Utrecht University

Park-NET: Identifying Public Urban Green Spaces Using Multi-Source Spatial Data and Convolutional Networks

Marta Małgorzata Kozłowska

Student number: 5155231

Master's thesis

Supervisor: S.M. Labib

July 2022

'Applied Data Science' programme at Utrecht University

Abstract

Public urban green spaces (PUGSs) are a vital part of cities. They have a positive influence on the ecosystem and the well-being of local citizens. This is widely recognizable now with the United Nations (UN) mentioning public urban green spaces accessibility in their Sustainable Development Goals (SDG). To make the most of PUGSs, create them, and manage them sustainably we need to understand them and their spatial and temporal characteristics better. There is also a need for more accessible and cheaper PUGSs datasets so that officials and locals can use that in designing more sustainable neighbourhoods and cities. While spatial PUGSs data for some cities are available in land use dataset, these are often inaccessible to outside researchers, and not updated frequently. Hence there are methodological limitations on how PUGS can be created more efficiently for a wider usage. Deep learning methods with the usage of satellite images are a great way to achieve that because they can capture geometric patterns of land cover types using freely available satellite images of moderate spatial and temporal resolution. This study evaluates two convolutional network architectures, U-Net and U-Net with a ResNet-34 encoder, for semantic segmentation of PUGSs at a metropolitan level from multiple cities worldwide. Open-source data, mainly Sentinel and ground truth data about PUGSs from various open data portals are used as an input data. The chosen best model had an average test Intersection over Union (IoU) of 0,5610, and average test F1 score of 0,64515 across two external cities, which is a moderate to good performance of the deep learning model in detecting PUGSs. It shows that this approach is promising to create reliable, new PUGSs datasets that can help officials in urban planning.

Keywords: Convolutional Neural Networks, Public Urban Green Spaces, Image Processing, Satellite Imagery, Semantic Segmentation

All code can be found on GitHub repository

Table of contents

1. Introduction

Nowadays more than half of people globally live in cities, and this proportion is expected to increase to 68% by 2050 according to United Nations (UN). That is why making our cities healthier to live in, by significantly transforming the way we build and manage our public urban green spaces (PUGSs) is even more important than before. In 2015 United Nations Development Programme created Sustainable Development Goals (SDG) as a "universal call to action to end poverty, protect the planet, and ensure that by 2030 all people enjoy peace and prosperity". Goal 11 is about "Sustainable cities and communities". It highlights that everyone should have universal access to safe, inclusive, accessible, green, and public spaces. Therefore, it is important to study PUGS, so we get an understanding of how they are exactly influencing neighbourhoods, and how we should be transforming our cities to make the most of them.

PUGS can be studied in various ways using tools and techniques related to GIS, satellite images, and recently also machine learning. Spatial delineation of GIS elements (so e.g. PUGSs) has often been based on a re-classification of available land cover data combined with information about the natural values of each cover class, or by visual interpretation of aerial imagery, remotely sensed data interpretation, and manual digitalization (Skokanová et al., 2020). These processes are often costly and time-consuming, and they are often outdated due to irregular update frequency. Therefore, in the last few years, there have been various studies that try to use different machine learning algorithms for the detection of green spaces or landcover classification. Among them are supervised maximum likelihood classification used on Sentinel-2A satellite imagery, provided in the frame of the European Copernicus program (Kopecká et al., 2017), random forest, and support vector regression used on Landsat (Sharifi & Hosseingholizadeh, 2019), and Convolutional Neural Networks (CNN), precisely a U-Net architecture and very high resolution (VHR) imagery (Huerta et al., 2021). It is especially important because CNNs have recently shown the state of the art performance in high-level vision tasks, such as image classification and object detection (Chen et al., 2014). Some studies implemented CNN with a transfer learning approach. Transfer learning builds upon learned knowledge from one dataset to improve learning in another dataset (Wurm et al., 2019). It uses weights pre-trained on datasets from other domains to improve learning accuracy and rate for a model on a second task (Ulmas & Liiv, 2020).

When trying to study urban green spaces in a more global approach some studies used datasets that are available worldwide, for example, Open Street Map (OSM), and Urban Atlas (Ludwig et al., 2021). This information might be outdated quite fast, because of rapid city growth, and updating these datasets is costly. There might also be some inconsistencies in these datasets because the definition

of urban greenspaces might differ across countries. That's why there is a need of creating a uniform dataset containing urban green spaces.

The goal of this study is to analyse to what extent can a reproducible CNN model that identifies public urban green spaces based on open source data be created. The process involves creating two CNN models. One was the baseline U-Net model from scratch, and the other was the U-Net with ResNet34 backbone (details discussed in Section 2.2) implemented from the Segmentation Models library (Pavel Yakubovskiy, 2019) to make use of the transfer learning approach. Sentinel satellite imagery channels and vegetation indices calculated from them, European Space Agency (ESA) WorldCover land cover, and PUGSs dataset from local open-source portals were used as input data. Updated PUGSs dataset at a metropolitan level, created by this study, would improve the understanding of public urban green spaces for better urban area management.

Subsequently, in Section 2 literature review of related studies, and network architectures is presented. In Section 3 input data, preprocessing methods, model setup, training, and metrics are described. Section 4 presents the results, external test performance, and created PUGS dataset. Section 5 is a discussion, and this study is concluded in Section 6.

2. Literature

2.1 Related work

The literature on the use of remote sensing data for applications in urban planning, environmental science, and others has a long and rich history (Albert et al., 2017). Recently with technological advancement and the evolution of deep learning automatization of the acquisition of urban green spaces is possible through the detection of spectral and geometric patterns available in satellite imagery (Khryaschev & Ivanovsky, 2019).

Convolutional Networks, precisely a U-Net architecture with ResNet34 and ResNet50 backbone were used to study urban green spaces by Huerta et al. (2021). They used very high-resolution satellite imagery, WorldView-2 with 0,5m spatial resolution, and focused on one area. They had green, red, and NIR bands, and they calculated Normalized Difference Vegetation Index (NDVI), Two-band Enhanced Vegetation Index (EVI2), and Normalized Difference Water Index (NDWI) from which they made three-band compositions to check which gives the best results. For measuring the model performance, they used the F1 score, known also as Dice coefficient, which is often used as a metric to evaluate semantic segmentation when there is class imbalance (F1 score is also explained in Section 3.5). They found that NDVI–red–NIR composition gave them the best results using the ResNet34 encoder with

a F1 score of 0.5748 and an accuracy of 0.9503, and that different band compositions are producing models with a different performance level.

A more global, and open-source approach had Albert et al., (2017). They used convolutional neural networks, Google Maps satellite images of 1.2 meters of spatial resolution and the Urban Atlas dataset. They used two different architectures – ResNet50 and VGG-16 and found that ResNet50 gave better results for their purpose. They also studied the transferability of their models, so the model learned on data from one city was applied to another city. They found that the model trained on a diverse set (few cities) yielded better performance when applied at different locations than models trained on individual cities. They also tested different tile sizes and found that 224x224 was the best tile size for them and suggested that images at smaller scales might not capture enough variation in urban form. They used only 3 visible bands – green, blue, and red.

While the previous studies used only spectral bands and combination of indices, GülçiN and Akpınar (2018) were doing Object-Based Image Analysis (OBIA) to combine spectral and shape features to map urban green spaces. They were extracting the features from orthophoto that had three visible bands – blue, green, and red in 10 cm resolution. They also calculated Green-Red Vegetation Index (GRVI). The study was focused on one area. They firstly did a segmentation and selected optimal values for the main parameters. To combine spectral and shape features, multiresolution segmentation was implemented in this research which has been proven to be one of the most successful image segmentation algorithms in the OBIA framework, though they did not use a deep learning approach.

2.2 Neural network architectures

The typical use of convolutional networks is on classification tasks, where for one input image a single or few class labels are the output. Although in many visual tasks, such as this study, desired output should include localization, i.e., a class label is supposed to be assigned to each pixel (Ronneberger et al., 2015). This is a semantic segmentation approach, and there are at least few different model architectures that can be used for such purpose.

First worth mentioning is Pyramid Scene Parsing Network (PSPNet) created by Zhao et al. (2016), which is a semantic segmentation model that utilises a pyramid parsing module, meaning that it has kernels of smaller sizes like 3x3 and bigger ones like 6x6, and etc. Pyramid parsing module is applied to harvest different sub-region representations, followed by upsampling and concatenation layers to form the final feature representation, which carries both local and global context information. Finally, the representation is fed into a convolution layer to get the final per-pixel prediction. The local and global

clues together make the final prediction more reliable (Zhao et al., 2016). Its architecture is presented in Figure 1.

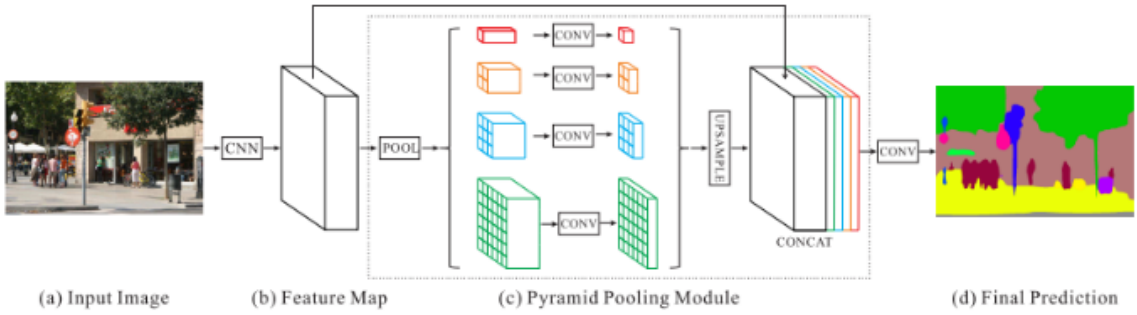


Figure 1 PSPNet architecture (Zhao et al., 2016)

Second one is LinkNet created by Chaurasia and Culurciello (2017). This network was created to make semantic segmentation in real time for tasks such as self-driving vehicles, augmented reality, etc. It is achieved by efficiently sharing the information learnt by the encoder with the decoder after each downsampling block. This proves to be better than using pooling indices in decoder or just using fully convolutional networks in decoder. This feature forwarding technique gives good accuracy values (Chaurasia & Culurciello, 2017). Its architecture is presented in Figure 2.

