SoftwareX 15 (2021) 100739

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

Automatic dataset builder for Machine Learning applications to satellite imagery

Alessandro Sebastianelli *, Maria Pia Del Rosso, Silvia Liberata Ullo

University of Sannio, Benevento, Italy

ARTICLE INFO

Article history:
Received 8 October 2020
Received in revised form 22 March 2021
Accepted 8 June 2021

Keywords:
Dataset creation
Big Data
Machine Learning
Python task automation
Google Earth Engine
Sentinel-1
Sentinel-2
Git-Hub

ABSTRACT

In this paper, the architecture of an innovative tool, enabling researchers to create in an automatic way suitable datasets for Artificial Intelligence (AI) applications in the Earth Observation (EO) context, is presented. Two versions of the architecture have been implemented and made available on Git-Hub, with a specific Graphical User Interface (GUI) suitable for non-expert users. The tool has been designed to work with different types of sensors, but up to now has been tested with Sentinel-2 and Sentinel-1 data. We strongly believe that this tool will be of great usefulness for researchers applying AI to EO and Remote Sensing (RS). At the best of our knowledge, there is not a similar freely available tool, collecting the same benefits.

Code metadata

Current code version	v55
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-20-00065
Code Ocean compute capsule	none
Legal Code License	MIT
Code versioning system used	git
Software code languages, tools, and services used	python
Compilation requirements, operating environments & dependencies	python 3.6.8, numpy, rasterio, geetools, earthengine-api, global-land-mask, matplotlib, descartes, pandas, geopandas, tk, scipy, imageio, pillow, scikit-image
If available Link to developer documentation/manual	https://github.com/Sebbyraft/SentinelDataDownloaderTool/blob/master/README.md
Support email for questions	sebastianelli@unisannio.it

Software metadata

Current software version	v2
Permanent link to executables of this version	https://github.com/Sebbyraft/SentinelDataDownloaderTool/blob/master/src_gui/main.py
Legal Software License	MIT
Computing platforms/Operating Systems	Linux, OS X, Microsoft Windows
Installation requirements & dependencies	python 3.6.8, numpy, rasterio, geetools, earthengine-api, global-land-mask, matplotlib, descartes, pandas, geopandas, tk, scipy, imageio, pillow, scikit-image
If available, link to user manual–if formally published include a reference to the publication in the reference list	For example: https://github.com/Sebbyraft/SentinelDataDownloaderTool/blob/master/README.md
Support email for questions	sebastianelli@unisannio.it

1. Motivation and significance

In recent years, in the field of Remote Sensing, a large number of applications have benefit in their data processing workflow by

* Corresponding author.
E-mail addresses: sebastianelli@unisannio.it (Alessandro Sebastianelli), mariapia.delrosso@unisannio.it (Maria Pia Del Rosso), ullo@unisannio.it (Silvia Liberata Ullo).

https://doi.org/10.1016/j.softx.2021.100739

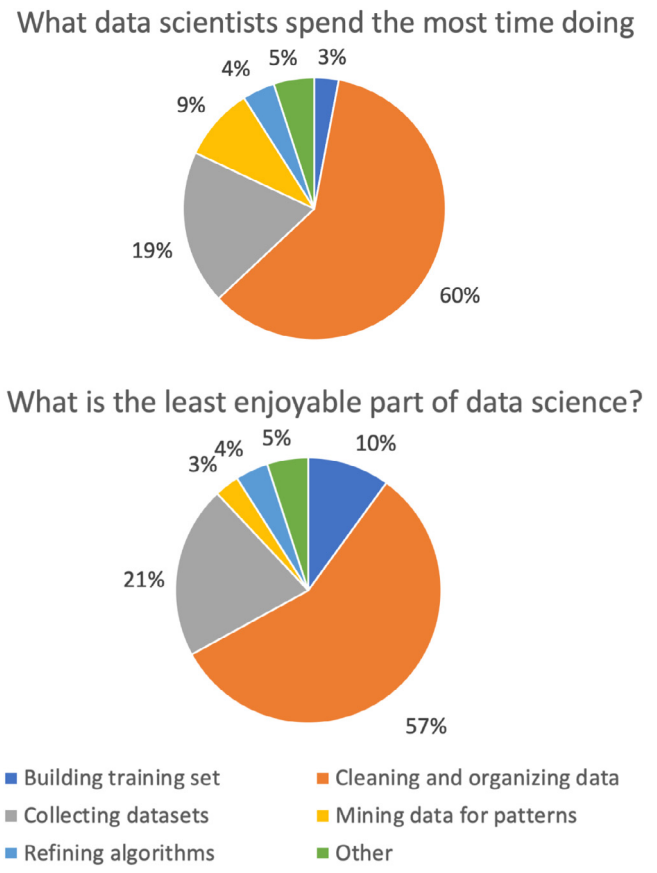

Fig. 1. Top: Time consuming activities of data science, Bottom: Least enjoyable activities of data science [6].

