Aerial image segmentation for green area detection

HENDRIK HAUGG, SEBASTIAN HUBER, SEBASTIAN KIUNKE, and JONAS LEITNER, Munich University of Applied Sciences, Germany

The paper presents a comprehensive study on the challenges of segmenting green areas in aerial imagery using multiple approaches. The authors explore four different methods for aerial segmentation, including a pure image processing approach, a self-trained U-Net model, the Segment Anything Model (SAM), and the MMSegmentation toolbox.

A detailed description of the research methodology, which includes the acquisition and pre-processing of aerial imagery and the design and training of the deep learning models is being provided. The paper demonstrates the effectiveness of their approach in accurately detecting and segmenting green areas in aerial images using quantitative measures.

The conducted experiments demonstrate that SAM and the pure image processing approach outperform the other two methods. However, the results vary significantly depending on the specific use case. For example, the performance of deep learning models deteriorates in complex images with overlapping objects. The authors suggest further research to improve these models and their application in real-world scenarios. Overall, the paper presents a comprehensive study on various approaches to aerial image segmentation, compares the performance of traditional and deep learning methods and highlights the strengths and weaknesses of different approaches. The paper offers insightful observations and sets the groundwork for future research in this area.

Additional Key Words and Phrases: Segmentation, CNN, SAM

Authors' address: Hendrik Haugg, hendrik.haugg@hm.edu; Sebastian Huber, s.huber@hm.edu; Sebastian Kiunke, skiunke@hm.edu; Jonas Leitner, jonas.leitner@hm.edu, Munich University of Applied Sciences, Germany.

1 INTRODUCTION - (*SEBASTIAN KIUNKE*)

In the last 20 years, from 2002 to 2022, the population in Munich has grown by 22% [8], while the available space has remained constant [4]. This leads to difficulties, as additional inhabitants require living space while the total area remains the same, but at the same time this living space must not displace other necessary areas. In order to ensure a balance between utilised and unutilised areas, the City of Munich collects key figures on land use and land distribution. These areas are further subdivided into buildings, vegetation, transport, agriculture, and soil sealing [4]. The calculation of the area size varies, as buildings and transport are generally larger and planned in advance and are therefore easier to measure. Although this also applies to large areas of vegetation such as the English Garden, it does not apply to small green spaces, e.g. between rows of houses or in areas that are difficult to access, where the cost-benefit factor of manual measurement is not justifiable. In co-operation with the Munich University of Applied Sciences, the Munich Surveying Office is therefore researching ways of automatically recording and measuring vegetation areas solely on the basis of aerial photographs. These colour images are taken from a bird's eye view and show a section of Munich in the respective resolution.

1.1 Contribution

In the course of this paper, various technologies and algorithms are explained and evaluated specifically designed to detect areas of vegetation of aerial photos with high accuracy. The main focus in this paper, based on the lecture of the corresponding name, is on deep learning algorithms and comparing them with conventional approaches.

1.2 Structure

Chapter 1 elaborates the topic and research question. Chapter 2 explains necessary knowledge for the understanding of concepts. Chapter 3 references and describes academic papers written in a similar context to this paper. Chapter 4 describes and explains the approaches of the individual solution ideas in detail. Chapter 5 defines metrics and algorithms for comparison of found solutions and the respective benchmarks. Chapter 6 summarises findings and results. Chapter 7 elaborates possible further work and improvements.

2 BACKGROUND - (*JONAS LEITNER*)

The models described below are utilised in the subsequent concepts.

2.1 Segment Anything Model

The Segment Anything Model (SAM) is a new AI model from Meta AI that was published in April 2023. It is a pre-trained model that allows for segmentation with various input prompts. SAM was trained on the largest segmentation dataset to date, which includes over 1 billion masks on 11 million images. As a result, it allows for zero-shot generalization to unfamiliar objects without requiring additional training. [3]

2.2 Grounding DINO

Grounding DINO is an open-set object detector capable of detecting arbitrary objects specified by human language inputs. The combination of grounded pre-training and the Transformer-based DINO allows for generalization to unseen objects, using human inputs such as category names. The Transformer-based architecture used by Grounding DINO is similar to that of language models, making it easier to process both image and language data. [7]

3 RELATED WORK - (*HENDRIK HAUGG*)

The field of remote sensing and image analysis has witnessed significant advancements in recent years, fueled by the transformational capabilities of deep learning techniques. This chapter delves into the existing body of literature surrounding green area segmentation on aerial images, with a specific focus on methodologies leveraging deep learning.

The study "Mapping Urban Green Spaces at the Metropolitan Level Using Very High-Resolution Satellite Imagery and Deep Learning Techniques for Semantic Segmentation" addresses the precise and comprehensive mapping of green spaces in cities. This research evaluates two deep learning model techniques for semantic segmentation of urban green spaces with the use of different convolutional neural network encoders on the U-Net architecture and very high resolution satellite imagery to obtain updated information on green spaces. The best model yielded a intersection over union (IoU) of 0.75 with an overall accuracy of 0.97, which reflects a reliable performance of the network in detecting patterns that make up the varied geometry of urban green spaces. [1]

Another study titled "A Lightweight Deep Learning Architecture for Vegetation Segmentation using UAV-captured Aerial Images" addresses the challenges associated with vegetation segmentation in urban environments, particularly utilizing unmanned aerial vehicles for data acquisition. This research explores the development of a lightweight deep learning architecture tailored for the efficient segmentation of vegetation from high-resolution aerial images. The proposed method is unique because it introduces a lightweight convolutional neural network (CNN) architecture that uses depth-wise separable convolution mechanisms to reduce the number of parameters of the model. The proposed architecture achieved an intersection over union of 82% and 71% on the NITRDrone dataset and UrbanDrone dataset, respectively, for vegetation segmentation using UAV-captured aerial images. This can be useful in mapping urban areas and agricultural lands. [18]

In summary, the integration of deep learning techniques has shown remarkable success in accurately mapping and segmenting urban green spaces. These studies show the potential for precise and comprehensive analysis, with applications ranging from urban planning to environmental monitoring.

4 CONCEPT

4.1 Preprocessing

4.1.1 Building removal - (Hendrik Haugg). In order to be able to focus on unknown data during segmentation, a method was developed to remove known data from the existing aerial images, concentrating on the removal of buildings. To obtain the data, the open source geodatabase OpenStreetMap was used, whose data can be retrieved using Overpass API. Polygons of buildings are stored in the database as simple polygons or, in the case of complex structures or e.g. houses with courtyards, as multipolygons. Since the aerial images are available as tif-files with georeference, it is easy to superimpose the polygons of the buildings in the correct position with the aerial images. With the help of the OpenCV Python library, the houses could then be "cut out" of the aerial images. An example of this function is shown in the following pictures.