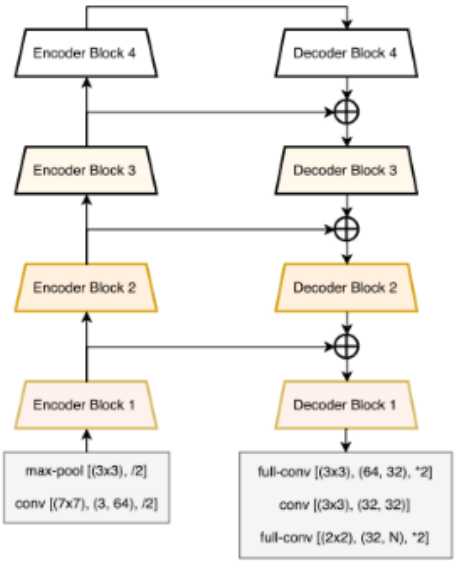


Figure 2 LinkNet architecture (Chaurasia & Culurciello, 2017).

Third example is a U-Net model architecture. It is a convolutional network that was created for biomedical image segmentation by Ronneberger et al. (2015) but various geoscience studies already used it for the segmentation or classification of satellite images and achieved good results (Huerta et al., 2021; Ulmas & Liiv, 2020; Rakhlin et al., 2018). Its architecture is presented in Figure 3.

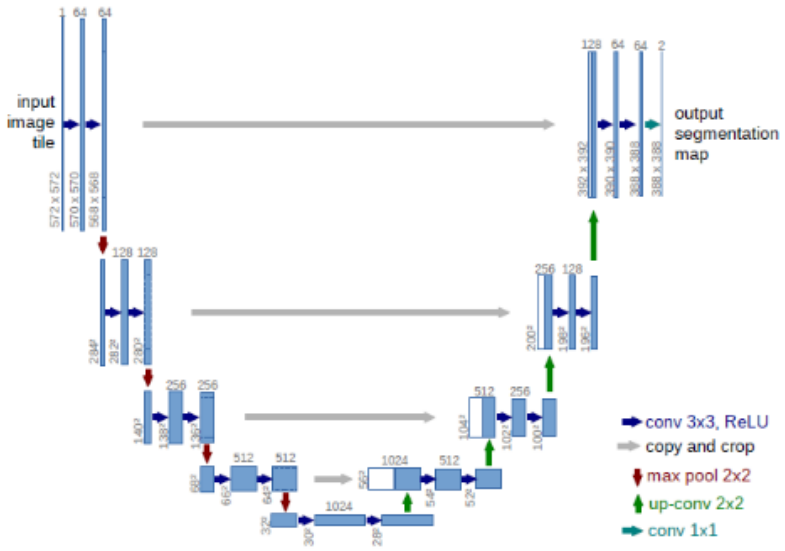


Figure 3 U-net architecture example from Ronneberger et al. (2015) Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations.

The U-Net architecture is based on a Fully Convolutional Network (FCN) and similarly has symmetric encoder–decoder composition. Encoder or sometimes called 'contracting path' is on the left side, and it's responsible for capturing context. It follows the typical architecture of a convolutional network, and it's used to detect features on an image. It consists of the repeated application of two 3x3 convolutions, each followed by a rectified linear unit (ReLU) and a 2x2 max pooling operation with stride 2 for downsampling. Each downsampling step doubles the number of feature channels and makes the feature maps' size smaller (Ulmas & Liiv, 2020).

The second, right part is called a decoder, and it's responsible for upsampling the feature map to the input image's original size. It consists of a 2x2 convolution ('up-convolution') that halves the number of feature channels, a concatenation with the correspondingly cropped feature map from the encoder part, and two 3x3 convolutions, each followed by a ReLU. The cropping is necessary due to the loss of border pixels in every convolution. At the final layer, a 1x1 convolution is used to map each 64-component feature vector to the desired number of classes. In total U-Net has 23 convolutional layers (Ronneberger et al., 2015). The second, decoder part is bringing back the spatial information of the input image (Ulmas & Liiv, 2020).

The biggest advantage of U-Net architecture is that it effectively fuses low-level and high-level image features by combining low-resolution and high-resolution feature maps through skip connections. That's why it's so useful for geoscience and segmentation purposes, or other when we want to have the output size that is equal to the inputs' size and not only labels of the detected features.

Some studies tried using a backbone approach, so using different architectures as an encoder in their chosen CNN architecture. A backbone network is used to extract basic features for detecting objects, and usually designed originally for image classification and pre-trained on the ImageNet dataset. If a backbone can extract more representational features, its host detector will perform better accordingly. In other words, a more powerful backbone can bring better detection performance (Liang et al., 2021). Various studies used this approach of using a pre-trained encoder (backbone) in other CNN architecture (Huerta et al. (2021); Ulmas and Liiv (2020); Pollatos et al. (2020), so a combination of this approach and transfer learning, and found this to be successful.

Transfer learning approach allows to use the learning gained in the first task to solve a new, more complex task (Ulmas & Liiv, 2020). This approach was mentioned by other studies to yield both better performance and faster training times because the network already has learned to recognize basic shapes and patterns (Albert et al., 2017). It is executed by using a pre-trained weights as the starting values for weights while training model for a second task, and as mentioned earlier it is often done by adding an encoder that has pre-trained weights to chosen architecture (Huerta et al. (2021); Ulmas and Liiv (2020); Pollatos et al. (2020)).

Various studies used different model architectures as encoders, ResNet50 was used as an encoder in a U-Net architecture by Ulmas and Liiv (2020) and by Pollatos et al. (2020). ResNet34 encoder in a U-Net architecture was compared with other encoders and was found the best by Rakhlin et al. (2018) and Huerta et al. (2021) for similar to this studies' tasks.

ResNet34 is a residual model with 34 layers, which uses a repeating pattern of layer blocks (Figure 4), with skip connections. They are to avoid the leak gradient problem, which helps to maintain performance and precision despite increases in the number of training layers (He et al., 2015). When ResNet34 is used in U-Net architecture as the encoder it is used to detect features on an image, and carries out a downsampling process, as in encoder–decoder composition explained earlier. At the point of the greatest compression in the network, a decoder is attached that follows the principle of the U-Net architecture to finally obtain an output equal in size to the input images' (Huerta et al., 2021).

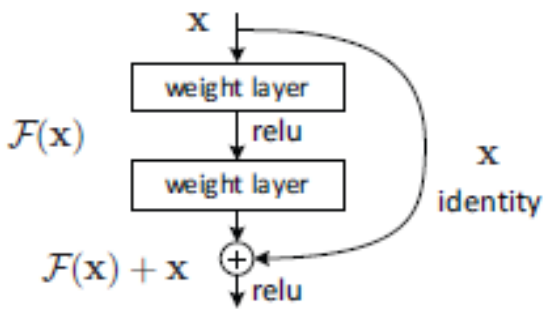


Figure 4 Residual learning: a building block from Resnet architecture (He et al., 2015)

3. Data and Methods

The workflow of the whole methodological process is presented in Figure 5.

