image using a previously trained model. When defining a new segmentation project or loading an existing project, all data is saved to a *.seg file. Pressing the *New Model* button will prompt the user for the segmentation project file name and pressing the *Edit Model* will prompt the user for an existing segmentation project file. Once the segmentation project file is defined, the Model Creator page is loaded, which allows users to perform all pre-processing and annotation of images and train a model using MicroNet. Pressing the *Segment Image* button loads the Model Inference page, which allows users to segment an image using a trained MicroNet model and access the built-in post-processing tools.

Loading the Home page also creates the toolbar at the top left of Figure 1. The *Home* button returns the user to the Home page. If a project file has been defined, the user will first be prompted to save the project. The *Save* button saves the currently defined project file. The *Help* button loads the help documentation for the MicroNet Assistant Tool in the user's default web browser. The help documentation includes a description of each feature present in the MicroNet Assistant Tool with associated videos.



Figure 1.—MicroNet Assistant Home Page.

3.0 Model Creator

The Model Creator page provides the user with different navigable tabs for pre-processing raw images for MicroNet, rapidly annotating the images for training, defining the settings for a MicroNet segmentation model, and training the model. Each tab and its functionalities are described in the following subsections.

3.1 File Selection

The *File Selection* tab allows selection of the images used for manual labeling and training of a MicroNet model (Figure 2). Users can set the directory containing the images that will be used for manual labeling using the *Set Image Directory* button. Supported image files (*.png, *.jpg, and *.tiff) in the directory will be displayed in the *All Images* list box. To add individual images to the project, select each file from the *All Images* list box and press the → button, moving them in to the *Manual Segmentation Images* list box. Similarly, images can be removed from the project using the ← button. All images can be added to the project by pressing the *Add All Images* button and all images can be removed from the project by pressing the *Remove All Images* Button.

Selected images can be viewed by pressing the *View Selected Images* button, which will display the images in a new window (Figure 3). Multiple images can be selected and viewed, use the → and ← buttons to navigate between images.

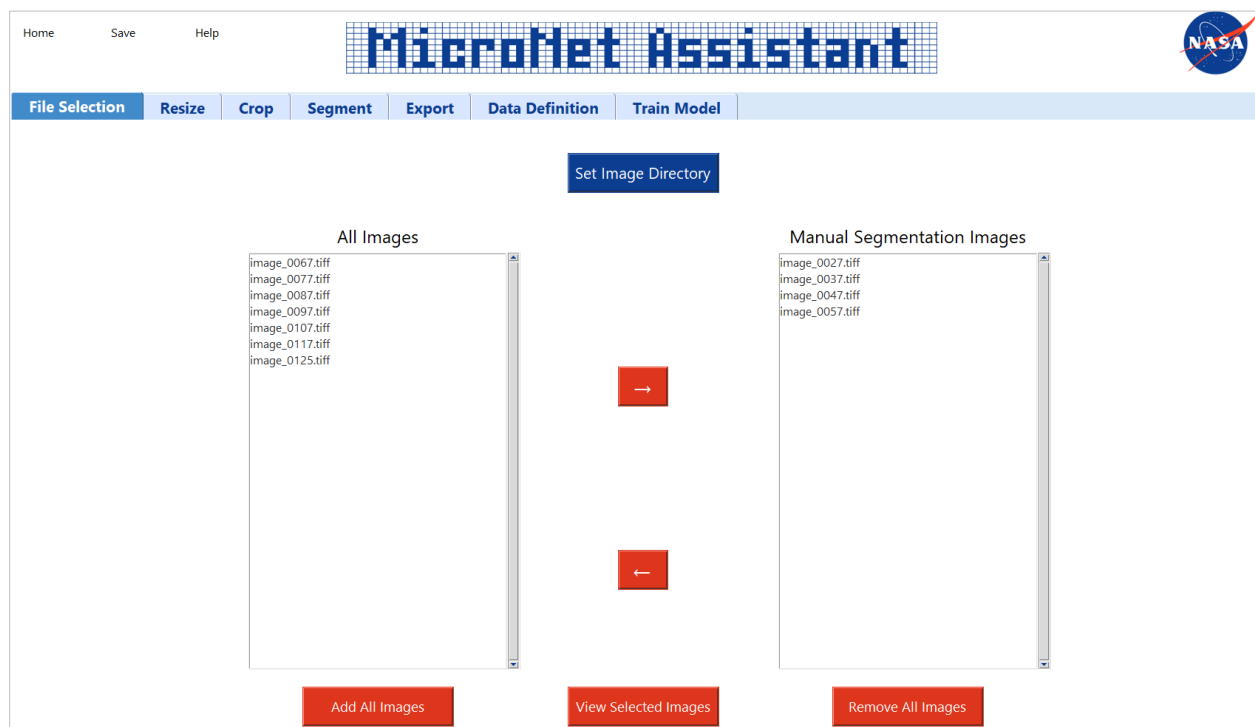


Figure 2.—File Selection tab.



Figure 3.—Viewing Images from the Image Selection page.



Figure 4.—Resize Images tab.

3.2 Resize Images

The *Resize Images* tab allows users to scale the images for segmentation (Figure 4). MicroNet performs best when the maximum pixel size in either the height or width direction for an image is between 1000 and 2000 pixels. To view an image's current size and resize it, select an image from the lefthand list and select *Load Image* to display the image.

Once an image is loaded, the the X and Y dimensions in pixels are shown to the right of the image. Image scaling is performed by changing the scale entry box below the image dimensions, where the entered value is multiplied by the current X and Y dimension values, maintaining the image aspect ratio. To edit just the selected image, uncheck the *Scale All Images* box. To scale all images in the project file, check the *Scale All Images* box. Press the *Scale* button to scale the image(s).

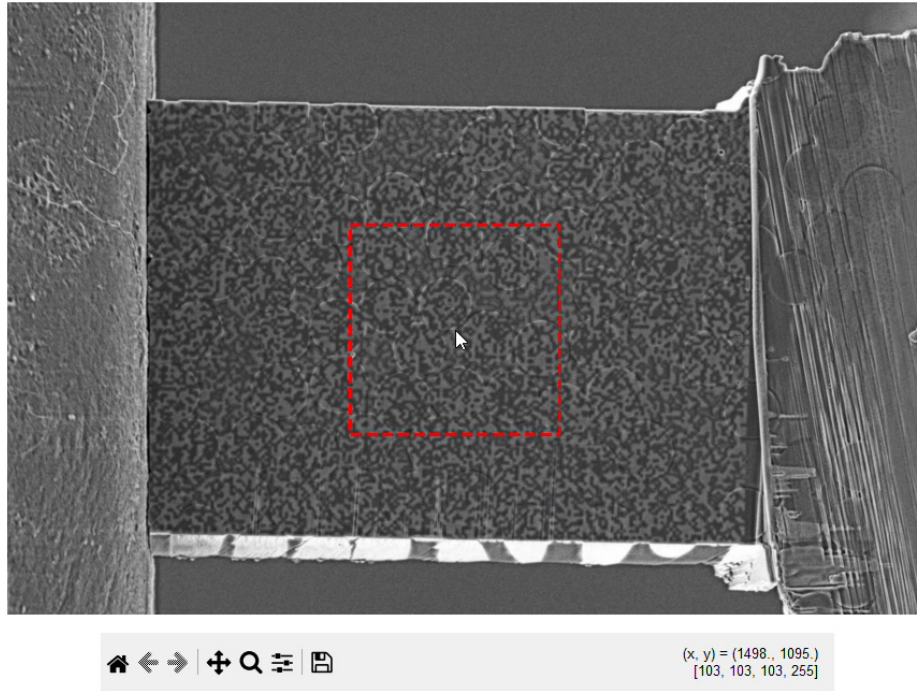
3.3 Crop Images

The *Crop Images* tab allows cropping full images into square regions for MicroNet training (Figure 5). MicroNet requires labeled images have equal width and height dimension when training a model. To crop a square region from an image, select an image from the lefthand list and select *Load Image* to display the image. If the image does not need to be cropped (i.e., it has previously been cropped to a square), press the *Add Whole Image* button to add the image to the righthand list.

To crop an image, first set the size of the crop window using the entry box below the image. Press *Enter* to set the new crop window size. MicroNet requires image dimensions be divisible by 32; however, when training, MicroNet will randomly select a compatible domain within the cropped image. Therefore, it is not required that the specified crop window be divisible by 32. To view the crop window, hover over the image (Figure 6(a)). Left click on the image to lock the current crop window, changing the crop window border from a dashed line to solid (Figure 6(b)). To add the cropped image to the project, press *Enter*. The cropped image name is automatically generated and saved in the righthand list. Multiple cropped images can be taken from the same full-size image. To discard the cropped image, press *Escape*.