(a) Original image.

(b) Image with buildings removed.

4.1.2 Contrast and saturation enhancement - (Jonas Leitner). Contrasts and colours are necessary for the human eye to recognise certain objects and their boundaries. By increasing the contrast, the difference between the lightest and darkest areas in the image is increased. This may help to improve object recognition. Increasing saturation intensifies the colours in the image. This can improve the recognition of green, for example. For this reason, two methods were developed using the OpenCV Python library to increase contrast and saturation during preprocessing.

4.1.3 Shadow reduction - (Sebastian Huber). In the execution of segmentation tasks, shadows present in aerial images could have a detrimental impact on performance. Therefore, the option of shadow reduction should be integrated as a preliminary preprocessing step to mitigate these effects. The method employed for shadow reduction is derived from an existing research paper, titled "Near real-time shadow detection and removal in aerial motion imagery application," by GF Silva [17]. The original algorithm in Silva's paper necessitates a two-step approach, involving detection followed by reduction of shadows. However, in this study, the said process has been streamlined for efficiency by combining these two steps into one single method. The code was copied form the public available Github repository ThomasWangWeiHong/Shadow-Detection-Algorithm-for-Aerial-and-Satellite-Images.

4.1.4 Image Preparation - (Sebastian Kiunke). Model-based, supervised deep learning algorithms depend on a good data basis [16]. On the one hand, this requires images to be available in sufficient quantity for accurate training and evaluation. On the other hand, a fitting resolution is necessary for the hardware to be able to process the input. Image 2a shows the original image. In Image 2b the original image is hard split into a lower resolution. To greater increase the

amount of training data and have the input to the model be present multiple times, an overlay split is used as shown in Image 2c.

(a) Original image. (b) Hard boundary split. (c) Overlap split.

A sample image from the Potsdam dataset with an absolute resolution of 6000px × 6000px is hard split into 529 images with a resolution of 256px. The same image with an addition 128px overlay split results into 2025 images in total. For data driven models image manipulation is not limited to creating overlay splits, but rotating or mirroring the images [15]. However, in this scenario of the Potsdam and own created Dataset the amount of images created by split is sufficient and too many images are not able to be processed due to lack of memory. Subchapter 4.1.1 describes the removal of buildings in .tif images. Since these store coordinates as metadata, and these mustn't be lost after split, the coordinates are updates for each image independent of the used split method allowing images to be resembled after evaluation.

4.2 Image Processing - (*Sebastian Huber*)

The first concept is the Image Processing based concept. This was created with the intention of being a baseline / reference implementation to compare the machine learning approaches against. Therefore this concept only uses normal image processing techniques to segment the areal images for green areas.

The coarse process is:

(1) Preprocess
   (a) Reduce shadows
   (b) Contrast and Saturation adjustment
   (c) Blurr
   (d) Remove houses
(2) Detect
   (a) RGB based Detection
   (b) HSV based Detection
   (c) Histogram adjustment
   (d) Binarization
(3) Postprocess
   (a) Morphologie

(b) Remove houses

As seen the process is divided up into 3 main parts 4.2.1 Preprocessing, 4.2.2 Detection and 4.2.3 Postprocess. The next sections elaborate on each step.

A major advantage of this concept is, that there is no need to reduce the size of images to process them in small, few 100 pixel sections. The larger an image is, the faster this works. Also this is generally very fast as none of the steps required to run a, potentially large neuronal network on each pixel most of the operations can even be exported to shaders and run in parallel on a GPU as only local data is required. The entire process is split up into many functions witch have one or more parameters to slightly change the behaviour of the transformation.

4.2.1 Preprocess. For prepossessing the target is to enhance the image and improve the greenness of pixels that should be green and reduction of noise.

To minimize false positive all known buildings from map data can be safely removed as were a building stands there can be no tree or lawn, except green roofs. However green roofs should not be detected and therefor helps twice. For this the module as described in 4.1.1 is used.

The shadow reduction as described in 4.1.3 is required to brighten up areas shaded by trees or buildings. As shaded areas can become very dark and than are hard to separate from (dark) gray pixels. This is further helped by the contrast and saturation adjustment as described in 4.1.2 boost the color of the pixels.

Last operations is a blur step that uses a Gaussian blur to reduce the noise created by resolution, leaves shading each other and other source. This reduces small one pixel errors.

4.2.2 Detect. The detection tries to find green areas either by examining the RGB values of a pixel of checking if the HSV values of a given pixel are part of a given range. After this the resulting image is adjusted in the Histogram to spread out the values. The last step is to make a binary image form the gray scale intermediate.

The RGB based detection used the convenient fact that the segmentation target are green areas and the RGB color space has a dedicated green axis. Therefor it simply calculates a "greenness" value for each pixel with:

$$grennness = \left\lfloor G - \frac{R+B}{2} \right\rfloor$$

This yields large values for pixels with have dominant green channel and are green, for gray and other colors this value will be small as the fraction is of similar size or larger a G alone.

However as this is still a gray scale value and not a segmentation of the image further processing is required. Fist the histogram of this gray scale image is adjusted as the calculation above mostly creates very small values. After this the image is passed to a simple binarization that checks if the value of a pixel is greater than a fixed threshold and segments the image accordingly.

The HSV base detection is much simpler it just checks for each pixel if the H, S and V components are within a set given limits and assigns the result of the test to the pixel.

4.2.3 Postprocess. The last step is to reduce the noise introduced by the binarisation step.

As the detection step in 4.2.2 works on a per pixel basis with no consideration of the surrounding pixels that lone pixels are common. To remove these lone pixels look similar to Salt and Pepper noise. To reduce / remove such noise it is common to use Morphology. As the created noise has black and white dots at the same time a Closing and an Opening operation is sequence is required to reduce both artifacts.

As the morphology operations operate on more than a 1x1 pixel basis, these can potentially undo some of the house removal from 4.2.1 Preprocessing, so the houses are removed in a last step again to avoid these basic errors.

4.3 Unet - (*Sebastian Kiunke*)

U-Net was initially released to the scientific audience by Olaf Ronneberger, Philipp Fischer and Thomas Brox in 2015 as "U-Net: Convolutional Networks for Biomedical Image Segmentation" [14] and is deeply embedded in image segmentation for tumour - [6] and nuclei - [22] segmentation.