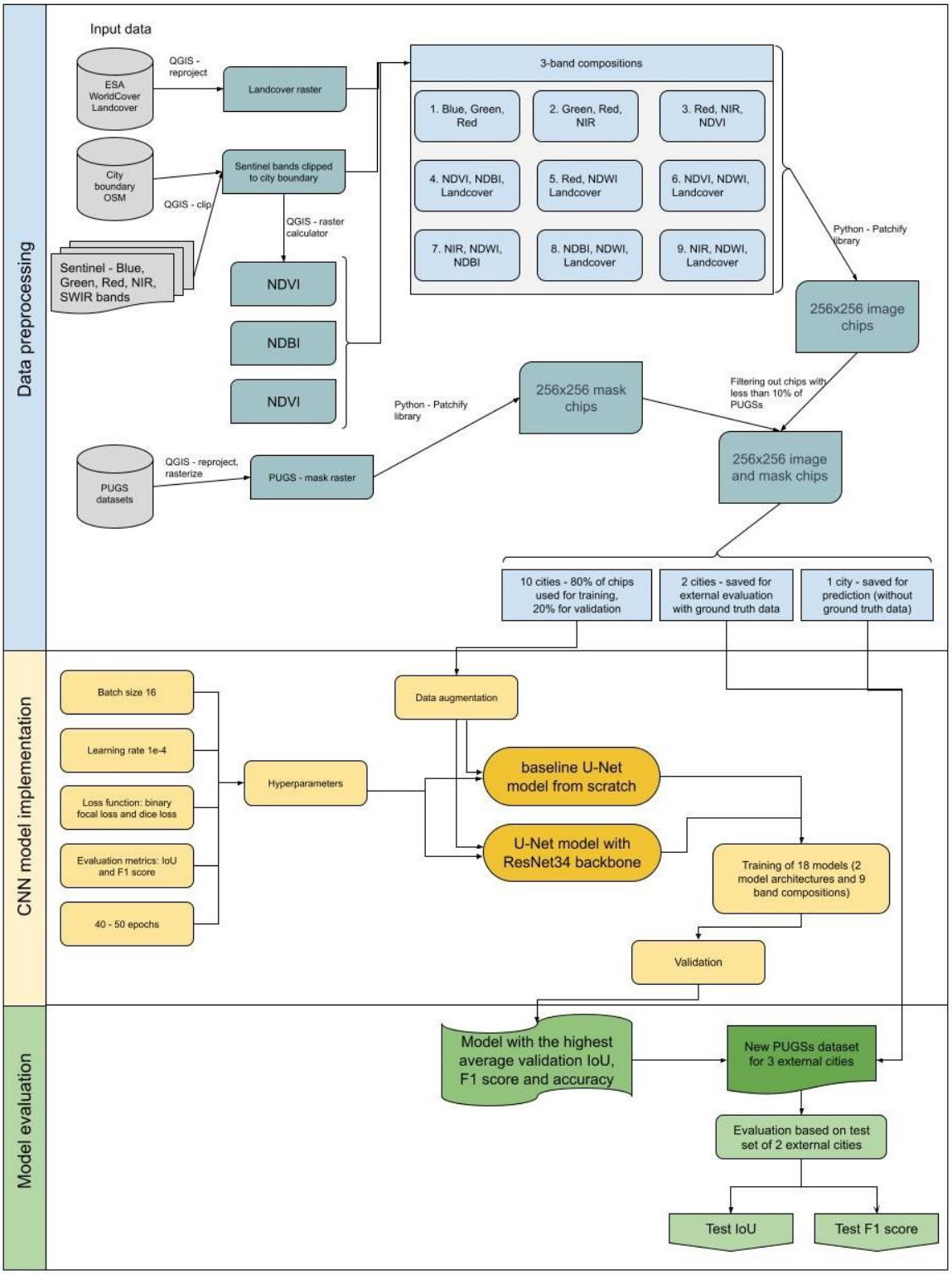


Figure 5 Summary of the current methodological process for semantic segmentation of PUGSs using deep learning. Input data in grey. Data preprocessing in blue, CNN model setup and implementation in yellow, and evaluation in green. Abbreviation: Public Urban Green Spaces (PUGS). Visualisation methods adapted from Huerta et al. (2021)

3.1 Study area

This study was not centred around one particular case study area, instead, it focused on multiple cities from across the world with a focus on European, and US cities. The cities, that were used in this study are presented in Table 1. Ten cities were used as training cities, two cities, Tel Aviv and Washington, were used for external testing, and Kampala was used for prediction.

Training and validation:			
Number	City Name	Country Name	Coordinates
1	Amsterdam	The Netherlands	52° 22' 40'' N 4° 53' 49'' E
2	Buffalo	USA	42° 52' 48'' N 78° 52' 43'' W.
3	Dhaka	Bangladesh	23° 46' 37 '' N 90° 23' 58'' E
4	Dublin	Ireland	53° 21' 0 '' N 6° 15' 58'' W
5	Ghent	Belgium	51° 2' 59'' N 3° 43' 59'' E
6	London	United Kingdom	51° 30' 35''N 0° 7' 5'' W
7	Manchester	United Kingdom	53° 29' 2'' N 2° 14' 40'' W
8	Philadelphia	USA	39° 57' 9'' N 75° 9' 54'' W
9	Seattle	USA	47° 36' 28'' N 122° 20' 6'' W
10	Vancouver	Canada	49° 14' 46'' N 123° 6' 58'' W
External testing and prediction:			
1	Tel Aviv	Israel	32° 6' 33'' N 34° 51' 19'' E
2	Washington	USA	47° 45' 3'' N 120° 44' 24'' W
3	Kampala	Uganda	0° 20' 51'' N 32° 34' 57'' E

Table 1 Cities included in this project

3.2 Input data

In this project a few different data sources were used: satellite images, public urban greenspaces datasets, landcover, and city boundary data.

Satellite images that were used are Sentinel-2 multispectral images. They were downloaded from the open-source online tool EarthExplorer (U.S. Geological Survey). Downloaded images had different image acquisition dates, and those dates are presented in Table 2. Sentinel-2 has radiometric and geometric corrections along with orthorectification to generate highly accurate geolocated products. It is especially important when we are using it to analyse areas in different parts of the world.

Sentinel-2 has 13 spectral bands with spatial resolutions of 10 m (three visible and a near-infrared band), 20m (6 red-edge/shortwave infrared bands), and 60m (3 atmospheric correction bands). In this study blue, green, red, near-infrared and short-wave infrared bands were used.

PUGS data was collected from different open-source data sources. Those sources are presented in Table 2 along with Sentinel image acquisition dates.

City	Name of PUGS data source	Link to PUGS data source	Sentinel image acquisition dates
Buffalo	"Protected Areas Database of the United States (PAD-US) 2.1" database	https://doi.org/10.5066/P92QM3NT	20.06.2020
Philadelphia			09.06.2020
Seattle			26.06.2020
Washington			14.07.2020
London	OS Mastermap Greenspace Layer	https://www.ordnancesurvey.co.uk/business-government/products/open-map-greenspace	25.06.2020
Manchester			29.05.2020
Amsterdam	Gemeente Amsterdam	https://maps.amsterdam.nl/open_geodata/?k=99	30.05.2020
Dhaka	Detail Area Plan (DAP) by RAJUK (City planning authority)	http://43.243.207.51/Website/R2_2_1/viewer.htm	17.01.2020
Dublin	Dublinked	https://data.smartdublin.ie/	01.06.2020
Ghent	Open Data Stad Gent	https://data.stad.gent/explore/dataset/parken-gent/table/	24.06.2020
Tel-Aviv	TLV Open Data	https://opendata.tel-aviv.gov.il/en/pages/item.aspx?ids=10	14.07.2020
Vancouver	City of Vancouver Open Data Portal	https://opendata.vancouver.ca/explore/dataset/parks-polygon-representation/information/	29.06.2020
Kampala	-	-	21.09.2020

Table 2 Data sources for PUGS and dates for

Landcover data was also used – European Space Agency (ESA) WorldCover, which is a worldwide land cover mapping, that provides a new baseline global land cover product at 10 m resolution for 2020 based on Sentinel-1 and 2 data (European Space Agency). Landcover data was used to evaluate if adding already classified data would be beneficial for the segmentation model performance.

The city bounding boxes, which were used to crop the satellite images were extracted from OpenStreetMap using the OSMnx package.

3.3 Data preprocessing

Preprocessing was done in QGIS 3.16.14 open-source software and included: cropping satellite images and landcover to city bounding boxes, resampling SWIR band to 10 m resolution, rasterization of vector urban greenspace dataset, and finally calculating 3 indices:

1. Normalized Difference Vegetation Index (NDVI)

$$NDVI = \frac{NIR - Red}{NIR + Red}$$

   where NIR represents the near-infrared channel and Red is the red channel

2. Normalised Difference Build-up Index (NDBI)

$$NDBI = \frac{SWIR - NIR}{SWIR + NIR}$$

   Where SWIR represents the short-wave infrared channel, and NIR the near-infrared channel

3. Normalised Difference Water Index (NDWI)

$$NDWI = \frac{Green - NIR}{Green + NIR}$$

   where NIR represents the near-infrared channel and Green is the green channel

After the QGIS preprocessing two rasters per city were created, first one with 8 bands – blue, green, red, NIR, NDVI, NDBI, NDWI and ESA landcover, and second categorical raster with two values – 1 representing PUGSs and 0 representing the background. Both rasters and all their bands had a 10 m spatial resolution.

3.4 Model setup

To train an image processing model image needs to be divided into smaller parts, called chips or patches (Huerta et al., 2021). Creating these chips was the first part of the model setup and was done in Python, on the Google Colab platform with the Google Colab Pro subscription.

Nine different three-band compositions were chosen, similar to what Huerta et al. (2021) did. Those compositions were: Blue-Green-Red, Green-Red-NIR, Red-NIR-NDVI, NDVI-NDBI-Landcover, Red-NDWI-Landcover, NDVI-NDWI-Landcover, NIR-NDWI-NDBI, NDBI-NDWI-Landcover, and NIR-NDWI-Landcover. For each band composition, one set of image chips per training city was done. This process included creating image chips for each city from both the satellite image and the mask raster. All the chips had a size of 256x256, but they were created with a different stride. Stride represents a step between two neighbouring chips, so a number of pixels between two consecutive chips. When stride is smaller than the chip size then chips are overlapping. Smaller stride, e.g., 30 makes the chips overlap more than bigger stride e.g., 60. Strides were chosen for each city separately depending on the city's size, which is more explained in section 4.1. Chips were created with the Patchify library

(Weiyuan Wu et al., 2021) and were saved as NumPy arrays on Google drive. Examples of chips are presented in Figure 6.

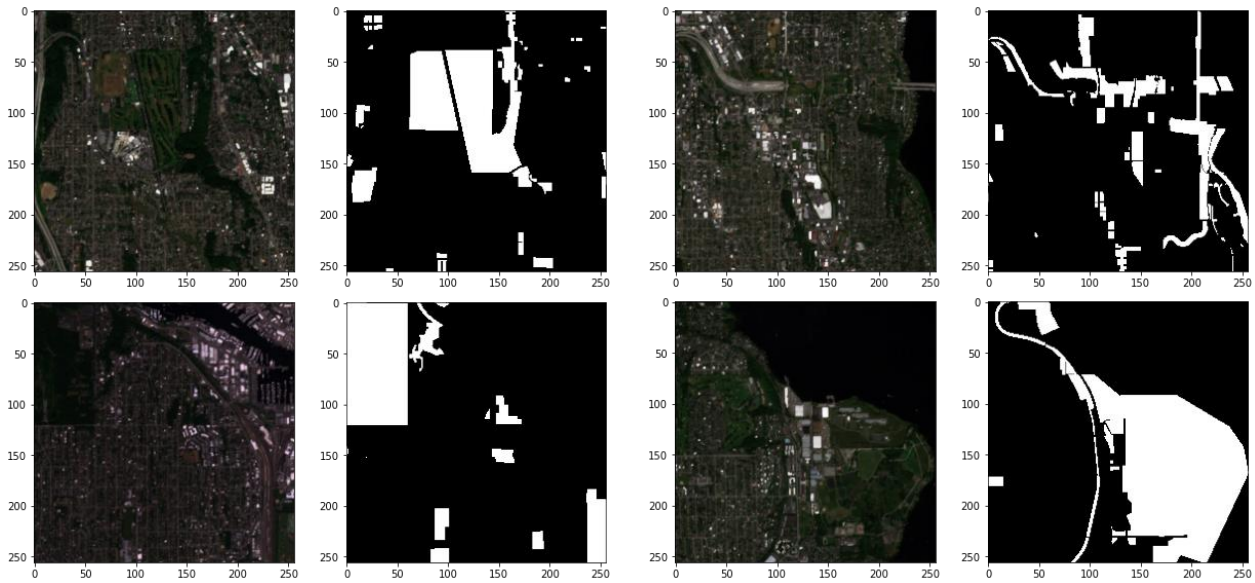


Figure 6 Examples of image chips

The amount of public green space differs between cities with Amsterdam and San Francisco having only 13%, and Dublin 26%, but it rarely goes up to 30% or 40% (World Cities Culture Forum), which makes the dataset imbalanced. Because of that, it was decided to filter out those image chips that have a smaller percentage of public green spaces than the given threshold of 10%. It was done to get a more balanced dataset to train the models on, and to make the data load smaller.