the introduction of Artificial Intelligence (AI), and in particular Machine Learning (ML) techniques [1]. However, when working with ML, the availability of a sufficiently large and statistically representative dataset becomes crucial [2–5]. A huge amount of data is necessary for training the neural network, chosen for a specific application, and this issue brings out all the problems related to Big Data, and their handling. Collecting, building, organizing and cleaning the data is a heavy time-consuming operation, which results in researchers' frustration, since it is felt as the least enjoyable part of data processing [6], as shown in Fig. 1, however a necessary step before focusing on the most interesting part of their own work.

In this paper, Authors aim to present the architecture of an automatic dataset builder, with a detailed description of its several blocks, in order to make available an innovative tool enabling researchers to create suitable datasets for Artificial Intelligence (AI) applications in the Earth Observation (EO) context. The annoying and repetitive operations are done in this way by the software in an automatic way, and the researcher can save the time for other more fruitful activities.

From the analysis of the state of the art, the availability of such architectures is very limited. A couple of them are present in the literature [7,8], but the proposed new model has many additional advantages: (1) the fully automatic chain processing, (2) the possibility to download and organize time series of data from multiple sources, (3) the presence of a Graphical User Interface (GUI) for non-expert users, (4) the possibility, especially for expert users, to add their pre-processing techniques to the processing chain, (5) the availability on Git-Hub (open access).

2. Software description

A general description of the tool is given in Section 2.1. Following, details on the current implementation of each function blocks are provided in Section 2.2.

2.1. Software architecture

As specified before, the proposed architecture describes the tool designed to build suitable datasets for ML applications in a simple and automatic way. A set of Python scripts allows the user to automatically download data from the Google Earth Engine catalog [9,10]. The functional diagram for the Satellite Dataset Creation Tool is shown in Fig. 2, where the several functional blocks are included, each of them dealing with a particular task.

The parameters to be set are the ones labeled in red in Fig. 2, and they are: the coordinates of the area of interest, its size, the dates, the bands, the number of images. In the next subsections, the various functional blocks will be explained. It is worth to highlight that during the entire process both the raw data and the processed data are saved in separated folder that can be accessed by the user when necessary.

2.2. Software functionalities

2.2.1. Generator

The generator produces a variable number of points, with longitude and latitude, distributed over the Earth surface. In the case of both Sentinel-1 and Sentinel-2 (the two satellites for which some examples of dataset creation will be provided in the next sections), a water-masking function has been introduced in the generation process, in order to focus on land applications [11]:

Water-masking. A mask, which allows identifying and delimiting water rich areas of the Earth with a certain resolution, is used; two classes are identified, water (value 1) and not water or land (value 0).

To generate the points, two random variables with uniform distribution have been used, one for the latitude values and the other for the longitude values. The latitudes are restricted to the range $[-56, 84]$, since Sentinel-2 does not acquire data beyond those values, while for the longitude the range is restricted to $[-180, 180]$ [12,13]. At each iteration a point is generated. The water mask is used to check if the generated point falls on the Earth's surface. If so, it is accepted and saved in a dedicated CSV file, otherwise it is discarded. The user can define the number of points to be generated and the size of the scene. Clearly, the size of the scene has a limit, and the greater the size the slower the overall downloading process.

2.2.2. Downloader

The downloader takes care of downloading the images using the coordinates previously generated, or defined by the user, and the time interval specified by the user. By default the script will try to download a one-year time series, with a monthly interval. For each month three Sentinel-1 images and three Sentinel-2 images are downloaded, this number was chosen to guarantee at least one image for each satellite that is in optimal conditions of light, cloud coverage, etc. The software is also designed to organize data in a hierarchical folder structure, for example for a region the structure is as follows (with the Sentinel-1 and Sentinel-2 data folders both contained in a master folder):