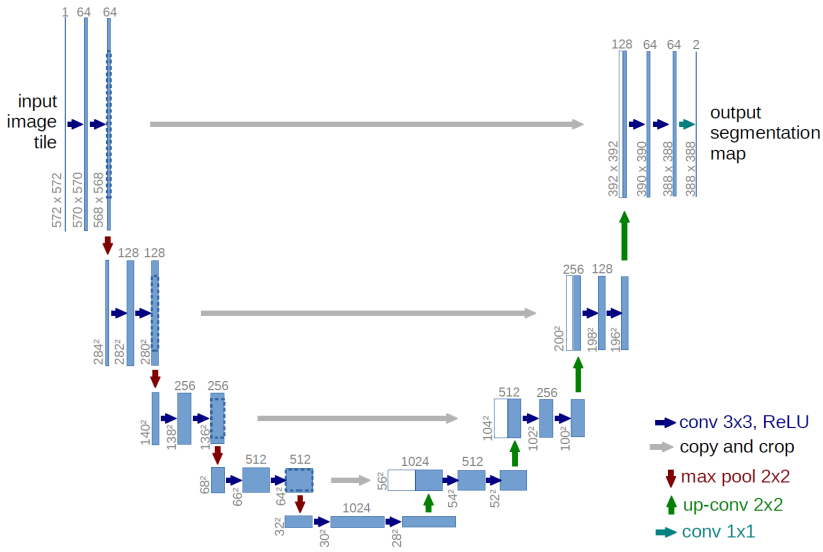

Fig. 3. The U-Net architecture. The name derives from the shape.

The deep convolutional network extends the prediction of single class labels to pixel localisation, which is required in the medical field not only to identify cancer cells, but also to localise them in an image. The architecture visible in Image 3 is based on a convolutional network and represents an encoder-decoder structure. The contracting path (left side) shrinks the image while increasing the depth of the features, and thus acts like a zoom on important features. The expansive path reconstructs an image from the detailed features. The encoder repeatedly highlights features of the image such as edges or textures with two 3×3 convolutions, followed by a ReLu to learn complex patterns. Finally, the image is reduced by 2×2 Max Pooling to focus on significant features and increase processing speed. The decoder works in reverse order by first enlarging the image to its original size and then combining features with paths. Finally, the image is refined by two more 3×3 convolutions and then by ReLu [14].

The extraction of vegetation and green areas in particular targets a similar problem with cancerous cells as not only the existence, but the localisation is required to be specified. Therefore, a model is trained on the U-Net architecture by combining bit masks of pre labeled images with aerial photographs.

Since compared to images of cancerous cells, the training data with green vegetation is available in great numbers, data augmentation like rotating or inverting images is not required and limited to an overlap split as described in Chapter 4.1.4.

The model itself is defined in TensorFlow using the Keras API. To reduce the possibility of overfitting, the training is early stopped if the training loss or validation loss, which is evaluated on images not used for training to measure performance on unseen data, does not improve within three episodes. U-Net is trained on two data sets, whereby the Potsdam data set is not used any further, as it contains strong inconsistencies in the labels of the data and some areas were labelled incorrectly or not at all as described in the Appendix B.

4.4 Segment Anything - (*Jonas Leitner*)

Another concept being explored is the segmentation of green areas using the Segment Anything Model (SAM) [3]. SAM offers three different input prompt options. The first option enables the setting of one or multiple input points on the image, with the additional possibility of setting negative points. The second option involves setting bounding boxes within which SAM will attempt to segment the object. The third option is an 'everything' approach called *AutomaticMaskGenerator*, which sets a grid of input points over the entire image.

There are two main approaches to using SAM. The first approach involves providing input points or bounded boxes as input prompts before obtaining the final result from SAM. The second approach involves letting SAM segment all it finds and then classifying or clustering the masks in a subsequent step. SAM's automatic mask generation performs well on ground-level pictures, as it was trained on, but it struggles with aerial imagery and frequently misses areas entirely (see Appendix A). Additionally, certain areas, such as multiple trees, were divided into numerous smaller masks instead of larger ones. We decided to pursue the approach with input prompts as it eliminates the need for a further classification step.

In order to increase flexibility and reduce manual labour, we have opted to utilise an additional powerful model, Grounding DINO. The model requires an input of an (image, text) pair and produces multiple pairs of object boxes with corresponding phrases after processing. The selection of object boxes can be influenced by a box threshold, where the highest similarity must exceed this threshold. The label assigned to the object box is determined by comparing the similarity value of the words in the text prompt to the text threshold. [7]

These generated object boxes can now be perfectly transferred to SAM as input prompts for segmentation. In order to simplify this process and also to obtain other extremely useful features for segmenting satellite images, we use Segment-Geospatial (samgeo), an open-source Python package [20]. The packet interface of samgeo is well-designed, which simplifies the overall segmentation process of satellite images. It uses rasterio to split Tiff images into smaller rasters, which can then be segmented. The masks can be merged back together without losing the Tiff metadata. Utilizing samgeo's direct interface for using Grounding DINO and SAM significantly simplifies building the pipeline structure for this concept.

The model created for the SAM approach can be configured with several parameters three of which are specific to Grounding DINO: a text prompt, a box threshold, and a text threshold. The text prompt contains category words of objects to be detected. The box threshold can be used to control the certainty of the model. The text threshold can be used to control the certainty of the labelling. This parameter is relevant when different categories are to be recognised. In our case, however, we are dealing with a binary classification between green and non-green areas. SAM provides several pre-trained checkpoints. The model type parameter configures SAM and offers a choice between the vit_h model, which is currently the best performing model, and the vit_l model, which has fewer parameters and is therefore slightly faster. As GroundingDINO and SAM have been found to produce better results on smaller images, we have utilised the *split_raster* function of samgeo to divide larger input images into smaller ones. The size of the tiles and the amount of overlap can be adjusted to control the rasterisation process. Another step to optimise segmentation is to

pre-process the input image beforehand. The preprocessing parameter can be used to specify a list of any preprocessing steps in the desired sequence. These steps have been presented in subsection 4.1.

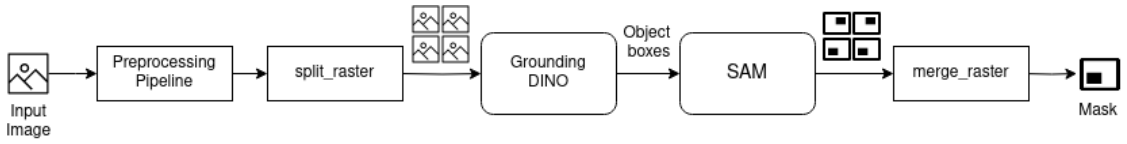

Fig. 4. Complete pipeline for segmentation with Segment Anything Model (SAM)

Figure 4 illustrates the complete pipeline process. The input image undergoes preprocessing before being split into small tiles. These tiles are then processed by Grounding DINO to predict object boxes, which are subsequently passed to SAM for segmentation. Once all tiles have been processed, the masks produced by SAM are merged to generate a predicted mask that matches the input image. One advantage of this approach is its flexibility, as it can handle various text prompts and quickly segment different categories, such as solar systems, for example.