Next step was data augmentation. It was done using the ImageDataGenerator library, with a batch size of 8, and included:

- rotation of images up to 45°,
- shift of the pixels of the image horizontally by a range of 0.1,
- shift of the pixels of the image vertically by a range of 0.1,
- shear up to 0.2,
- zoom up to 0.1,
- random horizontal flip,
- random vertical flip,
- reflect fill mode

These techniques generated transformations for each epoch within the models and increased the size of the training samples.

Input data was also split to training and validation sets with the 80% of input data used for training and 20% for validation. When using a model with a backbone setup also included preprocessing the training and validation data to fit its architecture, and it was done using a function from the Segmentation Models library, from where the model was also from.

3.5 Model calibration and training

Eighteen different semantic segmentation models were implemented because there were nine three-band compositions and two CNN models used. The first model was a baseline U-Net model from scratch, so built layer by layer. Second model was a U-Net with a ResNet34 backbone that had weights pre-trained on the 2012 ILSVRC ImageNet dataset, and it was implemented from the Segmentation Models library. All the models were trained and validated to choose the best model.

The first model had each layer built separately, thanks to which there was a possibility to see how each layer was built and alter the dropout rate or size of the convolution kernel. This model was built from 9 layers, had a dropout between 0.2 and 0.5 at each layer, and a sigmoid activation function. For the training a learning rate of 1e-4, a batch size of 16, and 50 epochs were used.

The second model was implemented from a Segmentation Models library, so it was more like a 'black box' approach, but it had a ResNet34 backbone with weight pre-trained on the 2012 ILSVRC ImageNet dataset. Using transfer learning in this study was a great way to try to improve model performance and tackle the limited RAM issue. Learning rate of 1e-4, batch size of 16, and 40 epochs were used.

Setup and training of the models was done in Google Colab with the Pro subscription, with provided Tesla P100 PCIe 16 GB GPU and 25 GB of available RAM. This allowed for using data from 10 cities, after deleting chips with less than 10% of PUGSs and using three-band compositions to train models. The data from 10 training cities were divided into two sets with the proportion of 80% for training and 20% for validation. Validation set was used to choose the best model. Additional data from 2 external cities were used for testing and evaluation of the chosen model. There is also a third external city for prediction purposes.

In machine learning tasks, the loss function is used to evaluate the difference between training results and labelled data. To decide which loss function fits the best in this semantic segmentation project two factors were taken under consideration – binary aspect, and imbalance. This model has two output classes – PUGSs and background, which makes it a binary case. As mentioned before PUGSs are a small portion of a city. It can be between a few to a few dozens of percent, but usually below 40%. This causes an imbalance that could produce errors and bias towards the background class that covers most of the area of interest. Because of that, it was decided to use a combination of two loss functions that

were found to be useful by others – binary focal loss (Lin et al., 2017) and dice loss (Huerta et al., 2021). Both losses were used from a Segmentation Models library (Pavel Yakubovskiy, 2019), and the used total loss was calculated as follows:

$$\text{Total loss} = \text{binary focal loss} + \text{dice loss}$$

Binary focal loss is a focal loss for binary classification. This loss function generalizes binary cross-entropy by introducing a hyperparameter called the focusing parameter that allows hard-to-classify examples to be penalized more heavily relative to easy-to-classify examples. Therefore, the focal loss is designed to address the scenario in which there is an extreme imbalance between foreground and background classes during training (Lin et al., 2017), and the library implementation defines it as:

$$FL(gt, pr) = -gt\alpha(1-pr)^{\gamma}\log(pr) - (1-gt)\alpha pr^{\gamma}\log(1-pr)$$

Where:

- gt – pixel value of ground truth
- pr – pixel value of prediction
- α – float or integer, the same as a weighting factor in balanced cross entropy, used a default value of 0.25
- γ – float or integer, a focusing parameter for modulating factor (1 - p), using the default value of 2

Semantic segmentation studies that used deep learning have proved that the dice coefficient (F1 score) is a loss function adequate for configurations when there is big imbalance of classes. Using it helps preventing errors and bias towards the background class that covers most of the area (Huerta et al., 2021). The Dice coefficient is a measure of overlap between two sets and was used both as a loss function and as a measurement met. The dice loss in the used library is defined as follows:

$$DL(tp, fp, fn) = \frac{(1+\beta^2) * tp}{(1+\beta^2) * fp + \beta^2 * fn + fp}$$

Where:

- tp – true positive
- fp – false positive
- fn – false negative
- β – float or integer coefficient for precision and recall balance, using the default value of 1

The F1 score (dice coefficient) was also used as a metric in this study and can also be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and

worst score at 0. The relative contribution of precision and recall to the F1 score are equal. The formula for the F1 score (dice coefficient) using precision and recall in the used library is defined as follows:

$$F1(precision, recall) = \frac{2 * precision * recall}{precision + recall}$$

When it comes to assessing the performance of the model this study used Intersection over Union (IoU) as the first performance measurement, and already explained F1 score as the second one. This is because there is a lot of imbalance in the dataset, so those measures fit that configuration best.

IoU, also known as the Jaccard index computes the amount of overlap between the predicted polygons and the ground truth data (Huerta et al., 2021), with values > 0.50 considered to be correctly predicted. The used function in the library defines it as follows:

$$IOU(A, B) = \frac{A \cap B}{A \cup B}$$

Where:

- A - the input ground truth,
- B - output segmentation.

4. Results

4.1 Model setup results

The model setup included steps like creating chips, filtering them and data augmentation. For the process of creating data chips different strides were used for different cities based on their size, and proportions of PUGSs in the city. The stride for each city was chosen arbitrarily and it is presented along with the final chips number in Table 3. The presented number of chips is after filtering out chips that had less than 10% of PUGSs on them.

From this data, it is clear, that there is an imbalance, and PUGSs pixels are a lot smaller percentage of input images then the background pixels. The proportion presented in this table though cannot be treated as a proportion of PUGSs in a city, but in the city bounding box. Bounding box is a bigger area than the city itself, making the imbalance bigger. That also caused that there were parts of the mask raster that had only background pixels, which created some chips without any PUGSs pixels.

It is visible that the data is imbalanced also when it comes to city representation. Some cities had more chips than the others, and cities with the most chips were Manchester, London, and Philadelphia. Manchester and London are both a lot bigger than the rest of the cities with an average percentage

of PUGSs pixels. Philadelphia on the other hand is not that bigger than the rest, it had a lot more chips because it had a higher percentage of PUGSs pixels.

No.	City Name	Image size after clipping to city boundaries (in km^2)	Percentage of PUGSs pixels	Used stride	Percentage of used chips	Number of chips after filtering
1	Amsterdam	2,6662	5,70 %	24	21,55%	723
2	Buffalo	1,5154	6,71 %	20	25,42%	604
3	Dhaka	1,9942	1,09 %	20	1,04%	34
4	Dublin	2,1066	9,15 %	25	30,47%	712
5	Ghent	3,9967	2,08 %	20	5,62%	435
6	London	27,1262	5,33 %	65	18,19%	1060
7	Manchester	19,5396	5,76 %	65	26,43%	1100
8	Philadelphia	7,9294	9,48 %	45	33,51%	1107
9	Seattle	3,8969	6,97 %	30	24,45%	812
10	Vancouver	1,8275	6,26 %	30	14,06%	429

Table 3 List of cities included in the project along with model setup results

Model setup also included data augmentation of matching image and mask chips. Examples are presented on Figure 7.

