SEMANTIC SEGMENTATION USING AUNET ARCHITECTURE ON SENTINEL-2 DATA

I. Kotaridis 1, *, M. Lazaridou 1

1 Aristotle University of Thessaloniki, Faculty of Engineering, School of Civil Engineering, Lab. of Photogrammetry - Remote Sensing, 54124 Thessaloniki, Greece - (iskotarid, lazamari)@civil.auth.gr

KEY WORDS: CNNs, UNET, superpixel segmentation, Python, Sentinel-2.

ABSTRACT:

This paper presents the development of a methodological framework, based on deep learning, for the efficient mapping of main land cover classes (built-up, vegetation, barren land, water body) on different urban and suburban landscapes. In particular, the proposed framework integrates the superpixel segmentation (an essential procedure) with deep learning. A combination of spectral bands and indices is introduced to produce optimal results, ensuring adequate discrimination between built-up and barren land classes. A UNET architecture is implemented, which can learn the characteristics of main land cover classes from the input data that can be deployed from a Colab notebook without excessive computational needs. The resulted classifications depict promising accuracy values (above 90%).

1. INTRODUCTION

Global population has increased rapidly over the last century, which triggered alterations in land cover of the natural landscape. In particular, the proposed framework can learn the characteristics of main land cover classes from a Colab notebook without excessive computational needs. The resulted classifications depict promising accuracy values (above 90%).

The major objectives of this paper are:

1. The combination of spectral bands and indices to produce optimal results, ensuring adequate discrimination between built-up and barren land classes.
2. The development of a methodological framework, based on deep learning, for the efficient mapping of main land cover classes (built-up, vegetation, barren land, water body) on different urban and suburban landscapes. In particular, the proposed framework integrates the superpixel segmentation (an essential procedure) with deep learning.
3. The implementation of a UNET architecture, which can learn the characteristics of main land cover classes from the input data that can be deployed from a Colab notebook without excessive computational needs.

* Corresponding author
4. The effective integration of the presented methodology in relevant thematic information extraction tasks.

2. MATERIALS AND METHODS

2.1 Study area and satellite data

In this paper, three Sentinel-2 level-2A (Bottom-Of-Atmosphere) corrected reflectance images were obtained. The first imagery concerns the train area (Thessaloniki city). The second and third images include the test areas (two Italian cities, Bari and Genoa). The criteria for the selection of scenes were the high quality of data and the limited cloud coverage. A subset was extracted from each scene for analysis in order to include an urban area of various density values. They comprise of several diverse land cover types including concrete, asphalt, water, vegetation and soil, as presented in natural color composites in Figure 1. The detailed features of these images are provided in Table 1.

<table>
<thead>
<tr>
<th>Product</th>
<th>Acquisition date</th>
<th>Spatial resolution (m)</th>
<th>Tile</th>
</tr>
</thead>
<tbody>
<tr>
<td>S2B_MSIL2A (Thessaloniki)</td>
<td>2021-08-01</td>
<td>10</td>
<td>T34TFL</td>
</tr>
<tr>
<td>S2B_MSIL2A (Bari)</td>
<td>2021-08-17</td>
<td>10</td>
<td>T33TXF</td>
</tr>
<tr>
<td>S2B_MSIL2A (Genoa)</td>
<td>2021-08-10</td>
<td>10</td>
<td>T33SVV</td>
</tr>
</tbody>
</table>

Table 1. Optical satellite data used.

2.2 Methodological framework

2.2.1 Tools: For the purpose of this study, Python code was developed and executed on Google Colab (Colaboratory). In specific, the following libraries were used:

- Numpy, a core library for scientific computing, for basic array operations.
- Pyrsgis to read and export GeoTIFFs.
- patchify library to split images into small overlapping patches by given patch and step size, and merge patches into original image during the prediction step.

Figure 1. Location map of the train area and test regions and the corresponding Sentinel-2 natural color composite subset images.
• Scikit-learn, machine learning package that offers functionality supporting supervised and unsupervised learning, for data pre-processing and accuracy checks.
• Keras with Tensorflow backend, a machine learning and artificial intelligence framework, for building and deploying the CNN model.
• matplotlib for creating visualizations (plots).

In addition, QGIS and Orfeo Toolbox (OTB) were used to preprocess the data. QGIS is a free and open-source Geographic Information System that supports the creation, editing, visualization, and publication of geospatial data. In specific, it was used for digital processing of the Sentinel-2 images. OTB, an open-source software library that supports processing of remote sensing data, was employed for input data normalization. Finally, the validation of the results and the visualization (maps) were also carried out in QGIS.

2.2.2 Training and testing dataset preparation: Dataset preparation produces ready-to-use samples for the UNET model. It can be distinguished into the following discrete phases: initial processing of Sentinel-2 imagery, superpixel segmentation, and data sampling (Figure 3).