4.5 MMSegmentation - (*Hendrik Haugg*)

In a further approach, the use of MMSegmentation was pursued. MMSegmentation is an open-source semantic segmentation toolbox based on PyTorch. It is a part of the OpenMMLab project, which is a collection of open-source machine learning tools. MMSegmentation provides a unified framework for implementing, evaluating, and benchmarking semantic segmentation methods. It supports a variety of semantic segmentation models as well as a variety of datasets. [9] First, the datasets that were most suitable for the segmentation of aerial images were selected from the large number of datasets. The following three datasets were selected:

- The **Potsdam** dataset is a collection of aerial images and corresponding semantic labels for urban areas. The images are in the form of true orthophoto and corresponding normalized digital surface models and have a ground sampling distance of 5 centimeters.The semantic labels include 6 classes: buildings, cars, low vegetation, impervious surfaces, trees, and clutter/ background. [5, 21]
- The **Vaihingen** dataset is another collection of aerial images and corresponding semantic labels for urban areas. The images are in the form of true orthophoto and corresponding normalized digital surface models and have a ground sampling distance of 9 centimeters. The semantic labels include 6 classes: buildings, cars, low vegetation, impervious surfaces, trees, and clutter/ background. [5, 21]
- The **LoveDA** dataset is a recent addition to the collection of urban scene datasets. The images are in the form of high-resolution optical satellite images, and they have a ground sampling distance of 1 meter. The semantic labels include various classes, including land cover types, man-made objects, and natural element. [23]

In order to generate results quickly, models were selected that are already pre-trained on the respective datasets. These include:

- **PSPNet** is a pyramid scene parsing network that utilizes global context information by different-region-based context aggregation. It achieves state-of-the-art performance on various datasets. [13]
- **DeepLabV3+** is a deep neural network for semantic image segmentation that combines the advantages of spatial pyramid pooling and encoder-decoder structure, achieving high performance on various datasets. [11]

- **HRNet** is a proposed network for high-resolution representation learning, which connects the high-to-low resolution convolution streams in parallel and repeatedly exchanges the information across resolutions. This results in a semantically richer and spatially more precise representation and shows superiority in human pose estimation, semantic segmentation, and object detection. [12]

In the following, the aerial images on the three pre-trained models were segmented and evaluated. The segmentation process can be divided into three sub-steps. First, the aerial images were divided into smaller sub-images in the format of the required input size of the model. Then a mask was created with the model, which shows the segmented sub-areas. Since all classes of the respective dataset are displayed in the output, the segmented elements suitable for the application had to be extracted here. The resulting mask could then be used to evaluate the model.

5 EVALUATION

5.1 Methodik - (*Sebastian Huber*)

The evaluation is based on a per pixel comparison of the calculated masks and a given ground truth mask. As the segmentation is only concerned about separating green areas like Grass, Trees and similar from the rest. Thefor the task can be seen as an binary classification problem, where for each pixel the decision if it is parat of a green area or not has to be done. This allows the use of metrics desired for binary classification. The evaluation counts the four basic cases, true positive (TP), true negative (TN), false positive (FP) or Type I error and false negative (FN) or Type II error per pixes. From these numbers and the total amount of pixels further metrics in the confusion matrix are calculated as shown in Table 1.

Name	Abbreviation	Calculation
sensitivity, recall, hit rate, or true positive rate	TPR	$TP/(TP+FN)$
miss rate or false negative rate	FNR	$FN/(TN+TP)$
specificity, selectivity or true negative rate	TNR	$TN/(TN+FP)$
fall-out or false positive rate	FPR	$FP/(FN+TP)$
precision or positive predictive value	PPV	$TP/(TP+FP)$
false discovery rate	FDR	$FP/(FP+TP)$
negative predictive value	NPV	$TN/(TN+FN)$
false omission rate	FOR	$FN/(FN+TN)$
intersection over union	IoU	$(TP)/(TP+FP+FN)$
accuracy	ACC	$(TP+TN)/(TP+TN+FP+FN)$
F1 score	F1	$2*PPV*TPR/(PPV+TPR)$

Table 1. Derived binary classification metrics with formulas

The implementation focuses on a reusable and versatile interface with a fast implementation. To be used easily in different contexts it supports 4 types of inputs:

- a pair of a segmented image and an expected mask as ndarrays in memeory
- list of pairs of segmented images and expected masks as ndarrays in memeory
- a pair of paths to a segmented image and en expected mask
- list of pairs of paths to segmented images and expected masks

When multiple images are evaluated, the sum of TP, TN, FP and FN for all pairs are calculated. The result of the evaluation is a object of the dataclass Evaluation containing all metrics from Table 1, TP, TN, FP and FN. This allows for proper IDE integration and auto completion when working with the evaluation.

To improve performance it is taken advantage of the fact that both images are binary images. The values a prepared to specific numbers such that form their sum and an subsequent bit-wise and operation with one set bit the case can be recovered. This implementation uses:

- expected mask: 0 for False, 54 for True
- segmented image: 0 for False, 99 for True

This allows the addition of the values to be unique in bits 4 for TP, 7 for FP, 3 for FN and 0 everywhere for TN. This can be done efficiently with the the addWeighted methode of open cv. The counting of the cases is than performed with bitwise_and with according constant followed by count_nonzero of numpy. For the last case of TN it is just pixel count minus count_nonzero on the result of the weighted addition.

5.2 Data - (*Jonas Leitner*)

Initially, we were provided with a labelled dataset of Potsdam [2]. However, we soon discovered errors in the labelling. As a result, we decided to manually label five different digital orthophotos (DOPs) of Munich. These aerial photographs are based on the Bavarian aerial survey and are provided by the Bayrische Vermessungsverwaltung [19]. The data is in DOP40 format, indicating a ground pixel size of 40cm. The available images for selection are 1km x 1km tiles. We have chosen five different tiles for our own data set. The selected tiles include urban areas with varying degrees of vegetation. One tile contains the English Garden and part of the Isar, but we have also added a tile with a large proportion of woodland. Table 1 provides an overview of the images, and our self-created dataset is available inside our Github repository.

Image	Location	Characteristics
32688_5332	Untersendling	urban area with lots of green space
32690_5335	Central Station	urban area with some green space
32691_5334	Old Town	city centre with little green space
32692_5335	Lehel	urban area with city park and river
32692_5347	Oberschleißheim	woodland with continuous road

Table 2. Breakdown of the images with their location and location characteristics.