- Sentinel-1 (or Sentinel-2) folder:
  - Scene 1 folder:
    - January folder

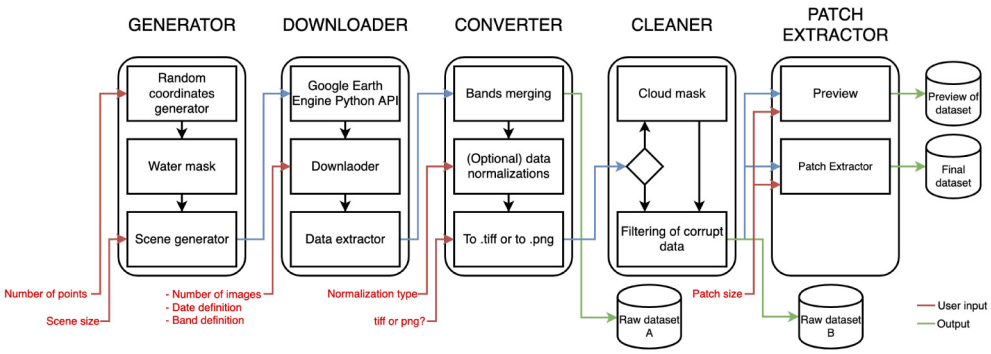

Fig. 2. Functional Diagram for the Satellite Dataset Creation Tool.

  - image 1
  - image 2
  - image 3
- February folder
- …

– Scene 2 folder
– …

The user can define the number of images, the date and the bands for each satellite. The default Sentinel-2 bands are B4, B3, B2 and QA60 (R, G, B and cloud mask) and for Sentinel-1 the default value is VV. The downloader deals exclusively with downloading the raw data, and this is the reason why the converter block becomes necessary after the downloader.

2.2.3. Converter

The converter mainly deals with taking raw data and applying some preprocessing techniques to produce as output easily treatable data, see Fig. 3.

For Sentinel-1 products the converter first standardizes or normalizes the data to bring them into a range suitable for Machine Learning purposes, then it saves the gray-scale data in png format and with data type uint8 (range 0, 255). For Sentinel-2 products the converter normalize the data, then through the RGB bands it builds a color image and then saves the data in png format with data type uint8 (range 0, 255). An example of converted data is shown in Fig. 3.

Up to now, three of the most common types of normalization techniques are implemented, the min–max, the standardization and the max technique, expressed by Eqs. (1), (2) and (3), respectively, where "$Image_{in}$" denotes the image to be normalized and "$Image_{out}$" the normalized image. The functions *min*, *max*, *std* and *mean* are used to calculate respectively the minimum, the maximum, the standard deviation and the mean of the input image. These are scalar values, built on all the image values. By using the aforementioned equations and the scalars, the matrix related to the input image is modified and a new matrix is calculated (the output image) [14,15].

$$Image_{out} = \frac{Image_{in} - min(Image_{in})}{max(Image_{in}) - min(Image_{in})} \tag{1}$$

$$Image_{out} = \frac{Image_{in} - mean(Image_{in})}{std(Image_{in})} \tag{2}$$

$$Image_{out} = \frac{Image_{in}}{max(Image_{in})} \tag{3}$$

The user can select the previously described conversion mode or set the converter so that it only creates the RGB image and saves both the Sentinel-1 and the Sentinel-2 acquisitions in ".tiff" format, by avoiding the data normalization in this case.

Normalization in fact is typically used to plot or, in the case of AI applications, to increase effectiveness during learning [16–19], but it is a process that modifies the image value range, and it can be irreversible. An expert user, by selecting the ".tiff" format, can decide to bypass this step, or even other preprocessing phases, in order to have raw data to which then applying his customized preprocessing techniques.

2.2.4. Cleaner

Unfortunately, it may happen that some images are corrupted or present a too high cloud coverage (in the case of Sentinel-2), therefore the cleaner block has been developed to overcome these problems.

For each satellite, it mainly deals with selecting for each region and for each date, the best image available among the three downloaded every month.

By using the 3 images (default value), the cleaner selects the best image for each month. In fact there can be some "errors" in the downloaded data. For example some Sentinel-1 downloaded data present some black or gray missing parts. Some Sentinel-2 downloaded data present the same problem but in addiction there can be images with a huge cloud coverage. See the images shown in Fig. 4 as an example.

For the missing parts or the cloud detection the software uses a threshold. It is worth to say that the cleaner is designed, for now, only to remove corrupted or cloudy data, and this is done in an automatic way. Yet, users who want to remove data with other type of characteristics (for example images acquired over dry areas), can use the cleaner in the manual mode. The manual cleaner allows the user to execute the same functionalities of the automatic cleaner without pre-defined settings. This extra option is available only when the tool is used in the semi-automatic GUI mode.