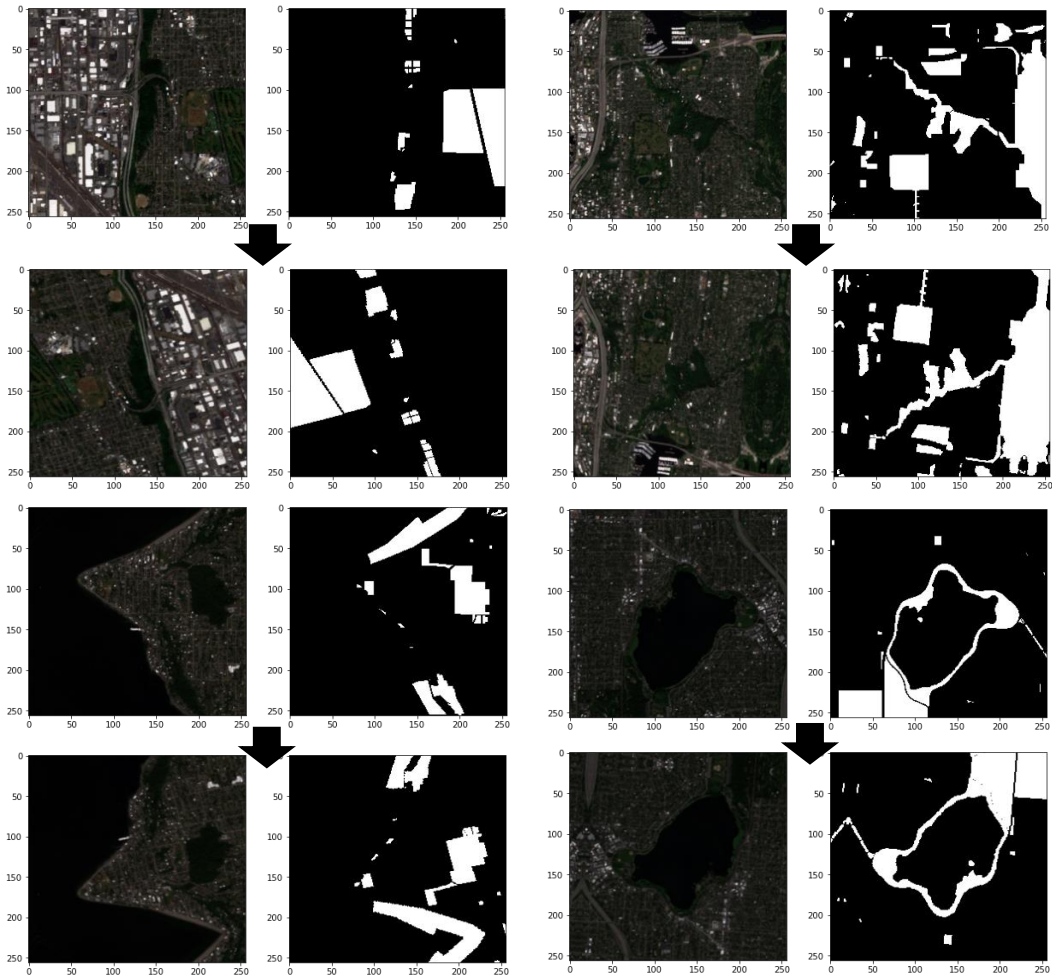


Figure 7 Examples of data augmentation on image and mask chips. First and third row are original chips, second and fourth augmented ones

4.2 Training, validation results and choosing the best model

In this section training and validation results are presented for all the created models. Training set was 80% of the input data, and validation was 20%. There were 18 models created in total, because there were 9 different three-band compositions, and two different model architectures – baseline U-Net model from scratch and U-Net model with a pre-trained ResNet34 encoder. All the models were trained on 10 cities mentioned in Table 2. The aim was to find the best combination of model architecture and band composition based on validation metrics. Table 4 shows the results for baseline U-Net model for scratch, and Table 5 for U-Net with a pre-trained Resnet34 backbone.

Baseline U-Net model from scratch							
	Training			Validation			
Band composition	Accuracy	IoU	F1 score	Accuracy	IoU	F1 score	Validation average
Blue-Green-Red	0,9249	0,6753	0,8055	0,9212	0,6581	0,7931	0,7908
Green-Red- NIR	0,9436	0,7435	0,8524	0,9406	0,7286	0,8423	0,8372
Red-NIR-NDVI	0,9405	0,7241	0,8395	0,9388	0,7139	0,8325	0,8284
NDVI-NDBI-Landcover	0,9485	0,7565	0,8610	0,9465	0,7444	0,8530	0,8480
Red-NDWI-Landcover	0,9450	0,7393	0,8497	0,9430	0,7257	0,8401	0,8363
NDVI-NDWI-Landcover	0,9495	0,7610	0,8639	0,9478	0,7493	0,8561	0,8511
NIR-NDWI-NDBI	0,9445	0,7406	0,8505	0,9422	0,7278	0,8418	0,8373
NDBI-NDWI-Landcover	0,9503	0,7660	0,8672	0,9471	0,7487	0,8558	0,8505
NIR-NDWI-Landcover	0,9489	0,7568	0,8611	0,9455	0,7370	0,8479	0,8435
Average	0,9440	0,7403	0,8501	0,9414	0,7259	0,8403	

Table 4 Training and validation accuracy, IoU, and F1Score of different band compositions for baseline U-Net model from scratch. Average validation values for separate band compositions models and average of metrics for all models are in green

For baseline U-Net from scratch average validation IoU was 0,7259, average validation F1 score was 0,8403. For U-Net with ResNet34 encoder it was 0,8681 and 0,9274, which is a 12% rise for validation IoU and 8% for validation F1 score. This visible rise shows that U-Net with a ResNet34 backbone achieved better performance than the baseline model, and that transfer learning is a very good approach. The best model should be chosen from the models with a pre-trained encoder.

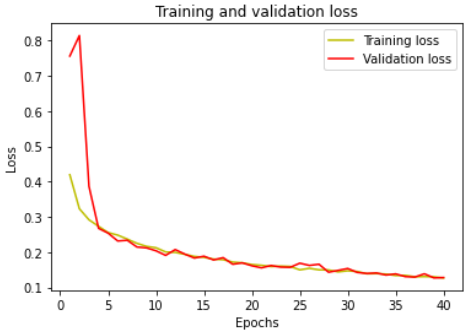
NIR-NDWI-NDBI composition with U-Net with ResNet34 encoder architecture achieved the best average validation score among all the models and was chosen as the best model to be used later for evaluation and predictions. It had validation IoU of 0,8770, and validation F1 score of 0,9326.

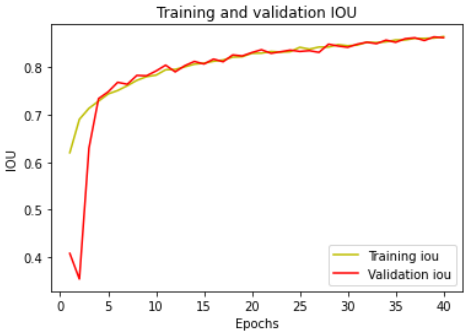
U-Net with a Resnet34 backbone model							
	Training			Validation			
Band composition	Accuracy	IoU	F1 score	Accuracy	IoU	F1 score	Validation average
Blue-Green-Red	0,9552	0,8669	0,9266	0,9534	0,8616	0,9234	0,9128
Green-Red-NIR	0,9581	0,8738	0,9308	0,9562	0,8677	0,9271	0,917
Red-NIR-NDVI	0,9570	0,8709	0,9291	0,9554	0,8659	0,9261	0,9158
NDVI-NDBI-Landcover	0,9548	0,8667	0,9265	0,9530	0,8612	0,9232	0,9125
Red-NDWI-Landcover	0,9607	0,8810	0,9351	0,9595	0,8769	0,9326	0,923
NDVI-NDWI-Landcover	0,9564	0,8699	0,9284	0,9545	0,8632	0,9243	0,914
NIR-NDWI-NDBI	0,9611	0,8813	0,9352	0,9600	0,8770	0,9326	0,9232
NDBI-NDWI-Landcover	0,9601	0,8787	0,9338	0,9586	0,8740	0,9310	0,9212
NIR-NDWI-Landcover	0,9573	0,8712	0,9293	0,9555	0,8656	0,9259	0,9157
Average	0,9579	0,8734	0,9305	0,9562	0,8681	0,9274	

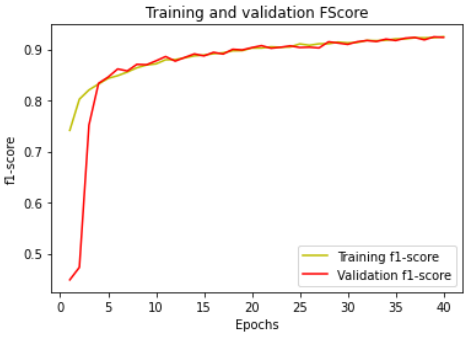
Table 5 Training and validation accuracy, IoU, and F1Score of different band compositions for U-Net with a Resnet34 backbone model. Average validation values for separate band compositions models and average of metrics for all models are in green. The chosen model is in light green

Transfer learning is not only helping in achieving better performance, but also in making the training process converge faster. For U-Net with ResNet34 encoder, the learning process is also less bumpy than for the model from scratch, meaning that the loss and IoU are changing more smoothly. Figure 8 compares the process of the training of the chosen best model, so NIR-NDWI-NDBI composition with U-Net with ResNet34 encoder and NDVI-NDWI-Landcover for baseline U-Net model from scratch, which had the highest validation average scores across the models from scratch.

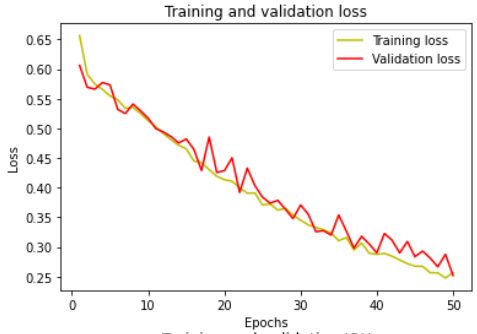
NIR-NDWI-NDBI U-Net with a ResNet34 backbone

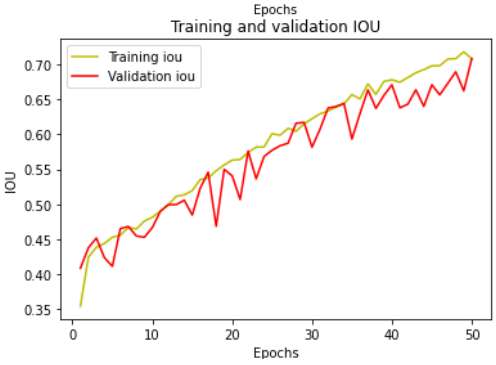






NDVI-NDWI-Landcover baseline U-Net model





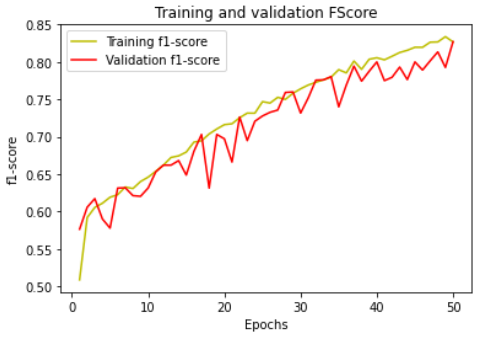


Figure 8 Comparison of the training process between the NIR-NDWI-NDBI U-Net with a ResNet34 backbone and NDVI-NDWI-Landcover baseline U-Net from scratch

4.3 External evaluation and prediction of PUGSs based on the best model

External evaluation was based on test set that consisted of two external cities – Washington and Tel Aviv. Both of those cities were not a part of testing or validation process. Test performance of the chosen model, so the NIR-NDWI-NDBI U-Net with a ResNet34 encoder is presented in Table 6.