By labelling our own data and carefully selecting it, we can improve the quality of our model evaluations and make them more meaningful than would have been possible before.

5.3 Results

5.3.1 Image Processing - (Sebastian Huber). As noted in 4.2 Image Processing the implementation provides many parameters to tune the performance. There up to 40 different parameters to select, with some categorical, continues or, bool. As this optimisation is very complicated due to the number of parameters, a Genetic Algorithm is used to find the best parameters possible. In this specific case the population is 200 with a mutation chance for each parameter of is about 1 over number of parameters. The population less then the median is replaced each time. New members are

combinations of the current best parameters. A mutation is just a uniform random selection of the given parameter. The goal of the process was to maximize ACC or IoU scores.

After letting this setup run for 30 generations the best model has a F1 score of 0.9407. This first run showed that the entire population tended to preferred the RGB based detection, with very slow convergence. Also the shadow reduction turned out to be very slow. Therefor further runs removed all parameters tied to HSV detection and shadow reduction. In addition the shadow reduction is now performed once before the optimization starts with default parameters as these looked good.

After running the optimization over night with reduced paramerters and speed up the best configuration reached an ACC over all training data of 93% and the IoU score of 85%.

When looking at the individual training images as shown in table 3

Image	IoU	F1 score	accuracy	precision	recall	TPR	TNR	FPR	FNR
32688_5332	0.784	0.879	90.575%	0.959	0.812	0.812	0.974	0.026	0.188
32690_5335	0.723	0.839	94.734%	0.936	0.761	0.761	0.989	0.011	0.239
32691_5334	0.696	0.821	95.851%	0.866	0.780	0.780	0.983	0.017	0.220
32692_5335	0.838	0.912	92.698%	0.904	0.920	0.920	0.932	0.068	0.080
32692_5347	0.927	0.962	92.841%	0.996	0.930	0.930	0.815	0.185	0.070
Overall	0.849	0.918	93.339%	0.958	0.882	0.882	0.971	0.029	0.118

Table 3. Results for the best parameters per image for F1 Optimisation for image processing

It can be seen, that the accuracy for any given input image is always above 90%. The FPR is typically significantly smaller than FNR. This points to the fact that the model is conservative in labeling green areas. The only exception to this is the last image, with represent the forest case. This also means if it labels something as green this is very likely to be true. After visual inspection a significant portion of the FNR comes from self shadowing of trees or brown lawns.These should be completely detected, however have many holes inside of them where self shaded areas are. Some of these missing detection's can be seen in the images as very small green areas and green street side banks. One of the largest miss identifications is the Isar river in the bottom right of the image 32692_5335. As FPR is generally low the river and a street in 32690_5335 not many miss identifications are done.

As an additional test the parameters were also optimized for IoU with the results as shown in Table 4

Image	IoU	F1 score	accuracy	precision	recall	TPR	TNR	FPR	FNR
32688_5332	0.779	0.876	90.408%	0.964	0.803	0.803	0.978	0.022	0.197
32690_5335	0.718	0.836	94.678%	0.947	0.748	0.748	0.991	0.009	0.252
32691_5334	0.699	0.823	95.937%	0.877	0.775	0.775	0.985	0.015	0.225
32692_5335	0.838	0.912	92.722%	0.907	0.917	0.917	0.935	0.065	0.083
32692_5347	0.932	0.965	93.256%	0.996	0.935	0.935	0.815	0.185	0.065
Overall	0.849	0.919	93.400%	0.961	0.880	0.880	0.974	0.026	0.120

Table 4. Results for the best parameters per image for IuO Optimisation for image processing

The results for IoU are almost identically to the F1 based approach. This is as expected because the 2 Metrics are calculated in a very similar manner where only the wight of TP samples differs as shown in the Table 1.

5.3.2 U-Net — (Sebastian Kiunke). The U-Net model is trained on the self created dataset described in Table 2. By using overlay split in the preprocessing step the five images are split into a total of 33205 images as described in Chapter 4.1.4. The results shown in Table 5 vastly differ from image to image, especially in accuracy and precision. One possible reason for this is the lack of versatility in the images that even the overlay split does not improve. Another reason is shown in Image 5 whereby the truth mask differs from the RGB image showing black spots meaning no vegetation when in reality there is.

Fig. 5. The bit mask (right) showing black spots, although the RGB image expects them to be white

Appendix B displays graphics of the training and validation loss and accuracy which indicate good training results.

Image	IoU	F1 score	accuracy	precision	recall	TPR	TNR	FPR	FNR
32688_5332	0.281	0.438	67.297%	0.774	0.306	0.306	0.936	0.064	0.694
32690_5335	0.224	0.366	86.357%	0.249	0.686	0.686	0.874	0.126	0.314
32691_5334	0.178	0.303	96.365%	0.711	0.192	0.192	0.997	0.003	0.808
32692_5335	0.203	0.337	83.382%	0.368	0.311	0.311	0.916	0.084	0.689
32692_5347	0.611	0.759	67.747%	0.950	0.631	0.631	0.727	0.273	0.015
Overall	0.470	0.639	60.746%	0.762	0.551	0.551	0.705	0.295	0.449

Table 5. Results U-Net

5.3.3 Segment Anything - (Jonas Leitner). SAM and Grounding Dino are pre-trained models that can generalise even for unknown objects. No weights had to be optimised through time-consuming training. However, adaptation is limited, and the model can only be optimised using hyperparameters or by modifying the input itself.

The initial observation was that batch segmentation produced considerably better results than performing the segmentation on the entire 2500x2500 pixel image. Furthermore, while the vit_h checkpoint of SAM is larger and more computationally intensive, it yields significantly better results. The optimization of the *text prompt* was determined through trial and error, as a systematic approach was not feasible. However, the hint in the official Grounding DINO repository to separate the categories with a period helped. The final text prompt is 'tree . lawn . grass'. In order to optimise the other hyperparameters of the model, the values were systematically tested by first selecting large oscillation step sizes and then smaller ones as the results improved. It is important to note that we initially tried to set all hyperparameters reasonably by random trial and error, and then optimised one until the best result was achieved.

This approach led to a *box_threshold* of 0.25 for Grounding DINO and a batch segmentation with tiles rasterised at 256x256 pixels and an overlap of 64.

The image pre-processing steps (as presented in subsection 4.1) were individually tested and in various combinations. However, only the removal of buildings was found to improve the results. Shadow reduction, contrast enhancement, and saturation enhancement did not produce any improvement. Despite this, parts of houses were still marked in the result masks. Therefore, in the final version, the houses are cut out after SAM predicts a mask. The results are presented in Table 6.