At the end of this operation the dataset should be composed, if the default settings are chosen, of 12 images, one for each month, for each satellite, for a total of 24 images for each region, free of corrupt or damaged portions, except for some unfavorable cases By default the size of each image is 1000 × 1000 pixels, so from each image it is possible to obtain numerous smaller patches.

2.2.5. Patch extractor

The patch extractor is an add-on that extracts smaller images from the final one to increase the samples in the dataset.

During this step, the smaller images are created with the preview of the dataset. Each image contains time series with Sentinel-1 and Sentinel-2 data from a specific geographic region. The images are organized in a matrix form, on the columns there are the different patches extracted, and on the rows there are the different acquisitions over time.

Each Sentinel-1 and Sentinel-2 image after cleaning can undergo the Patch Extractor step, as shown in Fig. 4.

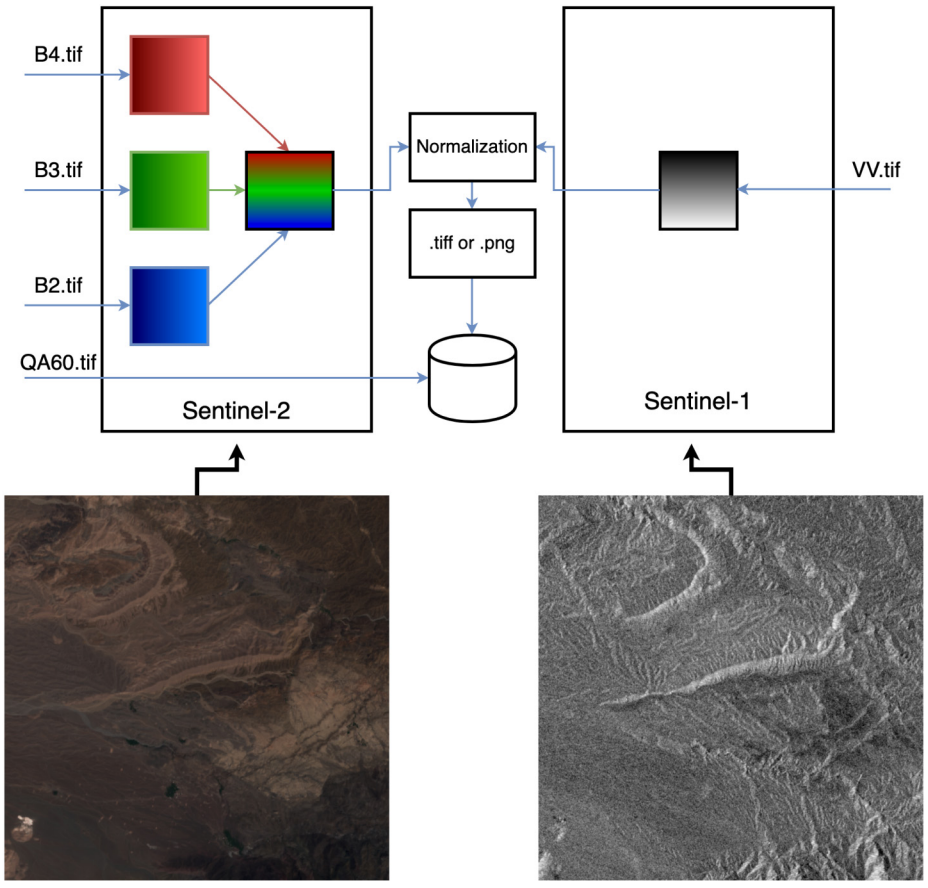

Fig. 3. Converter functional scheme.

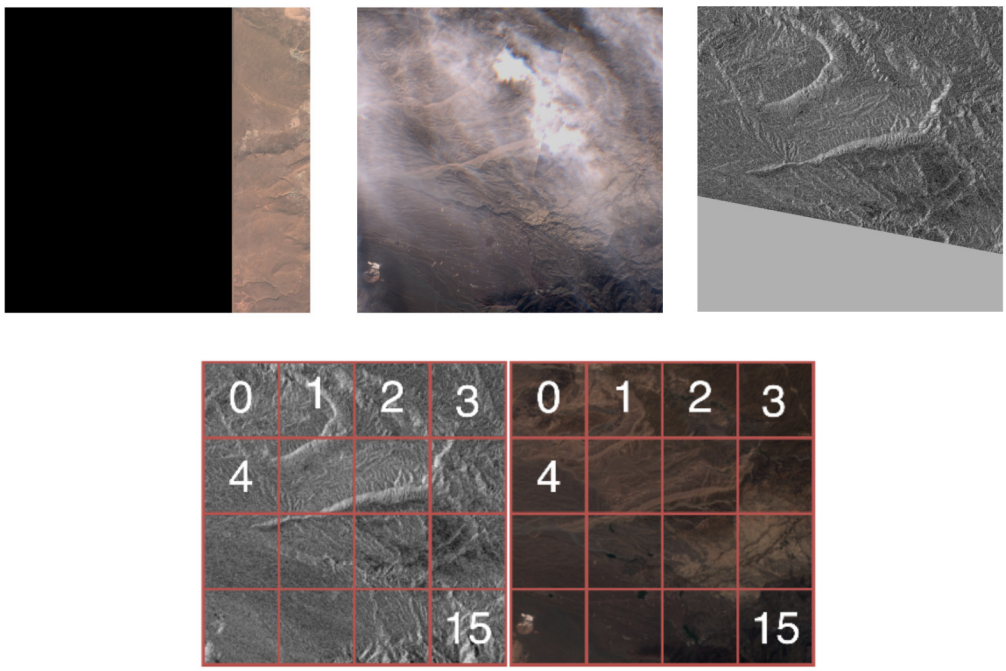

Fig. 4. Top: Some "errors" in the downloaded data. Sentinel-2 wrong on the top left and Sentinel-2 cloudy on the top center. Sentinel-1 wrong on the top right. Bottom: Example of extracted patches from the image after the cleaning step.

The preview can be used to verify all the steps applied by the tool and it can be used also to manually select some data of interest, if necessary. Indeed, the file name contains the information about the position of the images in the dataset. Then using the number of rows and columns, the user can extract a particular portion of the image.

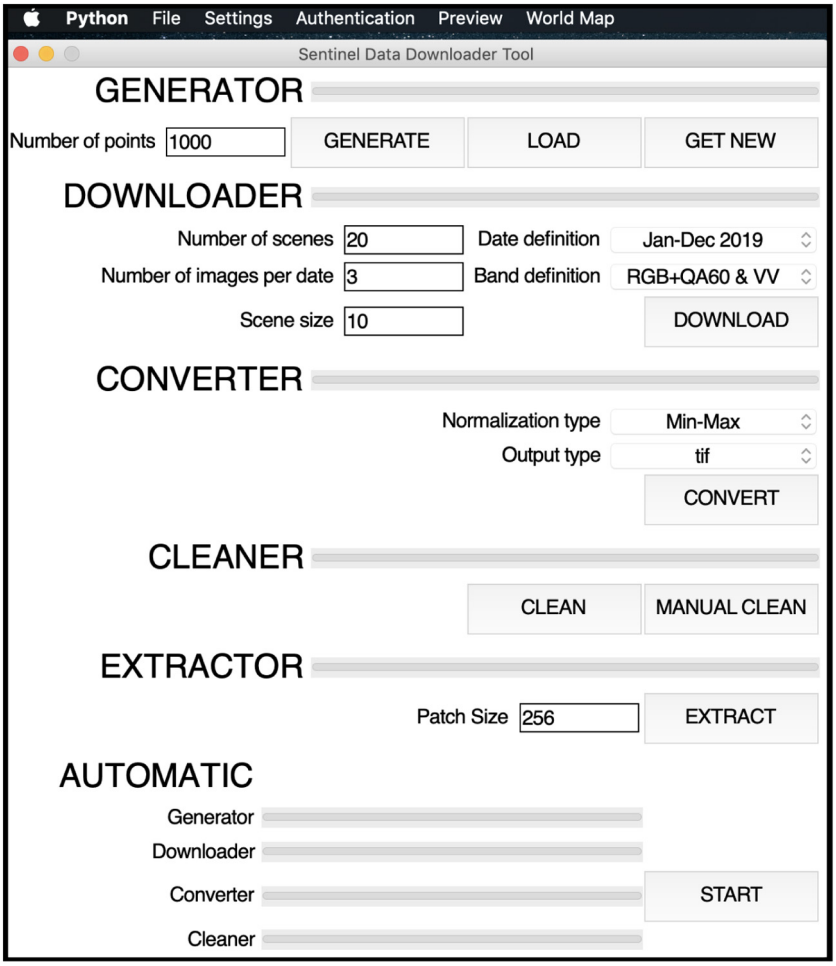

Fig. 5. GUI of the Satellite Data Downloader Tool.