	Washington		Tel Aviv	
	IoU	F1 score	IoU	F1 score
NIR-NDWI-NDBI U-Net with a Resnet34 encoder	0,6084	0,7160	0,5137	0,5743

Table 6 External evaluation (testing) of the NIR-NDWI-NDBI U-Net with a Resnet34 encoder model

The best model achieved an average test IoU of 0,5610, and average test F1 score of 0,64515 across two external cities. Washington had an average of 0,6622 test metrics, and Tel Aviv had 0,544, so the model performed visibly better on Washington then on Tel Aviv.

Differences between validation metrics of baseline models from scratch and models with a backbone were visible, but when we compare their prediction on external dataset this difference in performance is even more apparent. Figure 9 illustrates differences in prediction of Washington PUGSs between chosen NIR-NDWI-NDBI U-Net with a Resnet34 encoder, and model from scratch that achieved the best validation metrics, so NDVI-NDWI-Landcover baseline model. It is evident that the transfer learning model is performing a lot better on an external test set.

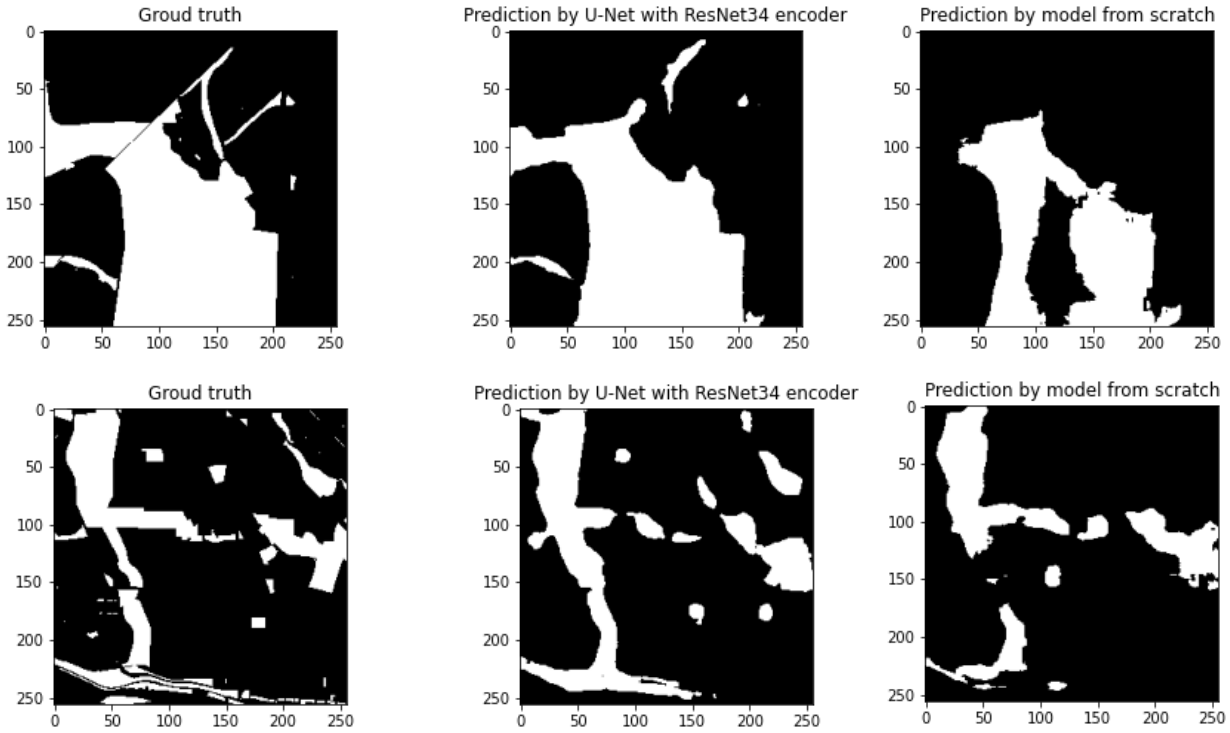


Figure 9 Comparison of PUGSs prediction in Washington. Left – ground truth, middle –the chosen NIR-NDWI-NDBI U-Net with a Resnet34 encoder model, right - best model from scratch, so NDVI-NDWI- Landcover. PUGS are white, and background is black

The best, chosen model, so NIR-NDWI-NDBI U-Net with a Resnet34 encoder was used to create new PUGSs datasets for 3 external cities – Washington, Tel Aviv, and Kampala. Figure 10 presents ground truth PUGSs dataset, and prediction for Washington. Predictions for Tel Aviv and Kampala are shown in the appendix. Looking at the predictions for Washington there are some parts, like the big green space in the north, and longitudinal green space in the middle, that were predicted good, but there are also some misclassifications that should be analysed up close.

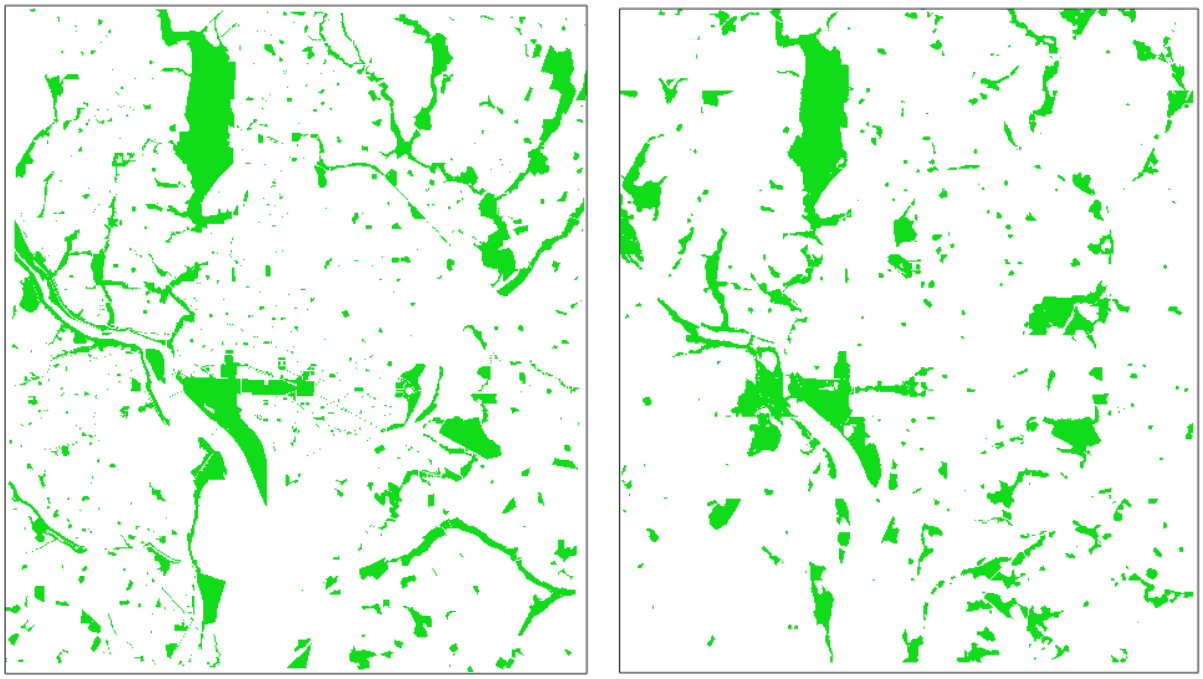


Figure 10 Left - ground truth PUGS data for Washington, right – predicted PUGS data for Washington.

Looking more closely at the predicted and the ground truth PUGSs data for Washington there are a few groups of misclassifications present:

1. Small PUGSs in high-density neighbourhoods.

   Figure 11 shows a comparison of the true colour Washington image (left) and PUGSs prediction on top of ground truth data (right). These examples show that when there are small PUGSs in a dense neighbourhoods model sometimes fails or predicts just parts of the PUGSs.

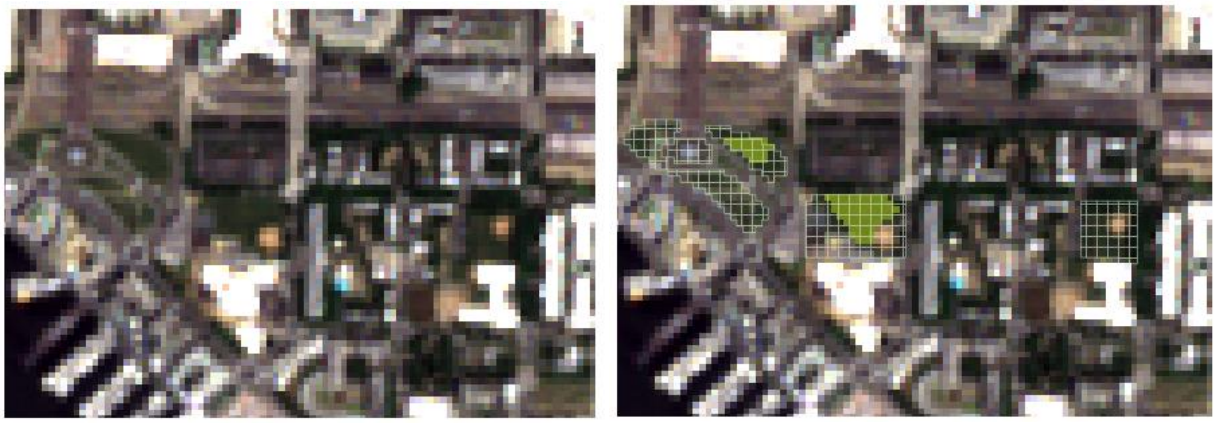


Figure 11 Left – true colour Washington image, right - satellite image, on top of that PUGSs predictions as green, and ground truth PUGSs symbolised with cross filling.