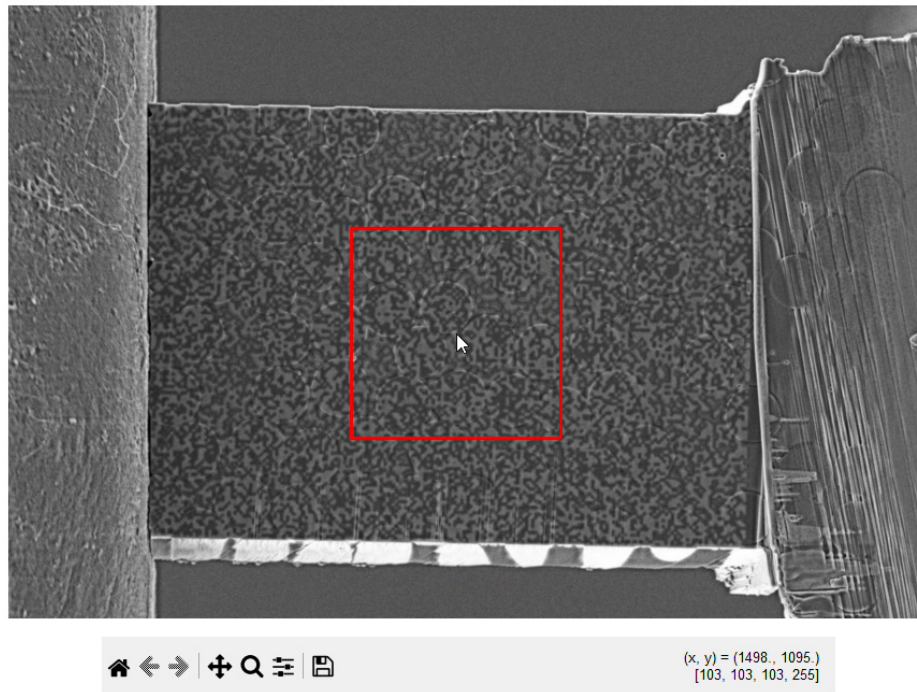
Figure 5.—Crop Images tab.



(a)

Crop Window:

700



(b)

Crop Window:

700

Figure 6.—Cropping an image: (a) Viewing the crop window and (b) locking the crop window.

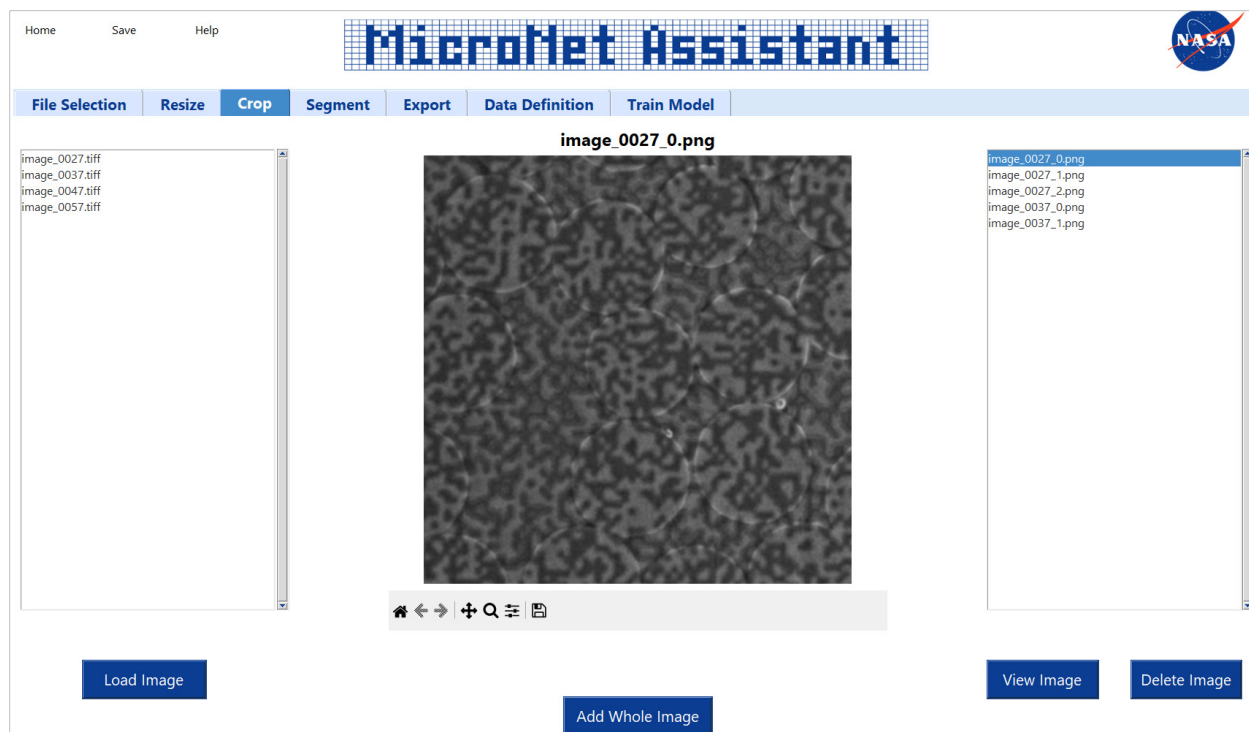


Figure 7.—Viewing a cropped image.

To view a cropped image, select a file from the righthand list and press the *View Image* button, displaying the cropped image on the screen (Figure 7). To delete a cropped image, select a file from the righthand list and press the *Delete Image* button. Only the cropped image will be deleted; the original image will remain in the lefthand list.

3.4 Segment Images

The *Segment Images* tab allows labeling of each image in the project for training (Figure 8). Images can either be manually labeled or labeled using an available built-in segmentation model. To begin labeling an image, select the filename from the list and a labeling method from the drop down menu below the list. Press the *Load Image* button to display the image on the screen and load the segmentation model associated with the labeling method, if applicable. If an image contains annotations, they will be displayed. There are currently two options available for labeling:

- Segment Anything (SAM)
- Manual

3.5 Segment Anything (SAM)

The Segment Anything Model (SAM) from Meta enables rapid labeling of images by taking input prompts as points are labeled either in the background or foreground for each individual body in the class, or features to be recognized by the segmentation model associated with the defined color. The user defines a few prompt points, and SAM is able to rapidly identify an individual body in the image. To segment a body using SAM, use the left mouse click to add foreground input prompt points (label area as in the class). Use the right mouse click to add background input prompt points (label area as not in the class). Foreground

points will display in green and background points will display in red. Multiple input prompts can be used for each individual body. The generated mask will automatically appear over the image with each prompt (Figure 9(a)). To keep the current mask in the class, press “Enter”. The mask will turn a darker shade when confirmed and prompt points will be removed from the image (Figure 9(b)). To discard the current mask, press “Escape”. Repeat for each individual body in the image. **Note: using multiple prompts for multiple bodies without individually confirming each body will likely result in poor results.**

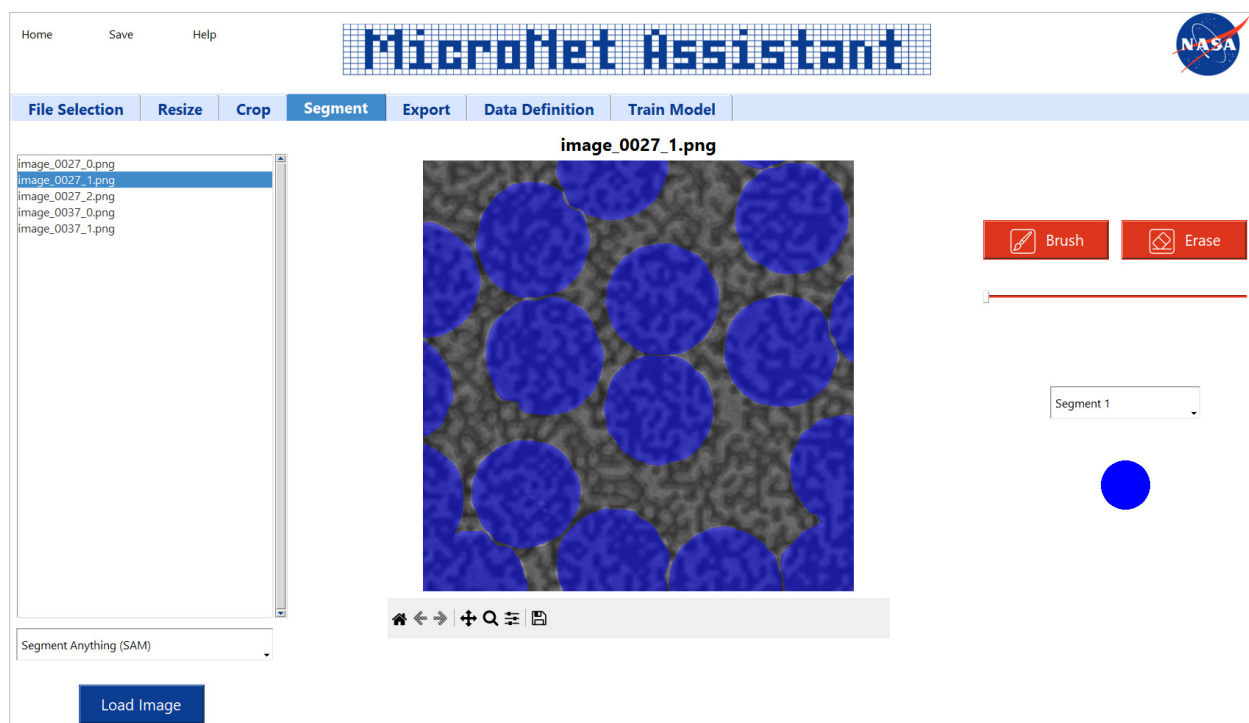


Figure 8.—Segment Images tab.