Image	IoU	F1 score	accuracy	precision	recall	TPR	TNR	FPR	FNR
32688_5332	0.626	0.770	82.286%	0.851	0.704	0.704	0.910	0.090	0.296
32690_5335	0.691	0.817	93.797%	0.875	0.767	0.767	0.976	0.024	0.233
32691_5334	0.620	0.766	95.195%	0.853	0.695	0.695	0.985	0.015	0.305
32692_5335	0.790	0.883	90.417%	0.881	0.884	0.884	0.918	0.082	0.116
32692_5347	0.980	0.990	98.023%	0.995	0.985	0.985	0.727	0.273	0.015
Overall	0.820	0.901	91.944%	0.875	0.930	0.930	0.913	0.087	0.070

Table 6. Results overview of SAM predictions per image

The overall accuracy of 91.94% seems decent. However, it should not be overemphasised as the distribution of green and non-green areas is uneven. However, the F1 score of 0.9, which is more appropriate for unevenly distributed classes, demonstrates that the results are decent. Another important metric, particularly for object recognition, is the Intersection over Union (IoU). It considers the overlap between the prediction and the actual values. The overall value is 0.82. However, this is significantly increased by the last image, which represents the forest. In the first three images, where there is less green area, the value is more in the range of 0.6 to 0.7. The recall and precision values indicate that the model tends to classify more in order to avoid overlooking any green areas (higher recall), but accepts the risk of misclassifying areas.

To facilitate identification of incorrectly classified values, a difference overlay was created for the five images. False positives are highlighted in red and false negatives in blue. The masks are available for viewing in our repository under segment-geospatial/results. In all overlays, it is evident that the most common false positives are the shadows of trees, which are frequently classified as green areas. Figure 32692_5335 highlights SAM's difficulty in recognising water, often misclassifying it as a green area. Additionally, SAM tends to segment areas that are not part of the green area but are enclosed within it. In contrast, it can be challenging to determine why SAM fails to classify certain regions. On one hand, a narrow blue border is often visible next to green areas where the green area is partially painted over in the true mask. On the other hand, artefact-like cuts caused by batch segmentation are also often noticeable. In one tile, an area may still be recognised as green while in another it is not. Unfortunately, it is difficult to identify the exact reason why certain areas are overlooked. If a more effective method for reducing shadows can be found, the prediction could be significantly improved. The current method, unfortunately, had no positive effect on the result.

5.3.4 MMSegmentation - (Hendrik Haugg). The MMSegmentation toolbox provides several models, partly pre-trained on specific datasets. In order to select the dataset that best fits the use case, the datasets were evaluated using a model trained on them. The same model was used for all datasets, in this case DeepLabV3+. All results for the required green areas were extracted using the specified configuration of the dataset with the predefined classes. All images have been

split into smaller parts to achieve the required input size, in this case 512x512px for all models. The results of the comparison can be seen in Table 7. Based on the results, the following work was continued with the LoveDA dataset[10].

Dataset	IoU	F1 score	accuracy
LoveDa	0.558	0.716	80.540%
Potsdam	0.171	0.291	41.754%
Vaihingen	0.276	0.432	30.716%

Table 7. MMSeg Dataset Comparison

A total of 3 different models were compared, *'DeepLabV3+'*, *'PSPNet'* and *'HRNet'*. When comparing the different models, pre-trained on the same dataset, the overall results are very similar, which indicates that the performance of the models is approximately the same. The results shown in Table 8 show the differences for various input images. In this case, the results from a segmentation with *'HRNet'* are shown. It can be seen that the segmentation delivers a poor performance for the input of images with small, widely distributed green areas. However, the results are acceptable for input images that show almost only green areas. The prediction is comparatively fast and takes approx. 1.2 minutes with the sample data for all 5 images.

Image	IoU	F1 score	accuracy	precision	recall	TPR	TNR	FPR	FNR
32688_5332	0.116	0.208	61.741%	0.971	0.116	0.116	0.997	0.003	0.884
32690_5335	0.127	0.225	82.872%	0.942	0.128	0.128	0.998	0.002	0.872
32691_5334	0.069	0.129	90.654%	0.949	0.069	0.069	1.000	0.000	0.931
32692_5335	0.428	0.599	70.497%	0.972	0.433	0.433	0.987	0.013	0.567
32692_5347	0.978	0.989	97.857%	0.994	0.984	0.984	0.592	0.408	0.016
Overall	0.558	0.716	80.525%	0.988	0.562	0.562	0.995	0.005	0.438

Table 8. Results MMSeg HRNet - LoveDa

5.4 Comparison - (*Jonas Leitner*)

When comparing the four approaches, it is evident that two of them outperform the other two. Table 9 provides an overview of the final metrics for each approach. Surprisingly, the pure image processing approach, which was intended as a baseline, performs the best, followed closely by the Segment Anything approach. The self-trained U-Net and the MMSegmentation approach, which was pre-trained using a different dataset, yield slightly lower results.

Concept	IoU	F1 score	accuracy	precision	recall
Image Processing	0.849	0.919	93.400%	0.961	0.880
Segment Anything	0.820	0.901	91.944%	0.875	0.930
MMSegmentation	0.558	0.716	80.525%	0.988	0.562
U-Net	0.470	0.639	60.746%	0.762	0.551

Table 9. Result comparison of different models

It is noteworthy that precision and recall have opposite values when comparing the Image Processing approach and the Segment Anything approach. The Image Processing approach is more conservative, but when it identifies green areas, it is highly confident that they are indeed green. This is in contrast to the Segment Anything approach, which

classifies slightly more areas as green and therefore has better coverage, this approach sacrifices precision. Furthermore, it is worth noting that the U-Net approach performs similarly well to MMSegmentation, despite being trained on only 5 self-labelled data, while MMSegmentation was trained on a large external dataset. Currently, comparison between the two approaches is limited, so it would be interesting to see how they perform when trained on more self-labelled data containing more variation. Another important point to note is that the Segment Anything approach needs 33.6 minutes to segment our complete data running on a graphics card to achieve a similar performance to the Image Processing approach, which only takes a second. This makes Segment Anything significantly more computationally intensive.

6 SUMMARY - (*JONAS LEITNER*)

Four different approaches were tested to segment green areas. After careful evaluation, it was determined that the Image Processing approach, which utilizes green detection, and the Segment Anything approach, which utilizes two state-of-the-art pre-trained models, were the most effective. The importance of well-labelled data was recognized, as we experienced the consequences of poorly labelled data. Therefore, we created our own labelled dataset using selected aerial images of Munich. It is important to note that in order for the U-Net and MMSegmentation models to make more accurate predictions, a larger amount of labelled data is required for specific training. However, the reliability of our model evaluation has significantly improved due to our labelling.

