Using our model for your own images

Karina Guo

Nov. 2023

Introduction

This file helps users track through our repository and download the relevant pieces of data to use our models on their own leaf images. It starts with downloading relevant files, setting up the directory, creating conda environments and running the script. We would like to note that the data downloaded may not all be used for this tutorial.

All files referred here are available at our GitHub or Zenodo repository

Downloading relevant files

1. Download the conda environments from the repository at *Machine_Learning_in_Computer_Vision_for_Leaf_Traits/Conda Environments/*
2. Download the models at *Machine_Learning_in_Computer_Vision_for_Leaf_Traits/Data/models/*
3. Download code from *Machine_Learning_in_Computer_Vision_for_Leaf_Traits /Code/ML_models/*

Directory setup + Running the scripts

Set up the file directory

You want to have the following folders in your working directory:

- */data* – This is an empty directory and was used by myself to create the model, it is not directly called in the prediction script but remains a historical fossil of technical development
- */test* – Place the **.jpg** images that the model will predict on
- */pred* – This is the directory the model will output its visual and quantitative predictions to
- */model* – Directory of models downloaded
- */code* - Directory of code downloaded

Once set up, move the downloaded files into their respective folder

Import conda environments

The conda environments include the packages required to run the scripts. Create them by running:

```
conda env create -f *.yml
```

Where *.yml represent the names of the environments downloaded from the repository

Changing the prediction script

Open up *code/predict_leaf.py*, here you'll be changing the file paths highlighted along with removing some lines of script. We are removing lines of scripts as there is a validation process, which matches the predictions to the ground truth leaves I measured on my data. You would not have this, please contact me if you would like to generate this. Refer to the repository for an example of the file paths below.

```
import sys
sys.path.append("code/") # Directory of /code with model_tools.py within
import model_tools

img_dir='' # File path to the directory of your images to be predicted on
out_dir='' # File path to the directory of image predictions
fext = "jpg" # Ensure your files are in .jpg format
training_path="" # File path to your current working directory
training_name="data"
yaml_file = "model/model.yaml" # File path of downloaded .yaml file
weights_file = "model/model_final.pth" # File path of downloaded .pth file
yaml_zoo=False
weights_zoo=False
num_classes=1
model_tools.visualize_predictions(img_dir, out_dir, fext, training_path,
training_name, yaml_file, weights_file, yaml_zoo, weights_zoo, num_classes,
score_thresh=0.7, s1=10, s2=14)
model_tools.predict_in_directory(img_dir, out_dir, fext, yaml_file, weights_file,
yaml_zoo, weights_zoo, num_classes, score_thresh=0.7)

groundtruth_name="test"
model_predictions_file = "pred/_predictions.npy" # Location of desired output of
predicted numpy binary matrix
thresh_iou = 0
matches_out_file="pred/model_matches.csv"

model_tools.match_groundtruth_prediction(groundtruth_name, training_path,
model_predictions_file, thresh_iou, matches_out_file)

### summarise a set of predictions
instances_out_file="/data/botml/leaf_dimension/EIGHT_DuplSeven_BS20_ExtTrain/pred/mod
el_summary.csv"
model_tools.predictions_summary(model_predictions_file, instances_out_file)
```

Load conda environment

For this section, you will only need the conda environment pytorch, to activate this run:

```
conda activate pytorch
```

Then, run the *predict_leaf.py* script

```
python predict_leaf.py
```

This script will generate a numpy matrix that will include the pixels of the leaves (*_predictions.npy*) and a .csv file that includes the pixels of the predicted leaves (*model_summary.csv)*. Note the latter file was used for a quick evaluation of the quality of our model by comparing the predicted leaf area to the ground truth leaf area I manually measured. Our final run of the models also conducted connected component analysis and constructing a convex hull around the pixels. Our script for this is available at the GitHub/Zenodo repository

/Machine_Learning_in_Computer_Vision_for_Leaf_Traits/Code/final_run/extract_leaves_mt.py & *leaf_dimension_calculations.R* . These scripts can be run using the conda environment MLpredictions

Our model iterations also include more than one classification of mask (e.g., Leaf50, Leaf90, Leaf100), however the script above only has 1 classification, and is reflected in *model_summary.csv*.

Converting to leaf area

Leaf area is represented in the *model_summary.csv* as pixels. To convert the pixels to centimetre^2, use ImageJ, or another similar software, to measure how many pixels represent a known distance on your image. Commonly employed technique is either, 1. Scanning your leaf images with a 1 cm grid on the side, 2. Scanning the leaf images at a known resolution.

Additionally, our workflow had two model used in succession of each other. The second model after this leaf masking step removed predicted masks that did not look like leaves. This can be employed using the script in the repository and involves first converting the masks to images, then using the classifier to remove undesired predictions, and only then calculating the leaf area.

As this tutorial does not currently include the integration of the second model, it is suggested by the author that the user filters out any suspiciously small/large leaves.