2. Some parts of PUGSs that are build-ups, not green space.

   Figure 12 shows an example when part of a PUGS is infrastructure or building, so a build-up area, not green. The model might predict that those parts are background, not PUGS.

Figure 12 Left – true colour Washington image, right - satellite image, on top of that PUGSs predictions as green, and ground truth PUGSs symbolised with cross filling.

3. Green spaces that were not in the ground truth data, that are probably not public.

   Figure 13 shows an example of an area that was predicted as being a public urban green space, but probably it is a golf club, which is not public.

Figure 13 Left – true colour Washington image, right - satellite image, on top of that PUGSs predictions as green, and ground truth PUGSs symbolised with cross filling.

5. Discussion

5.1 Key findings

Two presented multi-city models created to identify public urban green spaces at the metropolitan level across the world, and their external accuracy, show that this approach is promising. It allows for efficient creation of new PUGSs datasets that yields useful information about the location, geometry, and other spatial attributes. This can be very helpful for local authorities while managing cities and the decision-making process about future development for cities.

The transfer learning approach shows a really big improvement comparing to a baseline model built from scratch. ResNet34, which was used as an encoder for a U-Net model was pre-trained on the 2012 ILSVRC ImageNet dataset on blue, green, and red bands. The benefits coming from this approach are visible – average validation performance for U-Net with ResNet34 backbone is better than the baseline model built from scratch, and fewer epochs are needed for training.

Cities that these models were trained on were mostly from the US and Europe, and model performance on external testing dataset is higher for Washington than for Tel Aviv. Washington is a US city and is spatially close to some of the cities that the model was trained on. Being located in the same country and culture makes its structure and urbanization features more similar to training cities than Tel Aviv. Tel Aviv is on a different continent, has a different urbanization style, and is a city that has neighbourhoods that date back to the 14th century BC. Those big differences are probably the reason why Washington has a higher prediction performance.

This approach was based solely on open-source data, which makes it cheaper than using data from third-party providers but is especially important when thinking about cities from developing countries in Africa or Asia. The fastest-growing cities in the world are on those continents (Investment Monitor, 2022), and those are usually the countries that do not have the resources to buy (very) high resolution satellite images.

This study was done using a Google Colab Pro subscription, which shows that fairly advanced and heavy computationally work can be done using not that expensive services. This was because of a relatively big for this kind of study spatial resolution of the images, and deletion of image chips that had less than 10% of PUGSs in them. Both of those things contributed to making the data load smaller, and thus enabled as many as 10 cities to be used for training.

5.2 Literature comparison

This study is one of the many studies that focus on urban analysis with the usage of satellite images and CNN type models. The methodology used in this project is fairly similar to this chosen by Huerta et al. (2021), meaning that U-Net models with a Res-Net encoder are used with a group of three-band compositions to analyse urban green spaces. The biggest difference is that they used satellite images with 0.5 m spatial resolution in contrast to 10 m resolution Sentinel images used by this study. Huerta et al. (2021) also focused on one study area, whereas this study included 10 cities in the training process. They got the best results for NDVI–red–NIR composition using ResNet34 encoder with an F1 score of 0.5748, and evaluation IoU of 0.75. This study got the best results for the NIR-NDWI-NDBI composition with the same architecture - U-Net with ResNet34 encoder. It achieved an average test IoU of 0,5610, and average test F1 score of 0,64515 across two external cities. Model performance was calculated by this study based on 2 external cities that were not part of the training process, and Huerta et al. (2021) used a subset of 1% of the initial training dataset for evaluation purposes.

The second study, done by Albert et al. (2017) was identifying urban patterns, which is a different purpose than this study, but there are still a lot of similarities. Albert et al. (2017) worked on open-source data, but satellite images used by them had a spatial resolution of1,2 m. They experimented with transfer learning and found that it gave them around of 5% increase in performance. For this study the gain was bigger - 12% rise in validation IoU and 8% in validation F1 score. They also noticed that performance was poor when the model was trained on samples from one city and tested on samples from a different city. This was visibly worst when cities were more different, and a bit better when they were more similar. This is in line with the results that this study got, especially that model trained on mainly US and European cities had better performance on Washington than on Tel Aviv.

5.3 Project strengths and limitations

This study aimed at identifying public urban green spaces based on open-source data, and a multi-city model. This was achieved using a transfer learning approach, which made the results a lot better. Most of similar studies were performed on satellite images that had a lot smaller spatial resolution, so achieving an average test IoU of 0,5610, and average test F1 score of 0,64515 on an open-source satellite images is a moderate to a good result, and very promising for future research. Thanks to filtering out some of the image chips more global approach was possible, with 10 cities being used for the training process, which also contributed to a high performance.

There were three main limitations that this study faced. The first one is connected to how complex the definition of public urban green space is. It is hard to determine what is public and what is private with only satellite images and landcover data, and this was probably the main reason for not that high test

performance. This model was trained to identify public urban green spaces but looking at the prediction for external cities, especially examples for Washington presented in 4.3 Section, might suggest that it predicts urban green spaces, both public and private.

The second limitation was computational load, both the RAM capacity and the training time. This study had a global approach, and 10 cities were able to be included after filtering out some chips. It was done using the 25 GB of RAM available for Google Colab Pro subscription. Training time of one model was around an hour for transfer learning, and two hours for baseline model from scratch. This was specifically challenging, because 18 models in total were created.

This project was done using open-source data Sentinel satellite images, which have a quite big spatial resolution when comparing to different studies from this field. This had the disadvantage of being less detailed, yielding less information about objects, and not being able to depict smaller details.

5.4 Recommendations for future research

Future research should focus on improving few aspects. The first one is the number, and the location of cities that were used for training. There is a discrepancy between Washington and Tel Aviv prediction performance, which as discussed above is thought to arise from the fact that more cities similar to Washington were in the training set. So, if similar approach were to be used to predict PUGSs for cities in Asia or Africa, more cities from these areas should also be included in the training process. This should positively influence performance on those cities.

If future research is focused on having higher performance in general, than satellite images with smaller spatial resolution should be taken under consideration. Using them would give so many more details and should give a better performance. Focusing on the difference between private and public urban green spaces could be also done to improve the performance. More various data sources connected to PUGSs, such as their size, proximity to accompanying infrastructure like roads and bridges, frequency of visits or other data about urban features and the ways they are being used could be included in input data to achieve better performance.

Future research could also focus on including even more cities into the training set. To do so mentioned above technical aspects should be improved, such as having more available RAM, having more disc memory to host the data and models, and having a better machine to execute the training process faster. Usage of services that provide ready-to-use image chips such as Google Earth Engine could also be considered.

6. Conclusion

This study aimed at investigating the use of convolutional neural networks for identifying public urban green spaces through multi-source and open-source data at the metropolitan level with a global focus. This goal was attained with a moderate to good performance. This study focused on evaluating two deep learning model techniques for semantic segmentation of PUGSs with the use of U-Net architecture, and U-Net with a pre-trained ResNet34 encoder architecture. It used the transfer learning approach to improve the performance of the created model.

Results show that this approach is promising for creating new, open source based PUGSs datasets for various cities. Especially transfer learning approach, which gave an average test IoU of 0,5610, and average test F1 score of 0,64515 across two external cities. This is a quite good result considering limitations of this study, but also leaves a room for improvement for future research. Using a pre-trained encoder also allowed for using less epochs, which makes the learning process faster.

Usage of similar, deep learning models could improve processes of identification of public urban green spaces for management and development purposes. Nowadays most people live in urban areas and working towards making these cities greener and more sustainable with more accessible green spaces is one of the UN Sustainable Development Goals. If those goals are meant to be achieved new ways of urban management should be created, and this kind of deep learning approach could be part of this process.

Acknowledgements

This thesis project was made in collaboration with S.M. Labib, Department of Human Geography and Spatial Planning at Utrecht University and Jiawei Zhao, MSc in Applied Data Science at Utrecht University.

Appendix

GitHub repository link with all code - https://github.com/mar-koz22/Park-NET-identifying-Urban-parks-using-multi-source-spatial-data-and-Geo-AI/tree/main/Marta's-approach

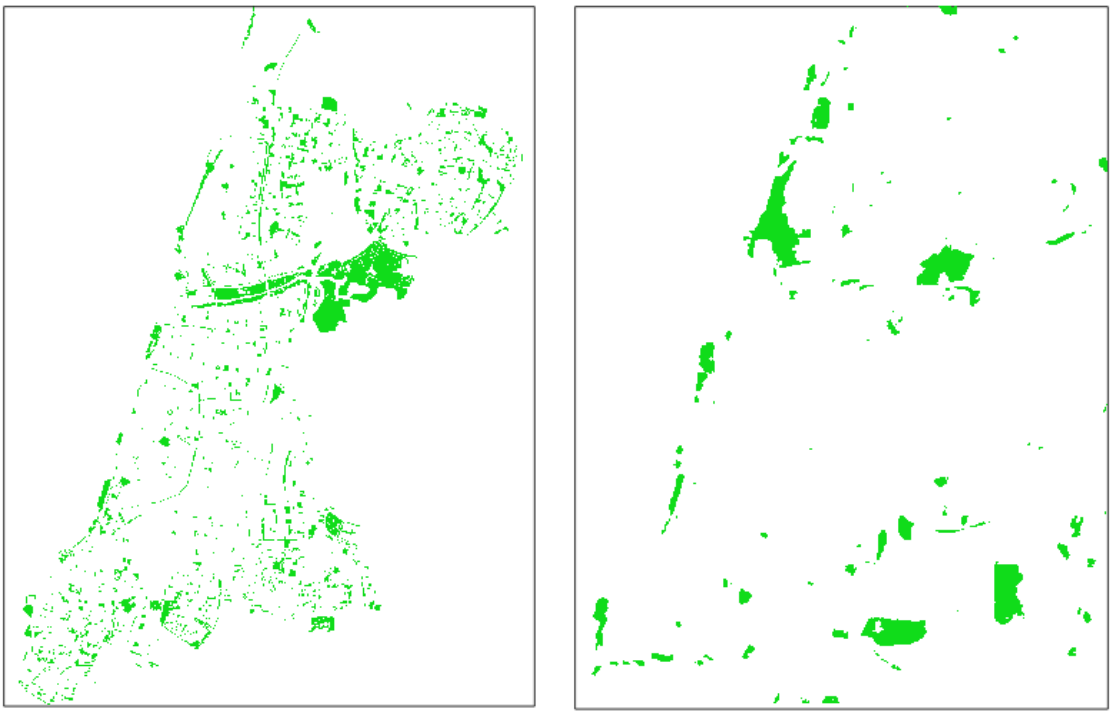


Figure 15 Left - Tel Aviv ground truth PUGSs dataset, right - predicted PUGS data for Tel Aviv using NIR-NDWI-NDBI U-Net with ResNet34 encoder architecture. PUGS are green, background is white

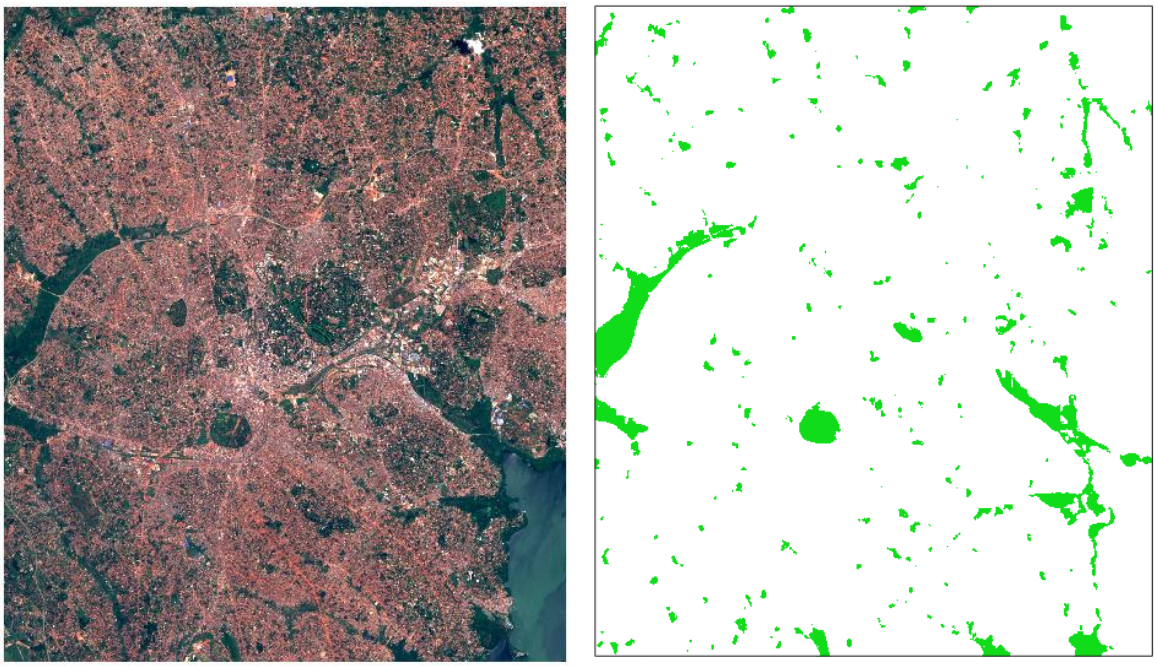


Figure 14 Right – true color satellite image of Kampala, left - predicted PUGS data for Kampala using NIR-NDWI-NDBI U-Net with ResNet34 encoder architecture.
PUGS are green, background is white

References

Albert, A., Kaur, J., & Gonzalez, M. (2017). *Using convolutional networks and satellite imagery to identify patterns in urban environments at a large scale* (arXiv:1704.02965). arXiv. http://arxiv.org/abs/1704.02965

Chaurasia, A., & Culurciello, E. (2017). *LinkNet: Exploiting Encoder Representations for Efficient Semantic Segmentation*. https://doi.org/10.48550/ARXIV.1707.03718

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2014). *Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs*. https://doi.org/10.48550/ARXIV.1412.7062

European Space Agency. (n.d.). *ESA WorldCover*. https://esa-worldcover.org/en

GülçiN, D., & Akpınar, A. (2018). Mapping Urban Green Spaces Based on an Object-Oriented Approach. *Bilge International Journal of Science and Technology Research*, *2*, 71–81. https://doi.org/10.30516/bilgesci.486893

He, K., Zhang, X., Ren, S., & Sun, J. (2015). *Deep Residual Learning for Image Recognition* (arXiv:1512.03385). arXiv. http://arxiv.org/abs/1512.03385

Huerta, R. E., Yépez, F. D., Lozano-García, D. F., Guerra Cobián, V. H., Ferriño Fierro, A. L., de León Gómez, H., Cavazos González, R. A., & Vargas-Martínez, A. (2021). Mapping Urban Green Spaces at the Metropolitan Level Using Very High Resolution Satellite Imagery and Deep Learning Techniques for Semantic Segmentation. *Remote Sensing*, *13*(11), 2031. https://doi.org/10.3390/rs13112031

Investment Monitor. (2022, February). *The ten fastest-growing cities in the world*. https://www.investmentmonitor.ai/analysis/fastest-growing-cities-in-the-world

Khryaschev, V., & Ivanovsky, L. (2019). Urban areas analysis using satellite image segmentation and deep neural network. *E3S Web of Conferences*, *135*, 01064. https://doi.org/10.1051/e3sconf/201913501064

Kopecká, M., Szatmári, D., & Rosina, K. (2017). Analysis of Urban Green Spaces Based on Sentinel-2A: Case Studies from Slovakia. *Land*, *6*(2), 25. https://doi.org/10.3390/land6020025

Liang, T., Chu, X., Liu, Y., Wang, Y., Tang, Z., Chu, W., Chen, J., & Ling, H. (2021). *CBNetV2: A Composite Backbone Network Architecture for Object Detection* (arXiv:2107.00420). arXiv. http://arxiv.org/abs/2107.00420

Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). *Focal Loss for Dense Object Detection*. https://doi.org/10.48550/ARXIV.1708.02002

Ludwig, C., Hecht, R., Lautenbach, S., Schorcht, M., & Zipf, A. (2021). Mapping Public Urban Green Spaces Based on OpenStreetMap and Sentinel-2 Imagery Using Belief Functions. *ISPRS International Journal of Geo-Information*, *10*(4), 251. https://doi.org/10.3390/ijgi10040251

Pavel Yakubovskiy. (2019). Segmentation Models. *GitHub*. https://github.com/qubvel/segmentation_models

Pollatos, V., Kouvaras, L., & Charou, E. (2020). *Land Cover Semantic Segmentation Using ResUNet*. https://doi.org/10.48550/ARXIV.2010.06285

Rakhlin, A., Davydow, A., & Nikolenko, S. (2018). Land Cover Classification from Satellite Imagery with U-Net and Lovász-Softmax Loss. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 257–2574. https://doi.org/10.1109/CVPRW.2018.00048

Ronneberger, O., Fischer, P., & Brox, T. (2015). *U-Net: Convolutional Networks for Biomedical Image Segmentation*. https://doi.org/10.48550/ARXIV.1505.04597

Sharifi, A., & Hosseingholizadeh, M. (2019). The Effect of Rapid Population Growth on Urban Expansion and Destruction of Green Space in Tehran from 1972 to 2017. *Journal of the Indian Society of Remote Sensing*, *47*(6), 1063–1071. https://doi.org/10.1007/s12524-019-00966-y

Skokanová, H., González, I. L., & Slach, T. (2020). Mapping Green Infrastructure Elements Based on Available Data, A Case Study of the Czech Republic. *Journal of Landscape Ecology*, *13*(1), 85–103. https://doi.org/10.2478/jlecol-2020-0006

Ulmas, P., & Liiv, I. (2020). *Segmentation of Satellite Imagery using U-Net Models for Land Cover Classification*. https://doi.org/10.48550/ARXIV.2003.02899

United Nations. (2018, May 16). *68% of the world population projected to live in urban areas by 2050, says UN*. https://www.un.org/development/desa/en/news/population/2018-revision-of-world-urbanization-prospects.html

United Nations Development Programme. (n.d.). *Sustainable Development Goals*. Retrieved 15 June 2022, from https://www.undp.org/sustainable-development-goals#sustainable-cities-and-communities

U.S. Geological Survey. (n.d.). *EarthExplorer*. https://earthexplorer.usgs.gov/

Weiyuan Wu, Divakar Verma, & Wangyin Yang. (n.d.). *Patchify*. https://pypi.org/project/patchify/

World Cities Culture Forum. (n.d.). *% of public green space (parks and gardens)*. http://www.worldcitiesscultureforum.com/data/of-public-green-space-parks-and-gardens

Wurm, M., Stark, T., Zhu, X. X., Weigand, M., & Taubenböck, H. (2019). Semantic segmentation of slums in satellite images using transfer learning on fully convolutional neural networks. *ISPRS Journal of Photogrammetry and Remote Sensing*, *150*, 59–69. https://doi.org/10.1016/j.isprsjprs.2019.02.006

Zhao, H., Shi, J., Qi, X., Wang, X., & Jia, J. (2016). *Pyramid Scene Parsing Network*. https://doi.org/10.48550/ARXIV.1612.01105