2.2.6. The Graphical User Interface

The Graphical User Interface (GUI) can be managed by both expert and non-expert users, in fact two modalities are made available. Expert users can, for instance, utilize the tool by running the scripts in environment like Jupyter Notebook. Non-expert users, instead, can run the GUI (Graphical User Interface) directly by using the interface shown in Fig. 5. Obviously the GUI offers less flexibility with respect to the functioning mode for experts, but it is more intuitive and of easy use.

As it can be seen from Fig. 5, the GUI presents two main sections, on the bottom the Automatic section with the START button, and on the remaining part, the possibility of a semi-automatic functioning:

- Automatic: the user can run a fully automatic process, using default settings (listed in a setting file), by pressing the START button
- Generator, Downloader, Converter, Cleaner, Extractor: the user can change some settings and can run the different processes separately

On the top of Fig. 5, there are also other options accessible to the user. The most interesting are the Preview and the World Map. Indeed these are two extra components that allow the user to easily navigate thorough the dataset and to plot over a world map the generated points.

The tool and a more detailed user guide can be found on the related Git-Hub page [20].

3. Illustrative examples

In this section, the implementation of datasets with the proposed tool is described, in order to show its effectiveness. In a first experiment, the tool under random selection mode is used. The goal is creating a final dataset composed by 8.000 Sentinel-1 and 8.000 Sentinel-2 images. In input to the tool a number equals to 2.000 is settled, so that the tool randomly selects 2,000 different geographical areas at different coordinates. For each geographical area, 4 Sentinel-1 and 4 Sentinel-2 images are automatically acquired with a time interval of one month. Afterwards, smaller images are selected by creating different patches, as underlined in the above subsection E. Patch Extractor, under a shape of 256×256 pixels, resulting in a matrix with $256 \times 256 \times 1$ dimensions for Sentinel-1 images, where only the VV polarization is chosen; and a matrix with $256 \times 256 \times 3$ dimensions, with an extra matrix for the cloud mask (QA60 band), for Sentinel-2 images, where the three R, G, B bands are selected. The images are organized in a hierarchical structure. with two master folders, for Sentinel-1 and Sentinel-2 data respectively, called sen1 and sen2, each containing a sub-folder for the area of interest, with inside each time-series of 4 images, as sketched in Table 1.

Note that the image path contains also a serial number (RGB_n), that goes from 1 to the dataset size, 8.000 in this example. A sample of the dataset is shown in Fig. 6, where the first time-series is shown.

Moreover, the proposed tool has been also employed in the creation of a Sentinel-2 dataset used for the detection of landslides with a deep learning approach, as discussed in [21], which

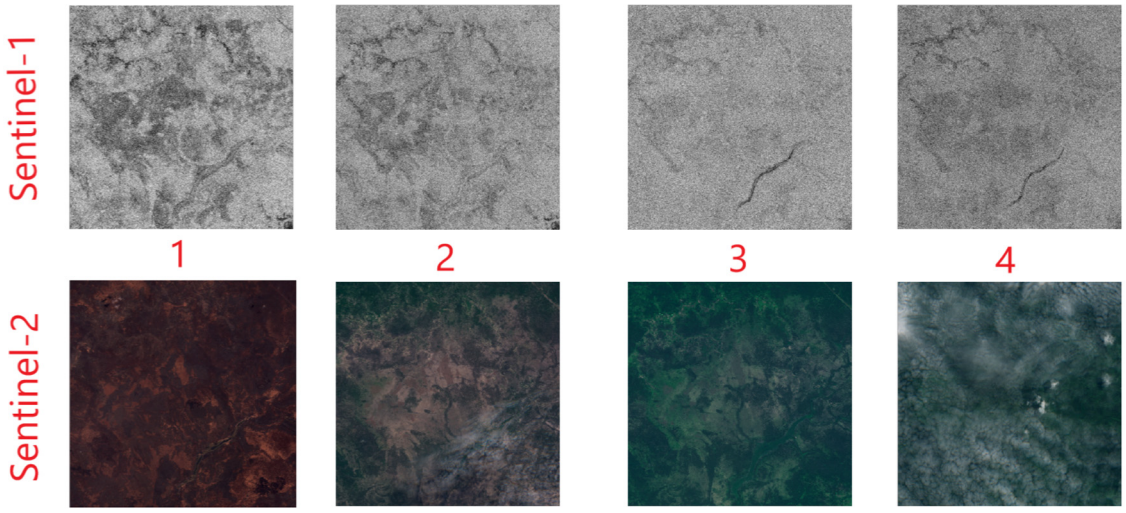

Fig. 6. Example of the implemented dataset — First time-series created for Sentinel-1 (on the top) and Sentinel-2 (at the bottom).