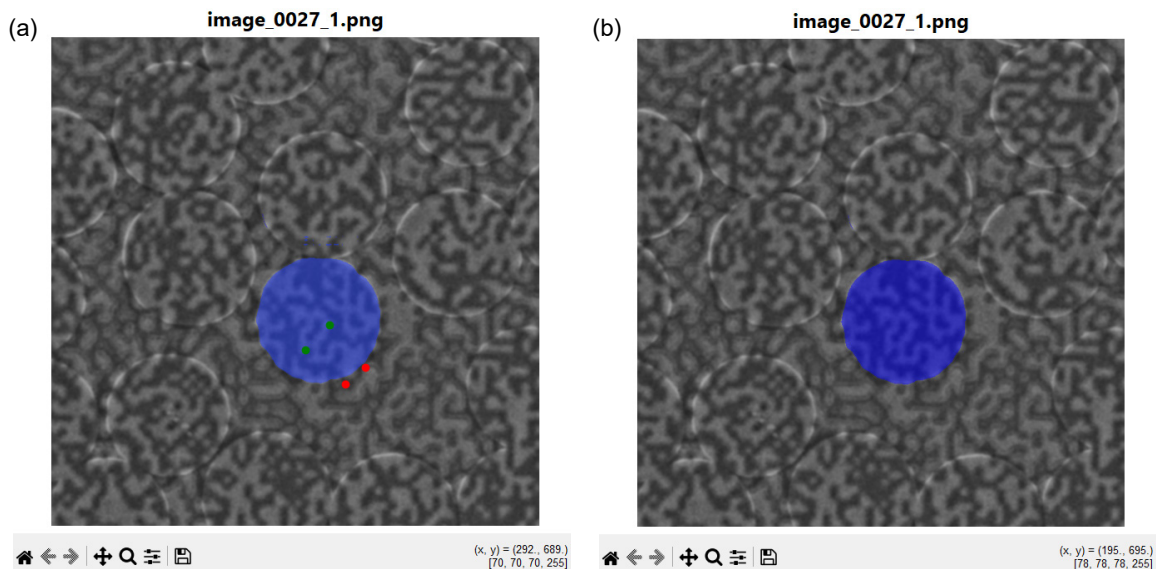


Figure 9.—SAM Labeling: (a) Adding background and foreground prompts and (b) Confirming the mask.

3.5.1 Manual

Manual labeling of images is available within all of the segmentation options. It can be used to edit/refine an automatic segmentation method or label images from scratch. Manual labeling is performed using the drawing tools on the righthand side of the page (Figure 10). The *Brush* button enables adding pixels to the class and the *Erase* button enables removing pixels from the class. When a drawing tool is active, the button will appear gray, and when it is inactive it will appear orange. The size of the brush/eraser can be changed using the slider bar.

Pixels can only be added/erased for the current class selected on the righthand drop down menu. The toolbar under the image can be used to zoom in and out of a region of interest. Hovering the mouse over the image will display the brush/eraser (Figure 11(a)). To add/remove points, click with the left mouse button and drag over the desired area, which will appear in white. Releasing the left mouse button will add/remove all pixels in the area (Figure 11(b)).

3.5.2 Adding Multiple Classes

When segmenting an image, the MicroNet Assistant Tool allows three distinct classes to be labeled for segmentation (and the background as a fourth class). To change the class, use the drop down menu on the righthand side to select a segmentation option (Figure 12). The associated color will automatically be updated.

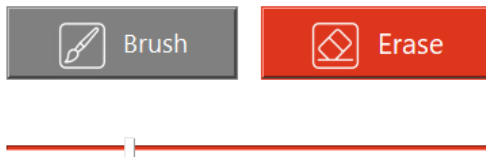


Figure 10.—Manual Drawing Tools with *Brush* active.

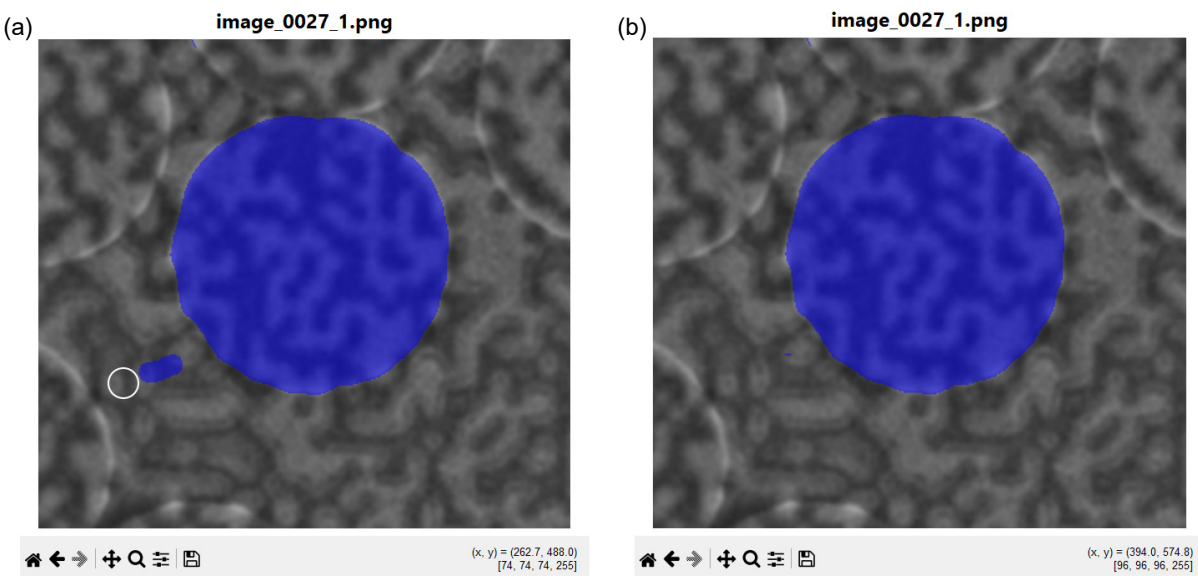


Figure 11.—(a) Displaying the brush/eraser on the image and (b) Manually removing pixels.

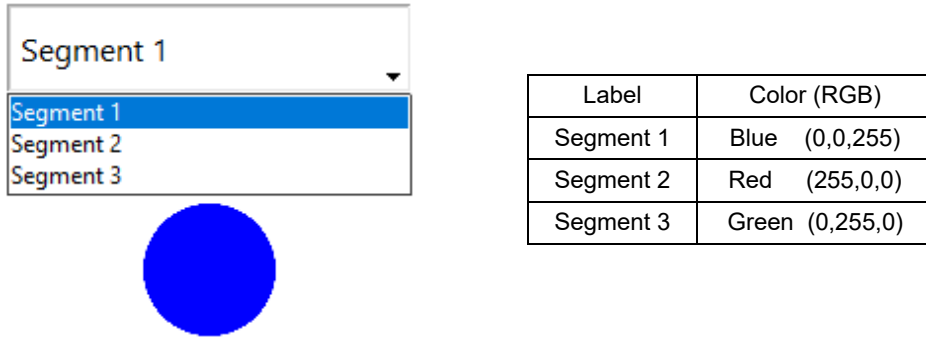


Figure 12.—Available colors/classes for segmentation.

When using the manual segmentation tools, only the selected class can be edited. If a pixel has been labeled for Segment 1 and should be labeled Segment 2, the user must first select Segment 1 from the drop down list and activate the erase tool. Only Segment 1 pixels can be erased if Segment 1 is selected, pixels in other segments will not be affected. Then the user must select Segment 2 from the drop down and activate the brush tool to add the pixels to the class.