It is evident that two approaches perform similarly in recognising green areas. However, the question arises as to whether a deep learning approach, which is significantly more computationally intensive than pure green value detection, should be chosen for the binary recognition of green areas. This raises the fundamental question of where the approach should continue to develop. If the goal is solely to detect green areas, the image processing approach is likely the better option. This method can be further optimized through pre- and post-processing. However, if a more versatile solution is needed that can identify other elements in the future, the Segment Anything approach with text prompts may be worth considering.

7 FUTURE WORK — (*SEBASTIAN KIUNKE*)

In this paper, various approaches are described and evaluated, showing great performance within the task of vegetation detection on aerial photographs. The models and algorithms accuracy is greatly increased, when objectives are removed from the image before running the detection. As describes in Chapter 4.1.1 this is already achieved by removing buildings. However, this preprocessing step is extendible by removing streets and other known objects that might cause false detection. For both SAM and the image preprocessing, the shadow reduction is the limiting factor with bad results compared to the time they take to process. Therefore, a better shadow reduction is expected to greatly improve the results. As MMSEG and U-Net are based on model, the data used for training is the key to great segmentation in images. Although our own created data set, greatly increased the evaluation metrics, a more versatile and larger set is expected to produce even better results.

ACKNOWLEDGMENTS

Thanks to DeepL for assisting with the translation.

REFERENCES

[1] Roberto E. Huerta, Fabiola D. Yépez, Diego F. Lozano-García, Víctor H. Guerra Cobián, Adrián L. Ferriño Fierro, Héctor de León Gómez, Ricardo A. Cavazos González, and Adriana Vargas-Martínez. 2021. Mapping Urban Green Spaces at the Metropolitan Level Using Very High Resolution

Satellite Imagery and Deep Learning Techniques for Semantic Segmentation. *Remote Sensing* (2021). https://www.mdpi.com/2072-4292/13/11/2031

[2] ISPRS. 2022. 2D Semantic Labeling Contest - Potsdam. https://www.isprs.org/education/benchmarks/UrbanSemLab/2d-sem-label-potsdam.aspx

[3] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. 2023. Segment Anything. *arXiv:2304.02643* (2023).

[4] Sylvia Kizlauskas. [n. d.]. Bodennutzungsflächen in München. https://stadt.muenchen.de/dam/jcr:ac84bba0-a762-4935-956d-4040dbf905e7/mb110104.pdf. Accessed: 05-01-2024.

[5] Minglong Li, Lianlei Shan, Xiaobin Li, Yang Bai, Dengji Zhou, Weiqiang Wang, Ke Lv, Bin Luo, and Si-Bao Chen. 2023. Global-Local Attention Network for Semantic Segmentation in Aerial Images. *ResearchGate* (2023).

[6] Xiaomeng Li, Hao Chen, Xiaojuan Qi, Qi Dou, Chi-Wing Fu, and Pheng-Ann Heng. 2018. H-DenseUNet: hybrid densely connected UNet for liver and tumor segmentation from CT volumes. *IEEE transactions on medical imaging* 37, 12 (2018), 2663–2674.

[7] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. 2023. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499* (2023).

[8] Statistisches Amt München. [n. d.]. Bevölkerung 1970 - 2022 nach Geschlecht und Staatsangehörigkeit. https://stadt.muenchen.de/dam/jcr:c45700b2-e2c3-4d66-99fa-55e4c77ab3ee/jt230101.pdf. Accessed: 05-01-2024.

[9] OpenMMLab. 2021. Overview. https://mmsegmentation.readthedocs.io/en/latest/overview.html

[10] OpenMMLab. 2021. Prepare datasets. https://mmsegmentation.readthedocs.io/en/latest/user_guides/2_dataset_prepare.html

[11] OpenMMLab. 2023. DeepLabV3+. https://github.com/openmmlab/mmsegmentation/tree/main/configs/deeplabv3plus

[12] OpenMMLab. 2023. HRNet. https://github.com/open-mmlab/mmsegmentation/tree/main/configs/hrnet

[13] OpenMMLab. 2023. PSPNet. https://github.com/open-mmlab/mmsegmentation/tree/main/configs/pspnet

[14] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*. Springer, 234–241.

[15] Massimo Salvi, U Rajendra Acharya, Filippo Molinari, and Kristen M Meiburger. 2021. The impact of pre-and post-image processing techniques on deep learning frameworks: A comprehensive review for digital pathology image analysis. *Computers in Biology and Medicine* 128 (2021), 104129.

[16] Iqbal H Sarker. 2021. Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions. *SN Computer Science* 2, 6 (2021), 420.

[17] Guilherme F. Silva, Grace B. Carneiro, Ricardo Doth, Leonardo A. Amaral, and Dario F.G. de Azevedo. 2018. Near real-time shadow detection and removal in aerial motion imagery application. *ISPRS Journal of Photogrammetry and Remote Sensing* 140 (2018), 104–121. https://doi.org/10.1016/j.isprsjprs.2017.11.005 Geospatial Computer Vision.

[18] Pankaj Kumar Sa Tanmay Kumar Behera, Sambit Bakshi. 2023. A Lightweight Deep Learning Architecture for Vegetation Segmentation using UAV-captured Aerial Images. *Elsevier Inc.* (2023). https://www.sciencedirect.com/science/article/abs/pii/S221053792200172X

[19] Bayerische Vermessungsverwaltung. 2023. OpenData - Digitales Orthophoto 40cm (DOP40). https://geodaten.bayern.de/opengeodata/OpenDataDetail.html?pn=dop40

[20] Qiusheng Wu and Lucas Prado Osco. 2023. samgeo: A Python package for segmenting geospatial data with the Segment Anything Model (SAM). *Journal of Open Source Software* 8, 89 (2023), 5663. https://doi.org/10.21105/joss.05663

[21] Siyuan Xing, Qiulei Dong, and Zhanyi Hu. 2023. SCE-Net: Self- and Cross-Enhancement Network for Single-View Height Estimation and Semantic Segmentation. *ResearchGate* (2023).

[22] Zitao Zeng, Weihao Xie, Yunzhe Zhang, and Yao Lu. 2019. RIC-Unet: An improved neural network based on Unet for nuclei segmentation in histology images. *Ieee Access* 7 (2019), 21420–21428.

[23] Zheng Zhou, Change Zheng, Xiaodong Liu, Ye Tian, Xiaoyi Chen, Xuexue Chen, and Zixun Dong. 2023. A Dynamic Effective Class Balanced Approach for Remote Sensing Imagery Semantic Segmentation of Imbalanced Data. *ResearchGate* (2023).