Table 1
Hierarchical structure of the dataset.

Satellite	Time series	Images path
Sentinel-1	1	sen1/zone_1/image_VV_0001_Jan_patch_0.tif sen1/zone_1/image_VV_0002_Feb_patch_0.tif sen1/zone_1/image_VV_0003_Mar_patch_0.tif sen1/zone_1/image_VV_0004_Apr_patch_0.tif
…	…	…
Sentinel-1	2000	sen1/zone_2000/image_VV_7997_Jan_patch_0.tif sen1/zone_2000/image_VV_7998_Feb_patch_0.tif sen1/zone_2000/image_VV_7999_Mar_patch_0.tif sen1/zone_2000/image_VV_8000_Apr_patch_0.tif
Sentinel-2	1	sen2/zone_1/image_VV_0001_Jan_patch_0.tif sen2/zone_1/image_VV_0002_Feb_patch_0.tif sen2/zone_1/image_VV_0003_Mar_patch_0.tif sen2/zone_1/image_VV_0004_Apr_patch_0.tif
…	…	…
Sentinel-2	2000	sen2/zone_2000/image_VV_7997_Jan_patch_0.tif sen2/zone_2000/image_VV_7998_Feb_patch_0.tif sen2/zone_2000/image_VV_7999_Mar_patch_0.tif sen2/zone_2000/image_VV_8000_Apr_patch_0.tif

interested readers can refer to for further information and to see a practical application of the same. In that case the tool has not been used under random selection mode, but the user selected the areas of interest by using a csv files with the coordinates of the AOI.

The automatic dataset builder has also a particular feature, which can be very useful for the ML applications. The implemented dataset can be split into two different subsets of data: for training and validation of future networks, by calculating the dissimilarity index between the cumulative histograms of the two sub-dataset, as given by Eq. (4):

$$dissimilarity = \frac{\frac{1}{N}\sum_{i=0}^{N}\left|\frac{h_{training}(i)}{\#bins} - \frac{h_{validation}(i)}{\#bins}\right|}{\frac{1}{N}\sum_{i=0}^{N}\frac{h_{training}(i)}{\#bins}} \quad (4)$$

In the equation, $h_{training}$ and $h_{validation}$ are respectively the training and validation cumulative histograms, calculated with a specific number of bins, N is the number of images used to compute the histograms. By repeating this test several times, changing each time the samples in training and validation set, the authors selected the best couple of training and test dataset that minimizes the dissimilarity.

4. Impact

The idea behind this tool is the complete automation of the dataset creation for applications in RS to facilitate the work of the researchers, especially when ML algorithms are used. In fact, the necessary interactions required by the user are very a few. At the best of our knowledge, there is not freely available a similar tool, collecting the same benefits. With respect to what we found in the literature [8,15,22], the proposed architecture presents itself as a tool as general purpose as possible. It supports a fully automatic chain processing, and the possibility to download and organize time series of data from different sources; it is designed for experts and non-experts users, by making available a GUI, suitable for non-expert users, and allowing the expert users to add their pre-processing techniques to the processing chain; and last, it is available on an open access platform as Git-Hub. The software contains also other functionalities that can help the user for example to navigate into the dataset, easily filter out data by visualizing it, calculate dataset statistics, split the dataset into sub-datasets and so on. The tool is structured to be highly integrable with other processing blocks, indeed each component can be used separately and can be followed by blocks defined by the user.

Two different scenarios have been used to test the tool: one with random acquisitions on the globe, and the other using specific coordinates, by proving its effectiveness.

5. Conclusions

In this paper the architecture of an innovative tool, enabling researchers to create in an automatic way suitable datasets for AI applications in the EO context, has been presented. Two versions of the architecture have been implemented and made available on Git-Hub, with a specific GUI for non-expert users. For now, the tool supports only data available from the Google Earth Engine catalog and it has been fully tested on Sentinel-1 and Sentinel-2 data. Future testing will include other data available on Google Earth Engine catalog and the integration of new sources of data. The proposed tool will be of great usefulness for researchers applying ML to EO and RS.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The work has been carried out by the University of Sannio researchers while hosted at the Φ-Lab of the European Space Research Institute (ESRIN) in Frascati [23]. A special acknowledgment goes to Pierre-Philippe Mathieu, Chief of ESA ESRIN Phi-Lab, for joint brainstorming and sharing of ideas.

