**Quick Overview of Semantic Segmentation to TPS workflow**

1. Start by creating a folder with the 5 necessary scripts (3 python and 2 R) and this help doc

WorkingDirectory

* readme.docx
* transfermodel\_utils.py
* mask\_from\_tps.R
* tps-oo.R
* train\_transferlearning.py
* make\_predictions.py

1. Run the PrepareData() function in transfermodel\_utils.py, which will restructure the data for model training like this:

Working Directory

* readme.docx
* transfermodel\_utils.py
* mask\_from\_tps.R
* tps-oo.R
* train\_transferlearning.py
* make\_predictions.py
* Training
  + GENUS\_SPECIES-Pop#-Specimen#
    - images
      * GENUS\_SPECIES-Pop#-Specimen#.png
    - masks
      * GENUS\_SPECIES-Pop#-Specimen#.png
  + GENUS\_SPECIES-1-02
    - images
      * GENUS\_SPECIES-1-02.png
    - masks
      * GENUS\_SPECIES-1-02.png
* Test
  + GENUS\_SPECIES-1-01.png
  + GENUS\_SPECIES-1-02.png

1. Run train\_transferlearning.py to produce an .h5 file of model weights

Working Directory

* readme.docx
* transfermodel\_utils.py
* mask\_from\_tps.R
* tps-oo.R
* train\_transferlearning.py
* make\_predictions.py
* Training
* Test
* transfer\_model.h5

1. Run make\_predictions.py to use the model on a folder of images not used in training

Working Directory

* readme.docx
* transfermodel\_utils.py
* mask\_from\_tps.R
* tps-oo.R
* train\_transferlearning.py
* make\_predictions.py
* Training
* Test
* transfer\_model.h5
* Contours
  + contour\_GENUS\_SPECIES-1-01.txt
  + contour\_GENUS\_SPECIES-1-02.txt

1. Use the WriteMultipletoTPS() function in transfermodel\_utils.py to create 1 TPS file for all the TPS files in Contours

Working Directory

* readme.docx
* transfermodel\_utils.py
* mask\_from\_tps.R
* tps-oo.R
* train\_transferlearning.py
* make\_predictions.py
* Training
* Test
* transfer\_model.h5
* Contours
* TPS
  + GENUS\_SPECIES-1.tps

**Details**

* These scripts were developed with Python 3.7, Tensorflow 1.15, R 3.6.1, Keras 2.3.1, Pillow 7.1.2, matplotlib 3.2.1, numpy 1.18.5, opencv 4.2.0.34, py\_tps 0.1.7, scipy 1.4.1, segmentation-models 1.0.1, tensorboard 1.15.0

**transfermodel\_utils.py**

* Dependencies: os, glob, PIL (a.k.a. Pillow), tensorflow, numpy, py\_tps, matplotlib, scipy, cv2, subprocess, and tqdm
* This script contains 6 functions: MakeFolders(), TIFtoPNG(), MakeMasks(), ResizeTestImages(), PrepareData(), and WriteMultipletoTPS()
* To make preparing the data for model training easier, PrepareData() runs MakeFolders(), TIFtoPNG(), MakeMasks(), and ResizeTestImages() all at once
* These functions share many of the same arguments:
  + PathtoImages: the full path name to the TIFF images that will be used in model training
  + WorkingDirectory: the folder containing all the necessary python and R scripts and where all outputs will be saved
  + PathtoTestImages: the full path name to different TIFF images on which the model will be tested
  + ImageSize: the width or height in pixels images should be for model training
  + \*TPS\_files: the tps files corresponding to the TIFF images that will be used in model training. There can be any number of TPS files, and they should be separated by commas
  + Crop: Images must be square for model training, so they will be resized either by padding with zeros (adding black space) or cropping. If Crop=False (default), padding will be used, and vice versa
  + FolderofContourFiles: after a model has been trained and used to predict contours on a population of images, this should be the full path name to that folder of contour files
  + PopName: what the final TPS file should be named, our convention in GENUS\_SPECIES-Pop#
  + Scale: the scale at which the TIFF images were originally taken, which should be written in the TPS file
* MakeMasks() uses mask\_from\_tps.R and tps-oo.R to make mask PNG images from tps files

**train\_transferlearning.py**

* Dependencies: os, sys, csv, argparse, time, datetime, random, glob, warnings, cv2, numpy, pandas, matplotlib, PIL (a.k.a. Pillow), tqdm, itertools, skimage, keras, tensorflow, segmentation\_models, tensorboard
* This script can be run from command line or a python interface. If using a python interface, the parameters can be entered in lines 72-81 of the script. If using command line, the arguments are:
  + --IMG\_WIDTH: the width and height in pixels of images the model takes (must be square)
  + --TRAIN\_PATH: the path to the folder containing every specimen (~WorkingFolder/Training)
  + --TEST\_PATH: the path to the folder of images on which the model will be tested (~/WorkingFolder/Test)
  + --WORKING\_DIR: the path to the folder containing all the python and R scripts (~/WorkingFolder)
  + --MODEL: what you would like to name the outputted .h5 model weights file (default is “transfer\_model.h5”)
  + --SET\_SEED: whether or not you would like to make the model training replicable (boolean, default=False)
  + --BACKBONE: the name of the model structure you would like to use (see segmentation\_models library, default is ‘vgg16’)
  + --PATIENCE: number of epochs without improvement before training stops early (default=20)
  + --VAL\_SPLIT: fraction of images that will be used to evaluate the model rather than train the model (default=0.2)
  + --BATCH\_SIZE: number of images trained on at once (default=40)
  + --EPOCHS: number of rounds the model should train on every image (default=100)
* After training is complete, tensorboard will print a hyperlink to your model diagnostics
* Troubleshooting: tensorboard may prevent this script from running via command line. If so, comment out the lines “import tensorboard / !rm -rf ./logs/ / %load\_ext tensorboard / %tensorboard –logdir logs –host=127.0.0.1”

**make\_predictions.py**

* Dependencies: os, sys, random, warnings, cv2, glob, argparse, numpy, pandas, matplotlib, tqdm, itertools, skimage, scipy, keras, tensorflow, PIL (a.k.a. Pillow), segmentation\_models
* This script can be run from command line or a python interface. If using a python interface, the parameters can be entered in lines 50-55 of the script. If using command line, the arguments are:
  + --TEST\_FOLDER: path to folder of images on which the model will be used (~/WorkingFolder/Test)
  + --MODEL\_PATH: path to the .h5 model weights file (~/WorkingFolder/transfer\_model.h5)
  + --WORKING\_DIR: path to folder containing all the python and R scripts (~/WorkingFolder)
  + --INIT\_IMG\_WIDTH: width in pixels of images on which model was trained
  + --FIN\_IMG\_WIDTH: original width in pixels of images before resizing
  + –BACKBONE: the name of the model structure used in model training (default = ‘vgg16’)
* Troubleshooting: if the model is poorly trained and no contours are extractable, line 113 will produce an error

**Sources**

* <https://medium.com/@pallawi.ds/semantic-segmentation-with-u-net-train-and-test-on-your-custom-data-in-keras-39e4f972ec89>
* <https://segmentation-models.readthedocs.io/en/latest/api.html>
* <https://dev.widemeadows.de/2016/12/12/getting-an-image-into-and-out-of-tensorflow/>