pyleaf Documentation

Release 1.0.0

Vishal Sonawane

Sep 19, 2020

CONTENTS

sphinx-quickstart on Mon Dec 3 19:04:20 2018. You can adapt this file completely to your liking, but it should at least contain the root *toctree* directive.

CHAPTER

ONE

LEAFAREACALCULATOR

1.1 basefunctions module

basefunctions.**CNNModel**()

Convolutional Neural network model. Input is 256 vector input dimensional array as Embedding layer. Additional layers: Conv1d, MaxPooling, Dense, Flatten,Dense.

Type of class label is categorical crossentropy. Adam Optimizer is selected. Display metric is accuracy.

Returns CNN model which is a prototype of the classification model specified in this method.

Return type keras.model

basefunctions.**PROCESS_IMAGE**(*sample_name_list*, *path_*, *red_square*)

Accept the image file name from LeafAreaCalculator and prcoess the passed image by cobverting it to model input vector and processing all pixels in the image. CNN model ouptput class color labels for every pixel. Simple math is then used to calculate leaf area(green pixels) from green:red ratio of all classified labels.

Parameters

- **sample_name_list** (*string*) – image file name to process
- **path** (*string*) – default path to search for image

Returns list of useful parameters: image_file_name, area in cm^2 and ** optional use parameters.

Return type List

basefunctions.**RETRAIN_MODEL_FROM_SCRATCH**()

Retrain the CNN model process entirely from scratch. This is action taken when use uses Tools->Retrain model from GUI menu.

basefunctions.**alter_calc**(*image*)

EXtract pixel level information from the passed image. Retrieved (r,g,b) pixel values for every pixel of the image and convert them to hsv values, for easier seperation of green colored pixels.

Parameters **image** (*string*) – name of image file

Returns List of pixels scanned across the image left top right, top to bottom.

Return type List of tuples

basefunctions.**generateClassLabels**(*sample*, *color*)

Generates class labels for all the pixels extracted from an image. Example: if red.png was fetched, all pixels of this image are labelled 'red'. Likewise, for all images in the default training folder (test_images/) In our default environment, we have green for all kinds of leaf images, red for all red squared, and any other is irrelevant to pur computations.

basefunctions.**getPixelFromLabel**(*label*)
Function to get the pizel from the class label. This reverse mapping is use to generate an image file fom available class labels.

Parameters **label** (*int*) – class label

Returns (r,g,b) equivalent of class label color.

Return type tuple

basefunctions.**image_area_calculator**(*basepath*, *sample*)
Process the desired image, and extract the pixel level information of every pixel in the image in (h,s,v) format. Utlizes alter_calc method to generate (h,s,v) values.

Parameters

- **basepath** (*string*) – default path of images
- **sample** (*string*) – name of image file

Returns returns the vector embedding which will be input to CNN model.

Return type list of lists

basefunctions.**kerasTokenizerUnit**(*topbestwords*)
Convert the (h,s,v) value of every pixel to a 256 dimensional feature vector that serves as an input to the CNN EMbedding layer.

Parameters **topbestwords** – maximum dimesion of vector, default: 256

:type integer :return: 256 dimensional vector :rtype: np.array

basefunctions.**load_model**()
Loads the already available model previously saved on disk.

basefunctions.**processInputTrain**()
Function that extracts all training images from default traininf images foled (test_images/) and converts them into required input vector format for CNN embedding layer.

basefunctions.**readFromDisk**()
Reads the previously sored pixel/class label values saved as pickled objects.

basefunctions.**regular**(*string_list*, *class_labels_norm*, *finalSequence_*)
Trains CNN model with the passed embedding vectors of the image. CNN model is trained with 600*600 image dimesions, so 360000 data pointes per training image, for all images in default training images directory, for 10 iterations. Note: At the time of documenting, the acuarcy of model achieved was 99.5%.

Parameters

- **string_list** – Dummy variable. Not used.
- **class_labels_norm** (*Vector*) – Normalized class labels for the colors.
- **finalSequence** (*List of vectors.*) – Input embedding vectors for image

basefunctions.**saveModelToDisk**(*model*)
Saves the currenly used CNN model configuration to disk.

Parameters **model** (*keras.CNN model*) – current CNN model

basefunctions.**saveToDisk**(*string_list*, *class_labels_norm*, *finalSequence_*)
Saves CNN model configuration and input/output pixel/class_label values to disk for later use. Utilizes pickle objects for compressed serialized storage.

1.2 leaf_area_calculator_gui module

class leaf_area_calculator_gui.**LeafAreaCalculatorGUI**(*root*)
Bases: object

Class LeafAreaCalculatorGUI is a Tkinter GUI manager, that is responsible for interacting with the user and displaying results. The backend application is a convulutional neural network that is trained with green, red and other colors, to generate a robust and reliable classifier that can detect green leaf within a given image in varying light conditions and leaf shades, and computes the area of the green leaf with respect to a fixed red square of 4 cm^2.