3.6 Export

The *Export* tab allows the user to review the manual labels for each image and export the images into a single folder for training (Figure 13). To view an image, select a file from the list and press *Load Image*. The labeled classes will display in one of the true colors (e.g., Segment 1 will be (0,0,255)). To export the images for training, click the *Export Images* button and choose the directory to save the images to. For each image, the original cropped image and the labeled image will be saved with “_mask” added to the filename.

3.7 Data Definition

The *Data Definition* tab enables defining which images will be used in the training, validation, and test sets for training the segmentation model (Figure 14). Images are initially placed in the *All Images* list. To move an image from one list to any other list, select the file name and press the *Add to –* button below the desired list. **Note: it is recommended that the training data set contains at least eight images, and the validation data set contains at least two images.**

3.8 Train Model

The *Train Model* tab allows users to define the settings for the MicroNet model and then train a model (Figure 15). Training options are separated into four sections: Model Definition, Labels, Image Augmentation, and Training Settings.

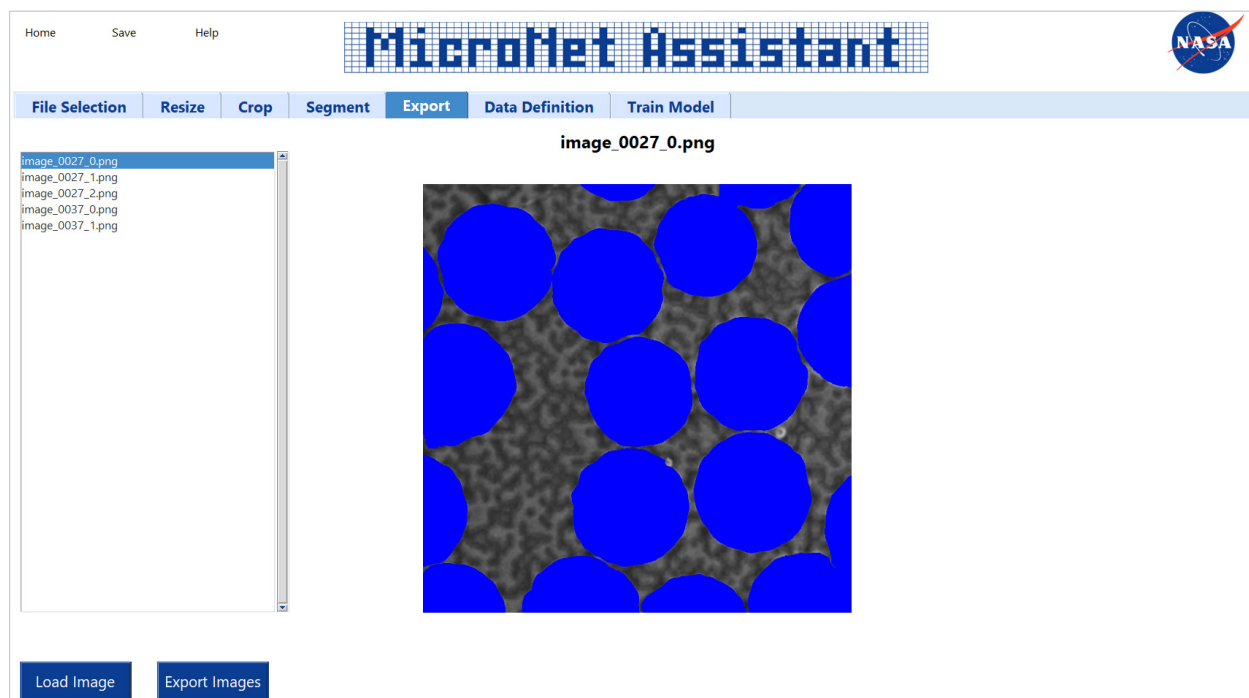


Figure 13.—Export tab.

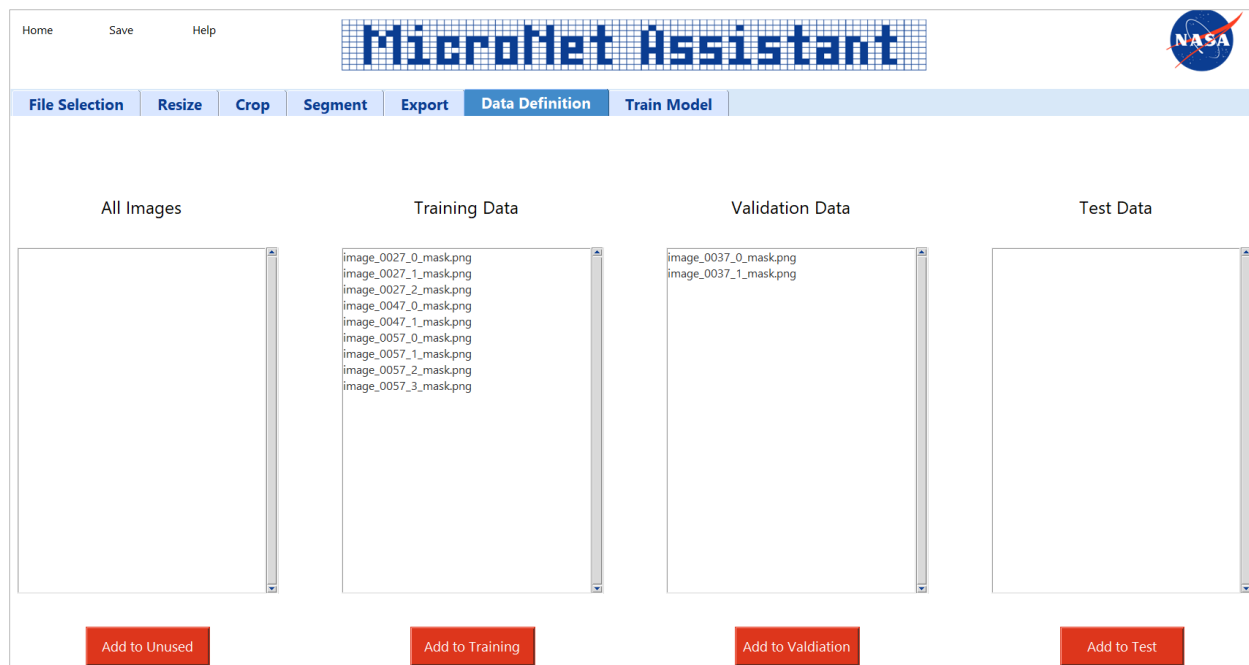


Figure 14.—Data Definition tab.

Home Save Help

MicroNet Assistant

File Selection Resize Crop Segment Export Data Definition **Train Model**

Model Definition

Architecture:

Encoder:

Pretrained Weights:

☐ Use GPU

Image Augmentation

Crop Window:

Option	Probability	Limit
Horizontal Flip	0.5	
Vertical Flip	0.5	
Random Rotate 90°	0.5	
Gaussian Noise	0.5	
CLAHE	1	
Random Brightness	1	0.2
Random Gamma	1	
Shapren	1	
Blur	1	3
Random Contrast	1	0.3
Hue Saturation	1	

Training Settings

Option	Value
Epochs	
Patience	30
Learning Rate	2e-05
Batch Size	6
Validation Batch Size	6

☐ Visualize Training Data

☐ Visualize Validation Data

Labels

Color	Label	
☒	Fiber	☒
☐	Matrix	☒

Train Model

Figure 15.—Train Model tab.

3.8.1 Model Definition

The Model Definition section allows users to define the model architecture, encoder, and pretrained weights for the model (Figure 16). Available architectures are given in Section A.1 (Table 1) and available encoders are given in Section A.2 (Table 2). Pretrained weights can either be from MicroNet alone or from training first on ImageNet (Ref. 4) and then fine-tuned on MicroNet. Additionally, a checkbox option is given to use the computer’s GPU for training. If selected and a GPU is found on the machine, GPU will be used; otherwise, CPU will be used for training.

3.8.2 Labels

The Labels section displays to the user the classes found in the labeled images (Figure 17). Users can name a class using the *Label* column of the table. Additionally, each column can be selected/unselected to be included in the training using the right-most column. If a class is unselected, it will be treated as the background class.

3.8.3 Image Augmentation

The Image Augmentation section allows the user to define the settings for random augmentations applied to each image in the training set (Figure 18). Image augmentation (e.g., randomly rotating the image, changing the contrast, blurring the image, etc.) can improve the robustness of trained segmentation models when given different images. A drop down menu provides the options for the crop window that will be randomly selected for each image in the training set. All test and validation images will have the same window size but cropped based on the center of the image. Entered probabilities (i.e., the probability that the image augmentation method will be applied) for each row must be between 0 and 1. Additionally, Random Brightness, Random Contrast, and Blur have a specified limit input, in which the associated value will be chosen from a uniform distribution between 0 and the limit set. Both the Random Brightness and Random Contrast limit must be between 0 and 1. Blur limit must be greater than 0.

