Downloading_S2MSIL2SR

March 28, 2024

1 Downloading Sentinel 2 MSI level 2 surface reflectance images using the Inovation Lab cloud computing environment

This Jupyter notebook demonstrates how to download S2 MSI level 2 surface refelectance with the ESA EO Africa inovation lab cloud computing environment.

Prerequisites for running this notebook

Several packages need to be installed and/or imported for running this script:

The `xml, gdal and os` modules should be imported first for downloading Sentinel 2 MSI level 2 surface reflectance iamges ;

1.1 Importing the relevant modules

[ ]:
```
import xml.etree.ElementTree as xml
from osgeo import gdal
import os
```

The full path to the unzipped Sentinel-2 folder: the .SAFE folder is the one required here.

[ ]:
```
homedir = '/home/eoafrica'
sentinelfile = '/eodata/Sentinel-2/MSI/L2A/2021/12/29/
 ↪S2B_MSIL2A_20211229T074229_N0301_R092_T37PEK_20211229T101323.SAFE'
#S2B_MSIL2A_20220217T073949_N0400_R092_T37PEK_20220217T102748.SAFE
sentinelfile = homedir + sentinelfile
```

Get the list of band-filenames (10m, 20m, 60m, and also the additional bands like TCI and SCL) from the Sentinel-2's xml file (MTD_MSIL2A.xml)

[ ]:
```
msil = open(sentinelfile + '/MTD_MSIL2A.xml')
msil = xml.parse(msil).getroot()
namespace_msil = msil.tag.split('}')[0] + '}'
filenames = msil.findall(namespace_msil + 'General_Info/Product_Info/
 ↪Product_Organisation/Granule_List/Granule/IMAGE_FILE')
print('There are ' + str(len(filenames)) + ' items in the sentinel image')
```

Loop through the band-filenames and store them one by one as a GeoTIFF file in the same folder as this python notebook.

[ ]:
```
sequence = 1
for name in filenames:
    name = name.text + '.jp2'
    print('['+str(sequence)+'/'+str(len(filenames))+'] '+ name)
    outputname = name[0:-4] + '.tif'
    outputname = outputname.split('/')[-1]
    if os.path.exists(outputname):
        print('File ' + outputname + ' already exists; skipping')
        sequence = sequence + 1
        continue
    try:
        #dataset=gdal.Open(sentinelfile + '/' + name) # this does not work␣
 ↪currently; the following four lines are the workaround
        f = open(sentinelfile + '/' + name, 'rb')
        mmap_name = "/vsimem/"+name
        gdal.FileFromMemBuffer(mmap_name, f.read())
        dataset=gdal.Open(mmap_name)
        dataset = gdal.Translate(outputname, dataset, format='GTiff')
        if dataset is None:
            print('Store failed')
        else:
            print('Stored as: ' + outputname)
        gdal.Unlink(mmap_name)
    except:
        print('no permission to read image')
    sequence = sequence + 1
```

[ ]:
```
import shutil
```

[ ]:
```
destination_dir = '/home/eoafrica/RF model/s2metadata'
shutil.copytree(sentinelfile, destination_dir)
print('Copied')
```

[ ]:
```
from osgeo import gdal

# Replace 'path_to_your_image.tif' with the actual path to your TIFF image
#Sentinel2_AWbasin/T37PEK_20211229T074229_B04_10m.tif
image_path = '/home/eoafrica/RF model/T37PEK_20211229T074229_B04_10m.tif'

# Open the TIFF file
dataset = gdal.Open(image_path)

# Check if the dataset was successfully opened
if dataset is not None:
    print("Dataset opened successfully.")

    # Get image dimensions
```

```
    width = dataset.RasterXSize
    height = dataset.RasterYSize
    bands = dataset.RasterCount
    print(f"Dimensions: {width} x {height} pixels, {bands} bands.")

    # Read the image data as a numpy array
    image = dataset.ReadAsArray()

    # Now, 'image' contains the pixel values of the TIFF image

    # Close the dataset
    dataset = None
else:
    print("Failed to open the dataset.")
```

[ ]: