SLSTR Validation Tool

This is a README file to describe all the functions and python files developed for the validation of SLSTR products concerning CLOUD Flagging data. Most of the work done so far takes into account how SLSTR product files are designed and organized. The SLSTR_package consists of 5 python files: gui.py; image_loading.py; patch_extraction.py; rad_to_reflectance.py; unpack_and_show.py. Before going to each python file, a description of SLTSTR validation use cases is given. It is also needed to say that some of the steps below are necessary to run algorithms for the validation of the SLSTR cloud masks.

1. SLSTR Use Cases

- After downloading and uncompressing of SLSTR product files from https://scihub.copernicus.eu/dhus/#/home, the overall organization of each product includes a all files concerning all SLSTR bands, some ancillary files and flagging data.
- Bands from S1 up to S6 are provided with Top of Atmosphere (ToA) in Radiances while S7-S9 and F1-F2 are given in Brightness Temperature (BT).
- Since it is needed to make band data comparisons coherent, a radiance to reflectance conversion is applied over S1-S6 bands. As mentioned in S3_SLSTR_HANDBOOK "a more useful unit to compare sensors is normalised reflectance, which can be generated from the radiance using the following formula:

  reflectance = PI * (ToA radiance /solar irradiance/ COS(solar zenith angle)) (1)

- The solar irradiance employed in equation (1) can be retrieved from the "quality" dataset for each respective channel (S1-S6).
- In the SLSTR Use Cases, it is thought users might be interested in analyzing one or more SLSTR products.
- It is suggested to unpack all SLSTR product files of interest in the same folder, which will be called SLSTR_root. Then all steps as below can be run:
  - For each product folder S1-S6, band conversion from ToA radiance to reflectance is needed;
  - SLSTR product data need to be go through cloudy and clear patch discrimination;
  - Data which pass through the previous step are saved into a datacube;
  - CloudFCN or other AI models can be run over the datacube to assess ;

2. Python Package (all python filenames are in bold font)

- SLSTR validation tool consists of 5 python files:
  - First, user runs **gui.py** which displays a graphical user interface to browse each single SLSTR product folder which needs to pass through the radiance to reflectance conversion
  - When user browses the SLSTR product folder, it is immediately run the conversion function contained in **rad_to_reflectance.py** that makes access to ancillary data to accomplish all necessary conversion steps;
  - The steps mentioned above (first two steps) are repeated for all SLSTR products of interest that are saved in the SLSTR_root folder;
  - The approach chosen to validate SLSTR product is planned to extract patches that are fully cloudy and clear patches to run Deep Learning over;
  - So, **image_loading.py** allows for the data extraction of cloud and bands into two different data cube using numpy python library.

- **Patch_extraction.py** is eventually used to extract patches whose size is compliant with the machine (or deep) learning model that is to be run over. At the end of the running of patch_extraction.py two new data cubes are given.

3. Miscellaneous
   - The function unpack_and_show.py is used to unpack all cloud flagging data embedded with the products of SLSTR as downloaded from the Copernicus hub. More in detail, after reading the flagging data as netCDF4 dataset the cloud flags can be unpacked:

```
data = netCDF4.Dataset('flags_an.nc','r','NETCDF4').variables['cloud_an'][:]
data = data.data
flags=np.unpackbits(data[...,np.newaxis].view("uint8")[...,::-1][...,np.newaxis],axis=-
1).reshape(data.shape+(-1,))[...,::-1] #(note that the current line and last one are to be meant
to be the same code line…)
```

   - If a user wanted to show a cloud mask he needs to choose one among the following possible flags offered by SLSTR (the user will be asked to choose one of the flags between squared brackets as below):

[0]visible
[1]1.37_threshold
[2]1.6_small_histogram
[3]1.6_large_histogram
[4]2.25_small_histogram
[5]2.25_large_histogram
[6]11_spatial_coherence
[7]gross_cloud
[8]thin_cirrus medium_high
[9]fog_low_stratus
[10]11_12_view_difference
[11]3.7_11_view_difference
[12]thermal_histogram
[13]spare
[14]spare

4. Dependencies
   Here is a list of all the dependencies of this package. If a user plans to apply some deep learning or machine learning code it will be necessary to import the packages and libraries placed to run models.

```
import os
import numpy as np
import glob
import sys
from PyQt5.QtWidgets import QApplication, QWidget, QPushButton, QLabel,
QToolTip, QMessageBox
from PyQt5.QtGui import *
from PyQt5.QtCore import pyqtSlot, QDir, QCoreApplication
from PyQt5.QtWidgets import QFileDialog, QDialog, QListView,
QAbstractItemView, QTreeView
from PIL import ImageTk, Image
from netCDF4 import Dataset
import glob
import cv2
from skimage import color
```