Model Definition

Architecture: UnetPlusPlus

Encoder: inceptionv4

Pretrained Weights: ImageNet + MicroNet

☐ Use GPU

Figure 16.—Model Definition section.

Labels

Color	Label	
Fiber	Fiber	☒
Other	Matrix	☒

Figure 17.—Labels section.

Image Augmentation

Crop Window:

672x672

Option	Probability	Limit
Horizontal Flip	0.5	
Vertical Flip	0.5	
Random Rotate 90°	0.5	
Gaussian Noise	0.5	
CLAHE	1	
Random Brightness	1	0.2
Random Gamma	1	
Shapren	1	
Blur	1	3
Random Contrast	1	0.3
Hue Saturation	1	

Figure 18.—Image Augmentation section.

3.8.4 Training Settings

The Training Settings section allows users to define the settings regarding training the model (Figure 19). In the table, either the epochs or patience must be defined. Setting a value for the epochs row will train the model for a set number of epochs regardless of performance. Setting a value for the patience will train the model until no change in performance is seen for the number of epochs set in that row. If both values are set, only the epochs value will be passed to the training function. Additionally, two checkboxes enable visualizing the Training Data and Validation Data. Checking either box will display each image with its corresponding labels in a new window when the *Train Model* button.

3.8.5 Training the Model

To begin training the model, press the *Train Model* button. The user will be prompted to select the folder that the images were exported to (both original cropped and labeled images). The MicroNet output will be displayed in a separate window to monitor progress during training (Figure 20).

When either the number of epochs or patience criteria is met, the training and validation learning curves will be presented to the user (Figure 21).

The program will then ask the user if they want to continue training. If *Yes* is selected, the model will continue training. If the number of epochs was set, the model will train for the same number of additional epochs. If the patience was set, the learning rate will be reduced by a factor of 0.1 and training will continue until the same patience is met. Once no more training is desired, the user will be prompted to save the model. Models are saved as a *.tar file.

Training Settings

Option	Value
Epochs	10
Patience	
Learning Rate	2e-05
Batch Size	6
Validation Batch Size	6

☐ Visualize Training Data

☐ Visualize Validation Data

Figure 19.—Training Settings section.

```

Training Model

Epoch: 0, lr: 0.00020000, time: 0.00 seconds, patience step: 0, best
iou: 0.0000
train: 0%|

?it/s]train: 0%|

| 0/2 [00:00<?,
| 0/2 [00:51<?, ?it/s, DiceBCELoss - 0.578, iou_sc
ore - 0.5739]train: 50%|#####
#####
| 1/2 [00:51<00:51, 51.70s/it, DiceBCELoss - 0.578
, iou_score - 0.5739]

```

Figure 20.—Monitoring Training.

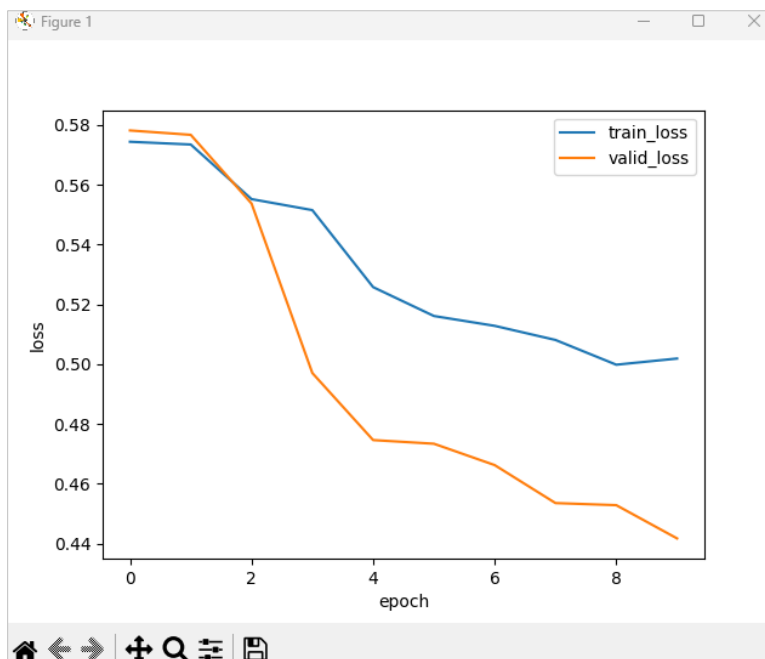


Figure 21.—Learning Curves.

4.0 Model Inference

The Model Inference page allows users to perform model inference by selecting a trained MicroNet model, loading an image, and performing a segmentation. It also enables users to utilize built in tool sets for image post-processing.

4.1 Segmenting an Image

To segment an image using a trained model, both the model and the image to segment must be defined. To load a model, press the *Load Model* button and select a *.tar file containing the MicroNet model (Figure 22). The available classes will display below the model name, as well as a button to display the model information, which will display in a new window when pressed (Figure 23). To load an image,

press the *Load Image* button and select an image. The window will display the image and the dimensions of the image. **Note: users should rescale the image to match scaling performed for training by changing the scale entry box and pressing the *Scale* button.**

When both a model and image are defined, pressing *Segment Image* will perform the segmentation. The segmented image will appear on screen with the option to save or discard the image (Figure 24). Pressing *Save Image* will prompt the user to save the image and pressing *Discard Image* will delete the segmentation and reload the original image.



Figure 22.—Segment Image page.

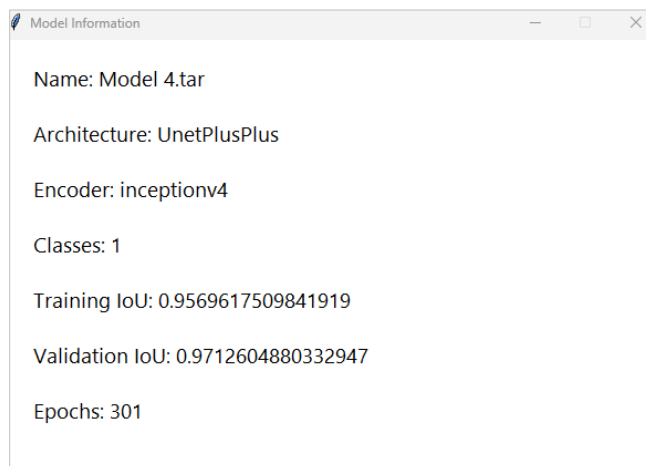


Figure 23.—Model Information window.

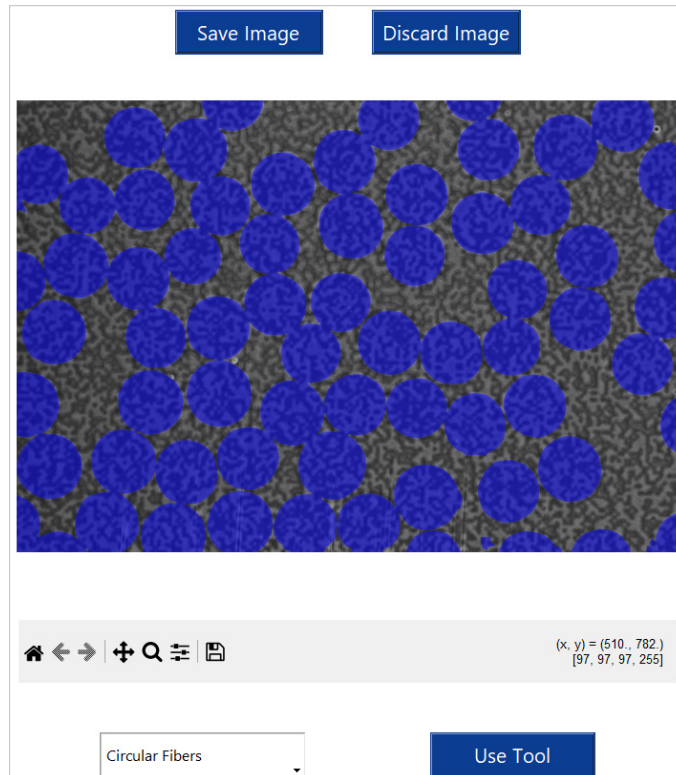


Figure 24.—Viewing a segmented image.

4.2 Post-Processing Toolsets

The MicroNet Assistant Tool contains built-in post-processing tools for idealizing and measuring segmented images. Tools can be accessed when an image is initially loaded or after the segmentation is performed on the Model Inference page. To use a post-processing tool, select a toolset from the drop-down menu at the bottom of the page and press the *Use Tool* button. A tool-specific window will appear with the current image displayed on the Model Inference page.

