Tutorial in MedHSI

foxel

July 4, 2022

1 Introduction

This repository contains code written in two main languages MATLAB and Python. The structure is as follows.

medHSImat contains the MATLAB code and is mostly used for data preprocessing, machine learning and graph creation.

medHSIpy contains the python code and is mostly used for deep learning applications.

conf contains all the configuration parameters to make the code work.

parameters contains useful parameters for calculations.

2 Setup and Initialization

1. Open text file 'medHSI/conf/config.ini' and update it according to your folder structure. Common items that need update on a new machine are:

   - DataDir : the data where .hsm and .h5 files from data collection are saved.
   - ExecutionDir : the folder where MedHSI is located.
   - ImportDir : the folder where additional import files are located.
   - OutputDir : the folder where output results will be saved.
   - MatDir : the folder where preprocessed data (.mat files) from the analysis will be saved, the database.
   - ParamDir : delete

   In this case, 'Database' refers to data collected from the same system, e.g. the calibration system, or the current system in the pathology lab. On the other hand, 'Dataset' refers to a subset of the 'Database', for example 'Dataset = pslRaw' refers only to raw samples excluding fixed samples from the 'psl' Database.

   If mat files for a certain dataset are already prepared in MatDir, then you can change 'Dataset' to the target dataset. You don't need to change 'DataDate'.

2. Copy the files from the supplementary material to the respective folders. You can overwrite previous files.

   - **parameters** to folder medHSI/parameters/
   - **import** to the folder set in config::[ImportDir] of medHSI/conf/config.ini

3. Set up MATLAB by installing necessary prerequisite packages. Open MATLAB and navigage to medHSI/medHSImat/setup/ and follow the instructions in README.md.

4. From here on the term **config::[XXX]** will refer to setting with name XXX in the config.ini file.

3 medHSImat: for MATLAB

3.1 Read a new database

1. Convert .hsm files to .h5 using the program for the 2D spectroradiometer. Save them as .h5 files in folder '01-Measurements' inside directory conf::[DataDir].

2. Add relevant information in config::[ImportDir]/ **pslDBDataInfoTable.xlsx** and **pslDBDiagnosisInfoTable.xlsx**. You can check pslDBDataInfoTable.xlsx for missing samples or misnamed files with the following.

```
#MATLAB
[flag, fileS] = initUtility.CheckImportData();
```

3. Check that the **Preprocessing.m** function matches your preprocessing scheme. If not, update it.

4. Read the .h5 files, preprocess and convert to class **hsi** (https://foxelas.github.io/medHSIdocs/classhsi.html). The class includes a foreground mask and other information.

   These conditions read all data that in **pslDBDataInfoTable.xlsx** are set as 'tissue' and as 'raw', so all unfixed tissue samples. A foreground mask is also extracted.

```
#MATLAB
contentConditions = {'tissue', true};
readForeground = true;
targetConditions = {'raw', false};
hsiUtility.PrepareDataset('pslRaw', contentConditions, readForeground,
    targetConditions);
```

5. Prepare labels using the LABELME tool. Save them as black and white images in folder '02-Labels' inside directory config::[DataDir].

   How to make labels with labelme:

   - Check https://github.com/wkentaro/labelme. It is a python based tool.
   - To install in anaconda run the following in a conda prompt.

```
# python3
conda create --name=labelme python=3
source activate labelme
# conda install -c conda-forge pyside2
# conda install pyqt
# pip install pyqt5 \# pyqt5 can be installed via pip on python3
pip install labelme
# or you can install everything by conda command
```

```
# conda install labelme -c conda-forge
```

* Suggested python version =3.9.

* You can check the tutorial https://github.com/wkentaro/labelme/tree/main/examples/tutorial.

* If it fails, check that visual studio c++ is included or that the python version is correct.

- Open the GUI by running the following

```
# Prompt
labelme # just open gui
```

- From the GUI, open the directory with the images you want to label. Use the tool to draw polygon masks on the images. Each label is saved as a .JSON file in the same folder.

  * In this case the target directory will be the snapshots in config::[Output]/XXX/00-Snapshots/,
  where XXX refers to the target dataset.

- Use commands:
  Make label:

```
# Prompt
labelme 001.jpg -O 001.json
```

  View label:

```
labelme_draw_json 001.json
```

  Export label:

```
# Prompt
labelme_json_to_dataset 001.json -o 001_json
```

- With anaconda make a for loop to prepare label images out of each .JSON file. The produced results will be saved in subfolders in the same directory as the .JSON files.

```
# Prompt
for %a in (150, 157, 160, 163) do labelme_json_to_dataset %a.json -o %a
```

- Update the database .mat files with the new labels.

```
# MATLAB
init.UpdateLabelInfos(dataset);
```

- Now the dataset is ready. You can read it as a list of hsi and hsiInfo classes.

```
# MATLAB
%% Load the entire dataset as a list of hsi and hsiInfo classes
[hsiList, labelInfoList] = hsiUtility.LoadDataset();
```

- You can also export the dataset as an .h5 structure, split it into patches or resize to a standard size with zero padding.

```
# MATLAB
%% Export an .h5 database of all the preprocessed and labeled data
% After reading, the database is saved in
    config::[OutputDir]\\config::[Dataset]\\*.h5.
hsiUtility.ExportH5Dataset();

%% Make subdatasets from the original dataset
% Make 32x32 patches
targetDataset = '32';
initUtility.MakeDataset(targetDataset, dataset);


% Make 512x512 padded squares
targetDataset = '512';
initUtility.MakeDataset(targetDataset, dataset);
```

  You can also prepare data validation folds (Leave-one-out, K-fold, custom) or apply functions individually on each image.

- For more information check
  -setup/demos/Tutorial_ReadDataset.m
  -setup/demos/Tutorial_ProcessHSI.m