




Article

Generating a Dataset for Semantic Segmentation of Vine Trunks in Vineyards Using Semi-Supervised Learning and Object Detection

Petar Slaviček ¹, Ivan Hrabar ² and Zdenko Kovačić ^{1,*}

¹ Faculty of Electrical Engineering and Computing, University of Zagreb, Unska 3, 10000 Zagreb, Croatia; petar.slavicek@gmail.com

² Autegra d.o.o., Vladimira Nazora 5, 43280 Garešnica, Croatia; ivan.hrabar@gmail.com

* Correspondence: zdenko.kovacic@fer.hr

Abstract: This article describes an experimentally tested approach using semi-supervised learning for generating new datasets for semantic segmentation of vine trunks with very little human-annotated data, resulting in significant savings in time and resources. The creation of such datasets is a crucial step towards the development of autonomous robots for vineyard maintenance. In order for a mobile robot platform to perform a vineyard maintenance task, such as suckering, a semantically segmented view of the vine trunks is required. The robot must recognize the shape and position of the vine trunks and adapt its movements and actions accordingly. Starting with vine trunk recognition and ending with semi-supervised training for semantic segmentation, we have shown that the need for human annotation, which is usually a time-consuming and expensive process, can be significantly reduced if a dataset for object (vine trunk) detection is available. In this study, we generated about 35,000 images with semantic segmentation of vine trunks using only 300 images annotated by a human. This method eliminates about 99% of the time that would be required to manually annotate the entire dataset. Based on the evaluated dataset, we compared different semantic segmentation model architectures to determine the most suitable one for applications with mobile robots. A balance between accuracy, speed, and memory requirements was determined. The model with the best balance achieved a validation accuracy of 81% and a processing time of only 5 ms. The results of this work, obtained during experiments in a vineyard on karst, show the potential of intelligent annotation of data, reducing the time required for labeling and thus paving the way for further innovations in machine learning.

Keywords: semantic segmentation; object detection; semi-supervised learning; intelligent data annotation; vine trunk segmentation; agriculture robotics



Citation: Slaviček, P.; Hrabar, I.; Kovačić, Z. Generating a Dataset for Semantic Segmentation of Vine Trunks in Vineyards Using Semi-Supervised Learning and Object Detection. *Robotics* **2024**, *13*, 20. <https://doi.org/10.3390/robotics13020020>

Academic Editor: Giulio Reina

Received: 6 December 2023

Revised: 10 January 2024

Accepted: 19 January 2024

Published: 23 January 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

In today's world, rapid technological progress and automation offer the opportunity to introduce innovations in many industries. Agriculture, as one of the most important sectors of the economy, is no exception. Considering that vineyards are sensitive to many environmental factors and require constant care and attention, it is even more important to improve and simplify the traditional methods of viticulture. The HEKTOR project (hektor.fer.hr (accessed on 18 January 2024)). Ref. [1] aims to develop an autonomous robot for the care of vineyards, equipped with tools that allow it to perform activities such as suckering, spraying, and monitoring. This robot is designed to make work easier in small and medium-sized vineyards in order to solve the problem of tedious and often dangerous manual labor [2]. Extensive work has already been carried out in the field of object detection in the vineyard for robot localization, autonomous task execution [3], and navigation [4]. Systems for efficient spraying [5] and the necessary hardware for vine suckering [6] have been developed.

One of the most important aspects for the success of this type of robot is its ability to understand its environment and interact with it correctly. This includes accurately recognizing the vine, assessing its state, identifying its shape and position, and adapting the movements and actions that the robot needs to perform. To achieve this understanding of the environment, advanced computer vision and machine learning methods must be used, such as object detection and semantic segmentation. In the field of machine learning, both object detection and semantic segmentation play an important role in a variety of applications; e.g., medical diagnostics [7,8], autonomous driving [9], and other industries [10,11]. The goal of object detection is to automatically label and classify each object in an image with a bounding box, while the goal of semantic segmentation is to automatically label each pixel in an image with the corresponding semantic class so that computers can automatically and accurately understand the content of images. Recent advances in semantic segmentation have largely been achieved by applying deep neural networks via supervised learning, as evidenced by several studies [12–20]. This also applies to object detection [21–30].

One of the biggest challenges in machine learning, especially in the context of semantic segmentation, is the lack of large, fully annotated datasets. Annotating images at the pixel level requires significant time and human effort, so there is often only a limited amount of data available for learning. Using a small or limited amount of data can lead to overfitting and a weaker generalization effect for the algorithms, especially when dealing with diverse and complex real-world scenes that are outside the range of the training data. For this reason, these advances are highly dependent on large, fully annotated datasets [31–34]. To overcome this problem, several labeling-efficient learning techniques have been proposed. These include semi-supervised learning [35–45], unsupervised learning [46–48], weakly supervised learning [49–56], and the adaptation of synthetic domains to real domains [57–62], all of which focus on semantic segmentation.