4.2.1 Circular Fiber Idealization

The Circular Fiber Idealization post-processing tool is used to identify and idealize individual circular fibers in a segmented microstructure and generate a voxelated geometric model for use in analysis tools. When the tool is launched, the image is displayed in a new window with a table for the user to define the reduction size, maximum nub length, maximum corner length, and the color of the fiber (Figure 26). Pressing the *Idealize Fibers* button runs the circular fiber idealization algorithm, which first uses a watershed algorithm to find individual circular bodies in the image and identify the boundaries of each found body. Circles are fit to each boundary using a nonlinear least squares regression algorithm. If fit circles are found to be intersecting, the intersection points are calculated, and the straight line connecting the two points is used as the fiber-fiber interface.

For computational efficiency in analysis tools, it is often useful to have a reduced number of voxels in the geometric model compared to the number of pixels in the image. Once the circular idealization algorithm is completed, the voxel dimensions can be reduced based on the *Reduction Size* defined by the user, maintaining image aspect ratio. Using the interpolation algorithm for determining material assignment can result in undesirable artifacts in the voxelated microstructure, particularly if the

discretization is coarse. ‘Gaps’ can be created at fiber interfaces, where a small number of matrix voxels are completely surrounded by fiber voxels. ‘Nubs’ and ‘corners’ typically occur at fiber boundaries, where a small number of fiber voxels protrude from the fiber body, deviating from the circular representation of fiber. ‘Touching fibers’, where no matrix channel exists between individual fibers, can result in overly stiff regions of the microstructure if contact or debonding is not modeled (Figure 26).

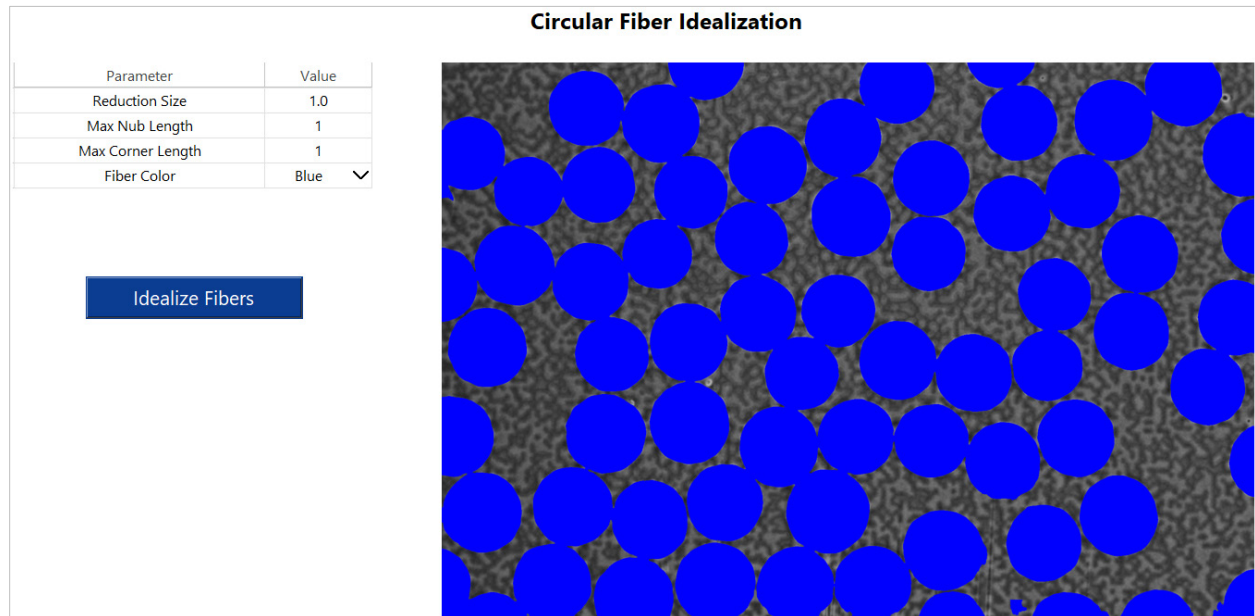


Figure 25.—Circular Fiber Idealization Tool Window

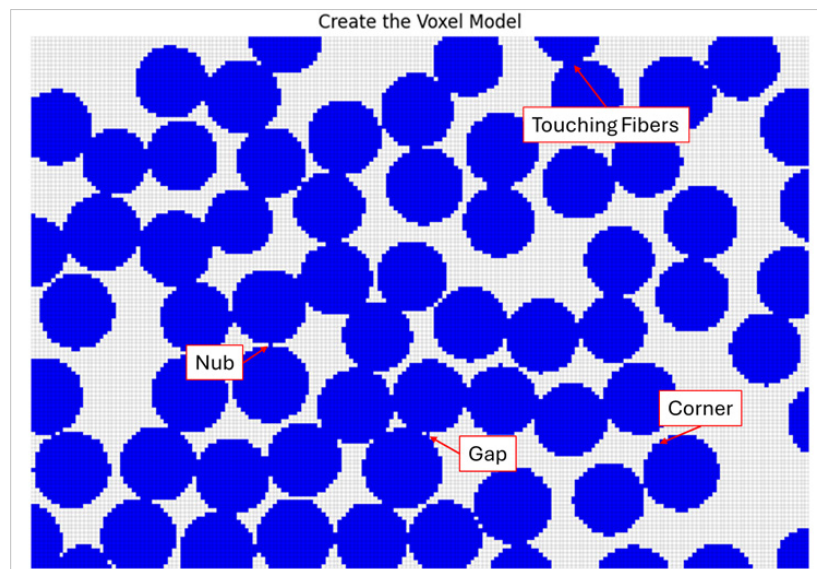


Figure 26.—Definition of gaps, nubs, corners, and touching fibers in the voxelated microstructure.

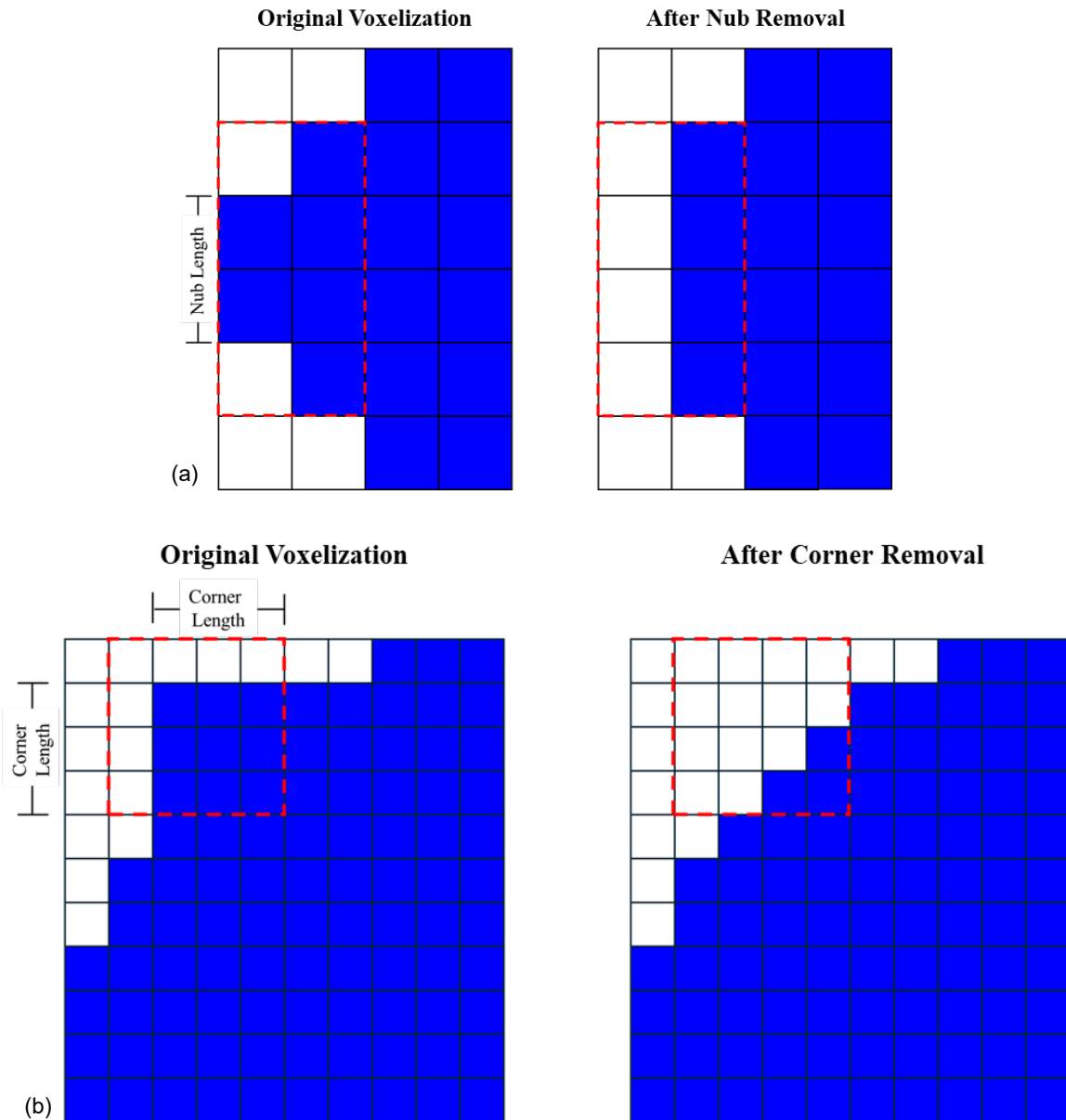


Figure 27.—Pattern recognition and removal algorithm schematic for (a) Nubs and (b) Corners.