A SAM - AUTOMATIC MASK GENERATION

The first attempt was to use SAM's automatic mask generation. captured 237 masks from the image. By adjusting parameters, such as reducing thresholds and increasing point density, 2378 masks were identified. However, some areas still remain unsegmented. Table 10 displays the modified configuration parameters, and Figure 1 presents the comparison images with the masks.

Parameters	Default Config	Modified Config
points_per_side	32	100
pred_iou_thresh	0.88	0.86
stability_score_thresh	0.95	0.92

Table 10. This table displays the modified parameters for the SAM AutomaticMaskGenerator's Modified Config in comparison to the Default Config parameters.

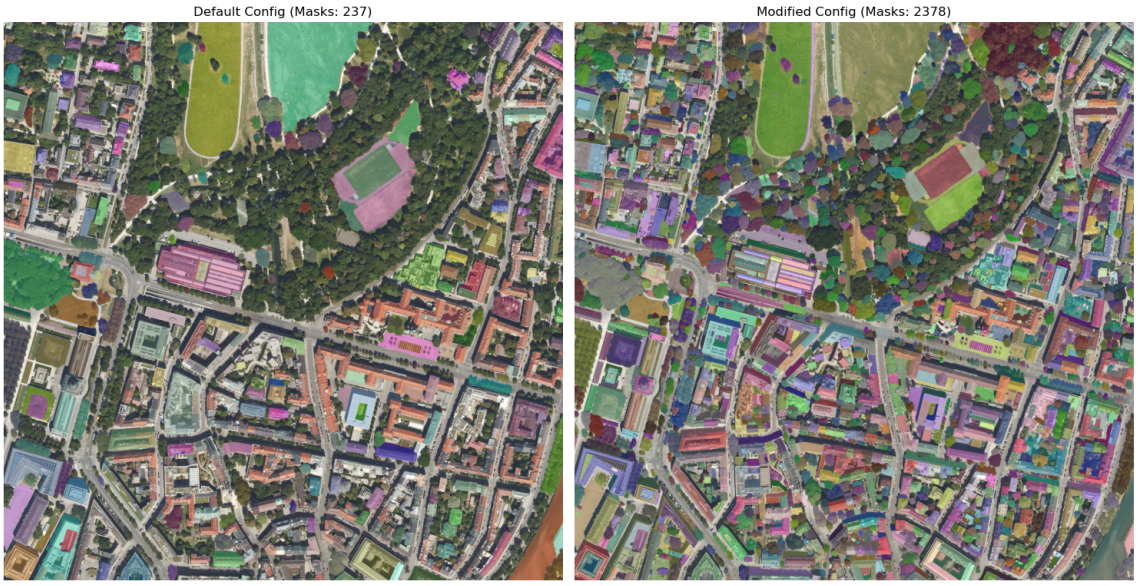

Fig. 6. Comparison of SAM AutomaticMaskGenerator with Default Config and a Modified Config

B U-NET DATASET DATA AND TRAINING RESULTS

Due to incorrect masks in the test data of the Potsdam dataset as shown in 7b and 7a, the Potsdam dataset is not used. The middle area marked black must be white, as it is vegetation and the right part should be white as it is not vegetation.

(a) RGB image. (b) Bitmask.

Fig. 7. Potsdam sample data with wrong labelled image mask.

The training and validation loss shown in Image 8 and accuracy in Image 9 portray good training results.

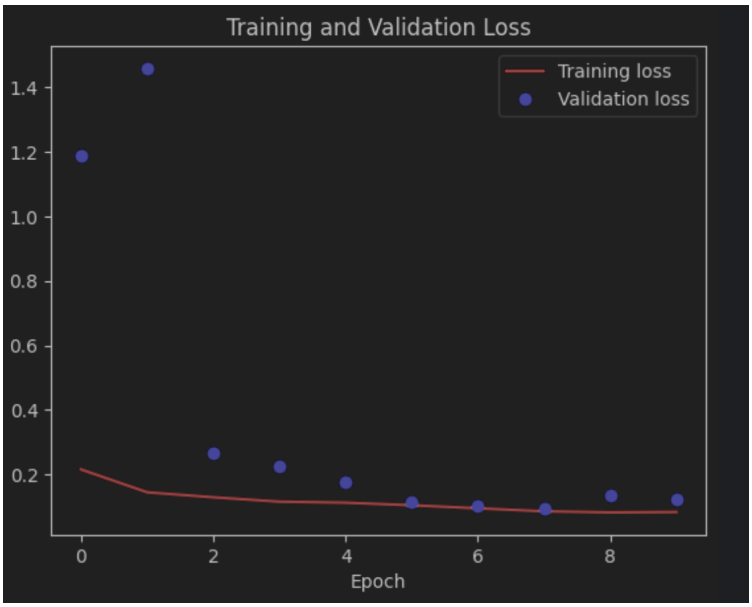

Fig. 8. The training and validation loss over the trained epochs.

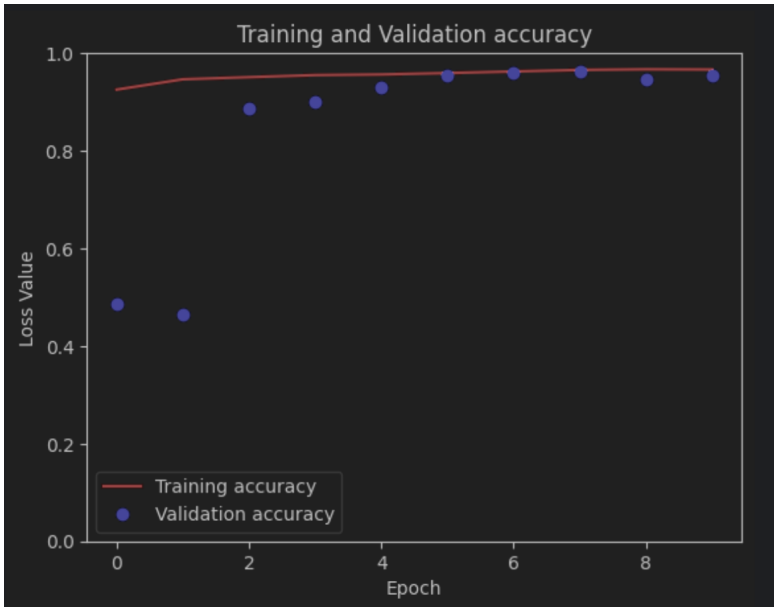

Fig. 9. The training and validation accuracy over the trained epochs.