Initial processing includes spectral indices calculation, normalization of spectral bands (R, G, B, NIR), stacking of the normalized bands and spectral indices to produce a single image product and clipping this product to the boundaries of the area of interest (AOI).

In this study, the combined use of four common spectral bands (R, G, B, NIR) and three suitable spectral indices (MNDWI, NDVIRc, NDTI) is proposed to extract the main land cover classes in a complex and heterogeneous environment, that is, a city and its surrounding areas. Distinguishing barren land from other environmental conditions is often a difficult task (Osgouei et al., 2019). We deduced from several experiments that the aforementioned input produces the optimal results. Table 2 summarizes the spectral indices and their corresponding equations. Figure 2 clearly illustrates the contribution of spectral indices in terms of spectral separability between the different land cover classes.

<table>
<thead>
<tr>
<th>Spectral Index</th>
<th>Equation</th>
<th>Sentinel-2 bands</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNDWI</td>
<td>Green – SWIR1</td>
<td>B3 – B11</td>
</tr>
<tr>
<td></td>
<td>Green + SWIR1</td>
<td>B3 + B11</td>
</tr>
<tr>
<td>NDVIRc</td>
<td>Red edge 1 – Red</td>
<td>B5 – B4</td>
</tr>
<tr>
<td></td>
<td>Red edge 1 + Red</td>
<td>B5 + B4</td>
</tr>
<tr>
<td>NDTI</td>
<td>SWIR 1 – SWIR 2</td>
<td>B11 – B12</td>
</tr>
<tr>
<td></td>
<td>SWIR 1 + SWIR 2</td>
<td>B11 + B12</td>
</tr>
</tbody>
</table>

Table 2. Spectral indices calculated in this study.

Another concern during the train/test dataset preparation was to normalize the input features into similar value ranges. Data normalization is critical to ensure that all the features are treated in an equal manner, considering that neural networks are sensitive to the distribution of data (Chollet, 2018). We implemented a data normalization (range from 0 to 1) procedure to obtain a land cover probability output. To achieve that, we used the maximum value from each individual spectral band (spectral indices are already normalized). For this purpose, the BandMathX application was accessed through the otb/Application Python module. The normalized spectral bands and indices were then stacked to from a single raster image. Finally, this raster image was clipped to the boundaries of the AOI.

Image segmentation was carried out in Terminus QGIS plugin. Terminus was developed in order to provide an easily accessible tool that allows users to perform image segmentation tasks. It is a fast and straightforward plugin that includes four popular image segmentation algorithms: Felzenszwalb’s, quickshift, SLIC and watershed. Each algorithm produces two outputs, a vector file and a raster file with the produced segments. The plugin offers the user the option to compute various statistics over each segment. If this is the case, these statistics are included in the fields of the output vector file and the bands of the multiband raster file that is created. This raster file contains the statistics of the pixels within each segment as the output bands. Thus, it can be displayed as a color composite of user’s choice.

For the purpose of this study, following several trial-and-error attempts, Felzenszwalb’s superpixel segmentation algorithm was employed. Felzenszwalb’s method is a graph-based image segmentation algorithm based on pairwise region comparison. It produces a segmentation of a multichannel image using a fast, minimum spanning tree-based clustering on the image grid. An important aspect of this algorithm is the ability to maintain detail in low-variability image areas whereas ignoring detail in high-variability areas (Felzenszwalb and Huttenlocher, 2004). In addition, the mean value statistic was included in the form of the bands of the output raster file. Segmentation produced a controlled oversegmentation that is preferable to under-segmentation, since splitting segments a posteriori is a more complicated task than merging them.
2.2.3 UNET architecture: A fully convolutional neural network (FCN) replaces the fully connected layers in CNN with up-convolutional layers and concatenates with a shallow, finer layer to produce end-to-end labels (Long et al. 2015). The standard CNN operates in an “image-label” way, while the “end-to-end” labelling mode in FCN is more suitable for pixel-based image classification, i.e., assigning each pixel the label of its corresponding class (Zhang et al. 2018). UNET is an architecture for semantic segmentation. It is an improved FCN model defined by its symmetrical U-shaped architecture consisting of symmetric contracting path (follows the typical architecture of a convolutional network) and expansive path. It combines low level features with detailed spatial information with high level features with semantic information to improve segmentation accuracy (Ronneberger et al. 2015; Zhang et al. 2018). A UNET model (like other CNNs) determines the relationship between characteristics (features) of an entity with a property (label). For this reason, several samples (features with their corresponding labels) are fed into the model and undergo a learning procedure to predict labels for new data (unlabelled data).