Once the image is reduced, additional post-processing algorithms are applied to remove the gaps, nubs, corners, and touching fibers. Gaps are identified by finding the connected components with all matrix subcells surrounded by fibers below a minimum allowable area and are converted to fiber pixels. Nubs and corners are found by defining the voxel pattern that represents each possible orientation and converting the undesired fiber voxels to matrix (Figure 27). The length, or number of voxels to consider as protruding, is defined by the user as the *Max Nub Length* and *Max Corner Length*. Touching fibers are removed by adding a one-voxel width channel between fibers.

When the algorithm finishes running, the idealized image will display in the window with calculated statistics for fiber volume fraction, average fiber radius, image dimensions, and material assignment (Figure 28). Users have the option to save the image or a *.csv file containing the material assignments for use in analysis toolsets.

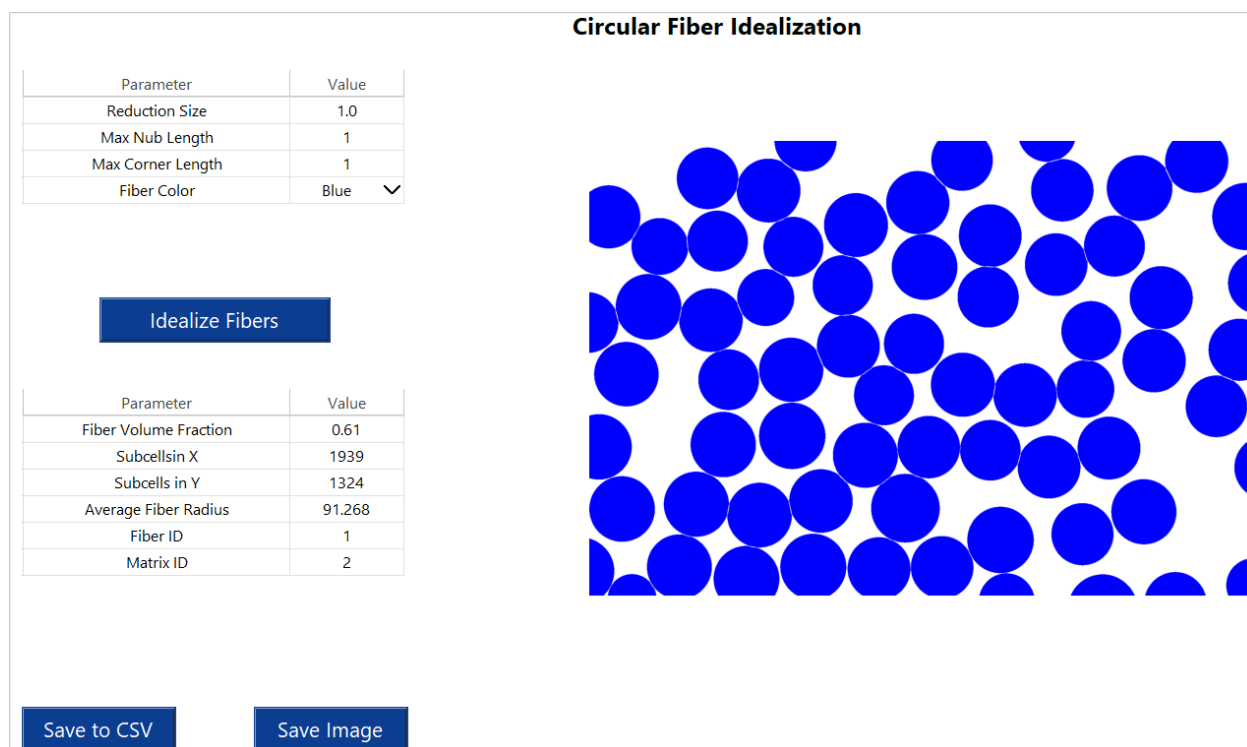


Figure 28.—Circular Fiber Idealization tool output window.

4.2.2 TGO Measurement

The TGO Measurement post-processing tool is used to measure the thickness of thermally grown oxide (TGO) layers between thermal barrier coatings. When the tool is launched, the image is displayed in a new window with a table for the user to define the sample spacing, spline smoothness, maximum normal step, scale, and TGO color (Figure 26). Pressing the *Measure Thickness* button runs two algorithms for determining the TGO thickness: a vertical measurement tool and a normals-based measurement tool. The vertical measurement tool measures the thickness at each sampling point, set by the *Sample Spacing* row. The normals-based measurement tool fits a spline to the top surface, which can be altered by the *Spline Smoothness* row, and determines the normal direction at each sampling point by stepping along the spline by the number of pixels defined in the *Maximum Normal Step* row. The thickness at the sampling point is then the distance along the normal direction until the bottom boundary is intersected.

When running the TGO Measurement algorithms, the TGO layer is automatically identified as the longest continuous segmented area along the image, and thus is robust to errors made in the segmentation model. If a scale bar is present in the image, the scale is automatically identified by algorithm and the conversion between pixels and μm is automatically performed. If a scale bar is not found, the user can either manually define the conversion or continue with the measurement in pixels. When the algorithm finishes running, both the vertical and normals-based results are displayed in the window (Figure 30). For each method, the sampling points and corresponding lines where thickness measurements are taken are displayed on the image. If the scale bar was automatically recognized, it is also identified on the image in green. Below each image, a plot of the thickness as a function of X position is given with the mean and standard deviation calculated. Users have the options to save the results from each plot to a *.csv file, or save the images and plots as an image file.

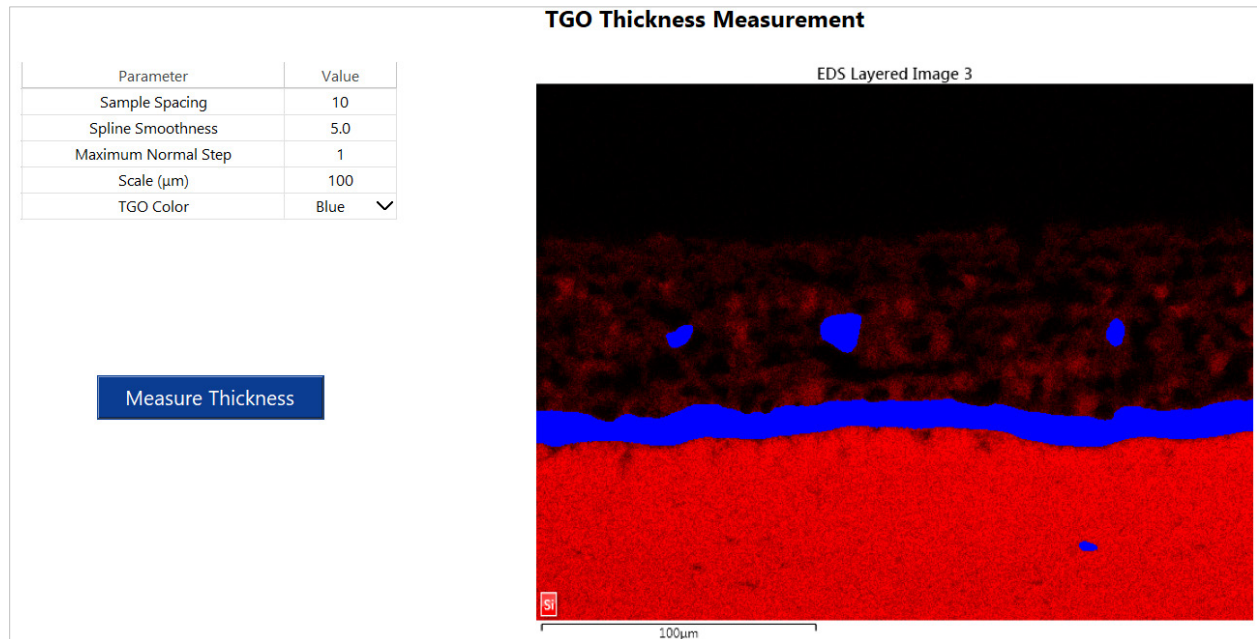


Figure 29.—TGO Measurement Tool window.