GUI includes two preview panes: left pane is the original window pane, that displays the original image and the name of image. right pane is the updated window pane, that displays the resultant analyzd image from classifier.

Under the left pane is a navigation bar, with which user can switch left or right through multiple images. This option is diabled if user is working on only one image.

About()
Displays the File->About pop up box.

IMAGE_HEIGHT = 300

IMAGE_WIDTH = 300

answer(*x*)
Allows user to select base image path from where all images in that folder can be polled for batch analysis. When user selects the default folder through the dialog box, all images in the selected folder are loaded in the original preview pane.

Parameters **x** (*Tk()*) – Tkinter Instance of main window or root

change_progress()
Update the progress value in the progress window.

change_result_preview(*every*)
Update the result window pane with the passed image.

Parameters **every** (*string*) – name of the image file to load.

change_unit(*event*)
Function that allows the numerical leaf area result to switch between cm^2 and mm^2. The result is displayed in Leaf Area label field.

Parameters **event** (*Tkinter Button Press 1.*) – Accept the button click event.

Returns Return 'NA' back if 'NA' is clicked.

Return type int

disable_red_button(*event*)
Disable the Set Red Button.

display_barcode(*filename*)

display_left()
The Left Arrow button of the navigation control under original preview pane. Disabled if navigation reaches 1/N , Enabled otherwise.

display_result(*area=None*)
Displays the area in cm^2 with an option to switch to mm^2 by clicking on the numerical result displayed.

Parameters **area** (*float, 2 bit decimal.*) – The area to display in the 'Leaf Area:' field.

display_right()
The Right Arrow button of the navigation control under original preview pane. Disabled if navigation reaches N/N , Enabled otherwise.

display_this_image(*passed_image*)
Function that collects a passed image from disk and displays in the original preview window and updated result window.

Parameters **passed_image** – The image name to be fetched from default_image_path and default_save_path

documentation()

new_winF()
Function that creates a progress window to display progress of leaf analysis with multiple images. A new window updates the status with analysis of every leaf in the default_image_folder, and closes automatically when program processes all images.

The window is positioned to be in the centre of the main window.

process_all()
Function that is triggered when Batch Process feature is activated by user. Iterate over all images one by one from the list of images in default_image_path, and update the Leaf Area and preview panes as the results become available. While the operation is in progress, new window is popped that displays dynamic progress of task.

process_image(*every=None*)
Function that accepts only one image name as input, and performs analysis. Updates the Leaf Area and preview panes as the result become available. If no input is provides, iteratively processes all images in the list maintained by class as list of all images in the default image path.

Parameters **every** (*string*) – Image name to process

reset_workspace()
Resets the workspace to defaults. This provides a fresh GUI configuration to user. Clears all images in the default save path directory.

retrain_model()
Function retrain_model: Performs re-training of the classifier on user input from Tools->Retrain model. Note: User should retrain the model only if training images have changed, or classification model seems corrupt.

retrain_model_run()
Function retrain_model_run: Displays model training status to user. A window is popped up with message: Model Training In Process. Window is closed when model training is complete.

save_results()
Function to display the save recent results to location of user choise. Displays results available in stored-MeasuredValues.csv data file.

setDefaultImageLocation()
Function setDefaultImageLocation: Sets the default image directory to find images for analysis.

set_grid()
Creates a blueprint of the various GUI elements in a window of size 820x500. Tkinter Frame is creates and place geometry manager is used to position the elements exactly where desired. Positions labels ans preview panes in the desired locations, as per the GUI guidelines of this project.

set_menus()
Function set_menus: Sets the Menu tool bar for the application. User can select default image directory and perform batch process usind File and Tools menu options.

update_original_preview(*photo*)
Update the original window pane with the passed image.

Parameters **photo** (*string*) – name of the image file to load.

view_results()
Function to display the recently saved results in a new window. Displays results available in storedMeasuredValues.csv data file.

CHAPTER

TWO

INSTALLATION

Requirement: Python 3 Refer the downloaded Python installer from official PyPI page:

https://pypi.org/project/pyleafarea/#files

```
pip3/pip install pyleafarea
```

You should now be able to see the package pyleaf in site_packages of your Python interpreter.

CHAPTER

THREE

USAGE

```
python3/python -m pyleaf
```

CHAPTER

FOUR

GUI FEATURES

On the GUI Application, you will find various options to get you started right away:

```
Batch Process:
Select default image folder from File->Select Default Image Path
Then select Tools->Batch Process to run leaf area analysis on all images in chosen␣
↪folder.

Single Image Analysis:
Use Select Images button to select one or more image(s) and click Process Images to␣
↪view the results.

Use the navigation page under original window preview to load all selected images.

Reset Workspace button clears the workspace for a fresh start.

Tools->Retrain Model to re-train CNN clasification model based on updated training␣
↪images in test_images/ folder.

Tools->View Recent Results to open up a window and view the image and their␣
↪calculated areas in tabular format.
```

CHAPTER

FIVE

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

b

l