A CNN model determines the relationship between characteristics (features) of an entity with a property (label). For this reason, several samples (features with their corresponding labels) are fed into the model and undergo a learning procedure to predict labels for new data (unlabelled data). In this paper, the samples from the segmented stacked image will be referred to as input features (X) and classified land cover data (a reference land cover map was generated) as input labels (Y).

This paper proposes a standard UNET architecture that is based on image patches to perform semantic segmentation. In such an approach, during the training phase, patches are generated from the input features. Once the UNET model is trained with these patches, it is validated on the test patches. We used a 64x64 window with a stride of 64 (window slide) that resulted in 375 non-overlapping patches that were fed to the UNET model. In order to evaluate the performance of the applied model, 70% of them (262) were used as training samples, while the rest 30% (113) were used as test samples.

This contribution has been peer-reviewed.

https://doi.org/10.5194/isprs-archives-XLIII-B3-2022-119-2022 | © Author(s) 2022. CC BY 4.0 License.

3. EXPERIMENTS

3.1 Computational resources

All source codes of the procedures previously described were seamlessly implemented in Python using the Keras framework (https://keras.io) via TensorFlow (serves as a backend engine) on GPU. Keras is a deep-learning framework for Python that provides a convenient way to define and train almost any kind of deep-learning model (Chollet 2018). The experimental results were produced on Google Colab. It allows user to write and execute Python scripts in the browser together with explanatory text in a single document (notebook). An important feature is the capability to import data from Google Drive as well as from Github.

3.2 Model performance evaluation

The proposed methodological framework introduced in section 2 was employed in the area of Thessaloniki for training, while experiments were conducted in two other areas to investigate its potential in land cover classification for urban growth monitoring. During the validation phase the performance of the trained UNET is examined on the corresponding test dataset. The model was trained for 50 epochs. A few common error metrics regarding the validation of the model are presented in Table 3 and the produced learning curves are presented in Figures 5 & 6 for both training and validation phase.
4. DISCUSSION AND CONCLUSION

In this research, a promising land cover classification method based on deep learning is proposed for Sentinel-2 data. The method combines superpixel segmentation with deep learning on input data that include spectral indices to distinguish built-up environment from barren land and overcome traditional pixel-based limitations. The imagery is first segmented into superpixels using Felzenszwalb’s algorithm, and then a UNET architecture is employed, which can extract land cover features.

The proposed framework was validated in two coastal cities on the Mediterranean Sea, and performed quite well depicting high accuracy values for an improved classification of main land cover classes.

In Genoa, built-up land cover class covers an area of 21.2 km² (24 %), vegetation 30.1 km² (34 %), barren land 3.5 km² (4 %), and water body 34.2 km² (39 %) from a total of 88.9 km². In Bari, built-up land cover class covers an area of 47.6 km² (30 %), vegetation 10.6 km² (7 %), barren land 59.3 km² (38 %), and water body 40.5 km² (26 %) from a total of 158.1 km².

In the proposed land cover classification framework, superpixel segmentation was carried out to cluster the raster file from the initial processing of Sentinel-2 imagery into small homogeneous regions. We strongly support that this step is necessary, since it transforms the input data in a way that when the UNET model is applied, the final classification map does not suffer from the common issues of a pixel-based approach.

To our concern, a critical aspect is the feasibility of implementation of the proposed method. Since it has relatively minor computational needs, it can be deployed for similar purposes from Google Colab, executing the code on Google's cloud servers. The setup of the pre-requisite ML libraries is quite easy task, since most of them are already installed. In addition, a comprehensive understanding of machine learning fundamentals is essential for a smooth implementation of the aforementioned procedures.

The proposed methodology can be implemented in urban growth monitoring, ensuring an automated procedure on multitemporal imagery. This will be included in a future research work. In due time, the Google Colab notebooks will be available in the authors’ GitHub repository.

3.3 Results

Following the successful training of the proposed model, it was applied to the test areas to produce a land cover map for each of them (Figure 7 & 8). It has to be underlined that the testing material has to undergo the same preprocessing steps as the input training data (i.e., steps 1 and 2 in Figure 3).

---

4 https://github.com/ikotarid?tab=repositories
Table 4. The error matrix used to report accuracy results in Genoa with land cover types indicated as follows: 1 = built-up, 2 = vegetation, 3 = barren land, 4 = water body.

<table>
<thead>
<tr>
<th>Produced labels</th>
<th>Reference labels</th>
<th>Total</th>
<th>User's accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>25</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>2</td>
<td>18</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Total</td>
<td>31</td>
<td>26</td>
<td>18</td>
</tr>
<tr>
<td>Producer’s accuracy</td>
<td>81%</td>
<td>92%</td>
<td>100%</td>
</tr>
<tr>
<td>Overall accuracy</td>
<td>92%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
REFERENCES