In this article, we explore an approach that leverages existing vine trunk detection data [63] to generate a dataset that can be used for semantic segmentation of vines with a minimum of human annotation. There are several approaches that attempt to label unlabeled data with minimal human effort. A team at Meta AI Research has presented its Segment Anything system [64], which creates segmentation masks using image embeddings and prompts. While this approach is very powerful [65,66], our initial tests showed only mediocre results. We believe that this was due to the low resolution of our inputs. Further investigation and experimentation are needed, but this is beyond the scope of this study. A team at the Rochester Institute of Technology has developed an approach that uses semi-supervised learning together with active learning [67] to transfer labels from a smaller dataset to a larger one. Another approach is to use generative models (GANs) [68] to generate fake examples that lead to confusion regarding the predictions of labeled and unlabeled examples. This encourages the model to generalize as it has to learn to distinguish between fake and real objects. Semantic segmentation with semi-supervised learning is typically treated with a student–teacher architecture [44,69] and in recent years in combination with contrastive learning [70,71]. The approach used in this work combines semi-supervised learning and contrastive learning with error localization networks [45], which focus the model and allow it to quickly fix common errors. This method is compared with other learning methods for semantic segmentation in the context of mobile robotic systems in Table 1. First, cherry-pick objects to segment refers to the ability of our method to select the objects to be segmented. This saves processing time as we do not segment objects that we are not interested in. Second, recognizing individual vines is important because semantic segmentation methods output a binary image containing all vine trunks, which means that an additional step is required to separate individual instances of vine trunks. Third, class interpretability is a problem that arises with unsupervised segmentation methods. Since we do not provide ground truth data to the training method, there is no reason why the method should not learn to segment, for example, the shadow of the vine trunk and the vine trunk. There is no way to accurately interpret the segmented class.

Finally, a small memory footprint is an important requirement for a mobile robotic system. While it is possible to have a supervised semantic segmentation model with low memory requirements, the input resolution would have to be reduced so much that the performance would degrade significantly. Since our method only segments low-resolution thumbnail images of recognized objects, the segmentation models can be extremely small. We also compared existing semantic segmentation methods to determine which is best suited for use with a mobile robot. The source code developed as part of this research is available at Supplementary Video S1.

Table 1. Comparison of the available training methods for semantic segmentation and our proposed method in the context of mobile robotic systems. The green tick shows that a method includes a certain functionality or feature and the red cross shows that it does not.

Method	Cherry-Pick Objects to Segment	Little Human Annotation Needed	Detection of Individual Vines	Class Interpretability	Low Memory Footprint
Supervised semantic segmentation	✗	✗	✗	✓	✗
Unsupervised semantic segmentation	✗	✓	✗	✗	✗
Semi-supervised semantic segmentation	✗	✓	✗	✓	✗
Semi-supervised semantic segmentation and YOLO object detection	✓	✓	✓	✓	✓

2. Methodology

2.1. Training YOLOv5 Object Detection Model

The main resource for this research was the publicly available vine trunk object-detection dataset VineSet [63], which contains about 22,000 annotated color images of six Portuguese vineyards, including about 2000 annotated thermal images. We could not use the thermal images in this context and therefore discarded them immediately. The annotations of the VineSet dataset were created in the Pascal VOC format. Since we were using the YOLOv5 architecture, we had to use the YOLO annotation format for object detection. For this reason, a program was developed to convert Pascal VOC .xml files to YOLO .txt files so that we could use them with the publicly available YOLO project. A closer look at the dataset shows that VineSet has several problems. A large part of the dataset is intended for the recognition of grape clusters and does not contain any vine trunks at all. An example of this can be found in Figure 1a. While it is useful to have negative examples (i.e., examples that do not contain the target object in the image), too many such examples (as in this case) can negatively affect the accuracy of the model. Approximately 30% of the dataset was programmatically filtered out because it did not contain vine trunks. The proportion of negative examples was reduced to 5% of the total dataset for the reasons already mentioned. In addition, the dataset contains a large number of unlabeled vine trunks in the background, as can be seen in Figure 1b,c. This is extremely disadvantageous for a model of this type, as it will learn that the examples of vine trunks are not really vine trunks. Furthermore, there are bounding boxes that only cover a very small segment of a vine trunk, as can be seen in Figure 1d. Such annotations do not represent the kind of objects we want to recognize and would significantly affect the training results. Unfortunately, there is no convenient and effective algorithmic method to filter out such examples, so the entire dataset was filtered manually. In total, about 65% of the images were discarded for these reasons. About 8000 images remained from the VineSet. In addition to the data from the VineSet dataset, we used all other manually annotated data collected as part of the HEKTOR project, which included about 40 images of vineyards in Zelina, 33 images of vineyards in Jazbina, and 250 images of vineyards on Korčula. Figure 2a–c show examples of data from VineSet, and Figure 2d shows an example of data from the island of Korčula.