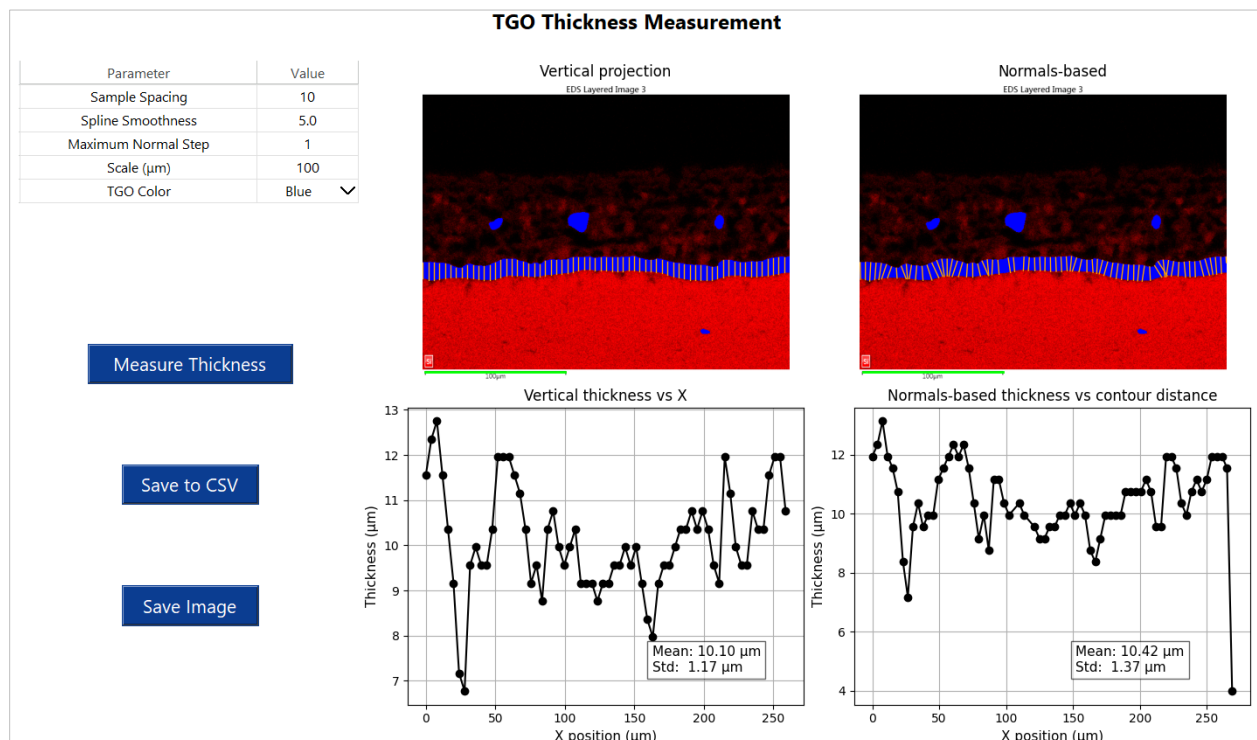


Figure 30.—TGO Measurement Tool output window.

Appendix A.—MicroNet Model Settings

A.1 Available Architectures

Table 1 lists the available architecture options in MicroNet.

A.2 Available Encoders

Table 2 lists the available encoders available in MicroNet and their relative performance. For each encoder, their top 1 and top 5 classification accuracies for the MicroNet benchmark data set is given for each model with pretrained weights determined from MicroNet only and ImageNet and MicroNet combined (trained on ImageNet, fine-tuned with MicroNet) (Ref. 1).

TABLE 1.—AVAILABLE
ARCHITECTURES
IN MICRONET

Available architectures
Unet
UnetPlusPlus
Linknet
FRPN
PSPNet
PAN
DEEPLabV3
DEEPLabV3Plus

TABLE 2.—AVAILABLE ENCODERS IN MICRONET

Encoder	Accuracy 1 – MicroNet, %	Accuracy 5 – MicroNet, %	Accuracy 1 – ImageNet + MicroNet, %	Accuracy 5 – ImageNet + MicroNet, %
densenet121	88.148	98.963	81.185	96.704
densenet161	87.815	99.074	85.963	98.111
densenet169	89.333	99.222	83.815	97.963
densenet201	88.407	99.074	83.741	97.593
dpn107	84.556	98	86.185	98.444
dpn131	84.593	98.296	82.778	97.074
dpn68	77.741	94.815	65.889	87.259
dpn92	74.185	91.815	69.778	89.333
dpn98	85.519	98.407	84.037	97.556
efficientnet-b0	83.926	97.444	92.815	99.778
efficientnet-b1	84.111	97.815	93.259	99.741
efficientnet-b2	84.63	98.111	93.741	99.889
efficientnet-b3	84.889	97.667	93.889	99.741
efficientnet-b4	84.519	97.185	94.519	99.741
efficientnet-b5	83.148	97.074	93.926	99.778
inceptionresnetv2	90.926	99.296	92.148	99.63
inceptionv4	93.63	99.704	93.741	99.815
mobilenet_v2	83.815	97.815	80.556	96.037
resnet101	77.296	94.704	86.259	98.222
resnet152	85.63	98.185	85.111	97.296
resnet18	79.815	95.667	81.185	96.926
resnet34	77.259	94.444	90.185	99.222
resnet50	62.037	83.741	90.259	99
resnext101_32x8d	87.556	99.037	92.815	99.778
resnext50_32x4d	69.037	89.296	91.148	99.407
se_resnet101	93.37	99.741	93.185	99.778
se_resnet152	92.926	99.852	93.444	99.778
se_resnet50	93.222	99.741	93.222	99.741
se_resnext101_32x4d	93.889	99.815	94.519	99.852
se_resnext50_32x4d	93.741	99.852	93.63	99.815
senet154	94.037	99.741	93.741	99.815
vgg11_bn	76.815	93.296	89.148	98.889
vgg13_bn	77.889	93.704	90.63	99.481
vgg16_bn	71.481	90.926	90.222	99.37
xception	93.815	99.63	93.444	99.741

Appendix B.—Index

batch size.....	17	MicroNet.....	3
Brush Tool	11	available encoders.....	28
Brush/Eraser Size.....	11	available model architectures	27
Circular Fiber Idealization	20	cropped image requirements.....	7
corners.....	21	ideal image dimensions	6
gaps	21	training set size	13
nubs.....	21	validaiton set size.....	13
touching fibers	21	model architecture	15, 27
composites	20	Model Creator page	4
Crop Images tab	7	Model Inference page	18
Data Definition tab.....	13	patience.....	16
encoder.....	15, 28	Post-processing	2, 20
epochs	16	pretrained weights.....	15
Erase Tool	11	Resize Images tab	6
Export tab.....	13	Segment Anything Model.....	3, 10
exporting images	13	Segment Images tab.....	9
File Selection tab.....	5	selecting classes for training.....	15
Home page	4	test data	13
image annotation		TGO Measurement	23
background label.....	10	normals-based measurement.....	23
defining multiple classes.....	12	vertical measurement	23
foreground label	10	thermally grown oxide.....	23
manual annotation.....	11	Train Model tab	14
prompt-based annotation.....	10	training data	13
image augmentation	15	using a trained model.....	18
learning curves	18	validation batch size	17
learning rate	17	validation data.....	13
		voxels.....	21

References

1. J. Stuckner, B. Harder and T. M. Smith, “Microstructure segmentation with deep learning encoders pre-trained on a large microscopy dataset,” *npj Computational Materials*, vol. 8, no. 200, 2022.
2. J. Stuckner, K. Fre, I. McCue, M. J. Demkowicz and M. Murayama, “AQUAMI: An open source Python package and GUI for the automatic quantitative analysis of morphologically complex multiphase materials,” *Comput. Mater. Sci.*, vol. 139, pp. 320-329, 2017.
3. A. Kirillov, E. Mintun, N. Ravi, H. Mao, R. C. L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W. Y. Lo and P. G. R. Dollar, “Segment Anything,” in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.
4. J. Deng, et al., “Imagenet: A large-scale hierarchical image database,” *2009 IEEE conference on computer vision and pattern recognition*, pp. 248-255, 2009.