References

[1] Del Rosso MP, Sebastianelli A, Ullo SL. Artificial Intelligence Applied to Satellite-based Remote Sensing Data for Earth Observation. Published by The Institution of Engineering and Technology (IET); 2021.
[2] Camps-Valls G. Machine learning in remote sensing data processing. In: 2009 IEEE International Workshop on Machine Learning for Signal Processing. 2009, p. 1–6.
[3] Roh Y, Heo G, Whang SE. A survey on data collection for machine learning: A big data - AI integration perspective. IEEE Trans Knowl Data Eng 2019. 1–1.
[4] Brownlee J. How much training data is required for machine learning?. 2019, https://machinelearningmastery.com/much-training-data-required-machine-learning/.
[5] Machine learning mastery - jason brownlee, impact of dataset size on deep learning model skill and performance estimates. 2019, https://machinelearningmastery.com/impact-of-dataset-size-on-deep-learning-model-skill-and-performance-estimates/.
[6] Gil Press. Cleaning big data: Most time-consuming, least enjoyable data science task, survey says. 2016, https://www.forbes.com/sites/gilpress/2016/03/23/data-preparation-most-time-consuming-least-enjoyable-data-science-task-survey-says/#11e80e496f63.
[7] Schmitt M, Hughes L, Zhu X. The sen1-2 dataset for deep learning in sar-optical data fusion. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences 2018;4:141–6.
[8] Ranghetti L, Boschetti M, Nutini F, Busetto L. "sen2r": An R toolbox for automatically downloading and preprocessing Sentinel-2 satellite data. Comput Geosci 2020;104473.
[9] Google. Google earth engine home page. 2020, https://earthengine.google.com.
[10] Gorelick N, Hancher M, Dixon M, Ilyushchenko S, Thau D, Moore R. Google earth engine: planetary-scale geospatial analysis for everyone. Remote sensing of Environment 2017;202:18–27.
[11] Todd Karin, Global Land Mask, https://github.com/toddkarin/global-land-mask, python package - GitHub.
[12] European Space Agency (ESA). Sentinel-2 - revisit and coverage. 2020, https://sentinel.esa.int/web/sentinel/user-guides/sentinel-2-msi/revisit-coverage.
[13] European Space Agency. Sentinel-1 - observation scenario. 2020, https://sentinel.esa.int/web/sentinel/missions/sentinel-1/observation-scenario.
[14] Patro S, Sahu KK. Normalization: A preprocessing stage. 2015, arXiv preprint arXiv:1503.06462.
[15] Al Shalabi L, Shaaban Z. Normalization as a preprocessing engine for data mining and the approach of preference matrix. In: International Conference on Dependability of Computer Systems. 2006, p. 207–14.
[16] Brownlee J. How to use data scaling improve deep learning model stability and performance. 2019, https://machinelearningmastery.com/how-to-improve-neural-network-stability-and-modeling-performance-with-data-scaling/.
[17] Loukas S. Everything you need to know about min-max normalization. 2020, https://towardsdatascience.com/everything-you-need-to-know-about-min-max-normalization-in-python-b79592732b79.
[18] Raschka S. About feature scaling and normalization. 2014, https://sebastianraschka.com/Articles/2014_about_feature_scaling.html.
[19] Du Y, Teillet PM, Cihlar J. Radiometric normalization of multitemporal high-resolution satellite images with quality control for land cover change detection. Remote Sens Environ 2002;82(1):123–34. http://dx.doi.org/10.1016/S0034-4257(02)00029-9.
[20] Sebastianelli, A. and Del Rosso, M. P. and Ullo, S. L., The Sentinel Data Downlaoder Tool Git-Hub repository, https://github.com/Sebbyraft/SentinelDataDownloaderTool.
[21] Ullo SL, Langenkamp MS, Oikarinen TP, Del Rosso MP, Sebastianelli A, Sica S, et al. Landslide geohazard assessment with convolutional neural networks using sentinel-2 imagery data. In: IGARSS 2019-2019 IEEE International Geoscience and Remote Sensing Symposium. 2019, p. 9646–9.
[22] Long Y, Xia G-S, Li S, Yang W, Yang MY, Zhu XX, Zhang L, Li D. DiRS: On creating benchmark datasets for remote sensing image interpretation. 2020, arXiv:2006.12485.
[23] European Space Agency. Φ-lab home page. 2020, https://philab.phi.esa.int/.