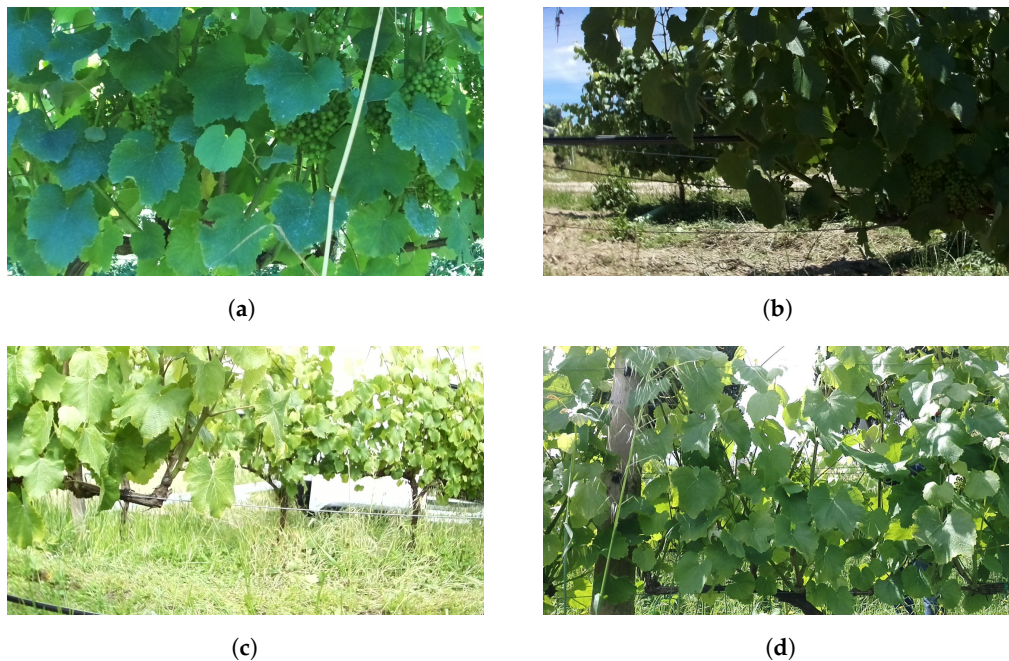


Figure 1. Examples from VineSet that negatively affect object detection training. The image in (a) is intended for grape cluster detection and does not contain vine trunks. The images in (b,c) contain unannotated vines in the background. The image in (d) contains a very small piece of vine that is annotated.

For the training, the medium YOLOv5 model was used, which was previously trained with the COCO dataset. Although there are newer YOLO architectures [29,72], our robot stack was built with YOLOv5 and the performance proved adequate, so we saw no need to upgrade to newer architectures. The model was trained for 50 epochs. To improve the generalization of the model, recommended augmentations were added. Some of these augmentations were adding Gaussian noise to the image, blurring the image, converting the image to a monochrome image, rotating the image, flipping the image on the horizontal axis, cropping the image at different points, and creating mosaics from multiple images. All these functions are already implemented in the YOLOv5 project and the hyperparameters that determine how often these augmentations take place are default values in the YOLOv5 project. The training time of 50 epochs was also based on the default training time specified in the YOLOv5 project. The dataset was split into 80% for training and 20% for validation. All images were automatically scaled to a resolution of 640×480 as that is a default input size for our medium YOLOv5 model.

The graphs in Figure 3 show that the model reached an mAP_{0.5:0.95} of about 68% on the validation set after 50 epochs. The mAP_{0.5:0.95} represents the mean average accuracy over a range from 0.5 to 0.95 and is the most important metric for evaluating this model. It represents the average Intersect Over Union (IOU) at different threshold values in the range from 0.5 to 0.95 in steps of 0.05. In this context, the threshold value means the confidence level above which the network output is considered to be a detection. The IOU is an accuracy measure for the prediction of the model, which is calculated as the intersection between the prediction of the model and the ground-truth labeling divided by the union of the prediction and labeling. Figure 4 visually illustrates the calculation of this metric. Visual inspection of the Video S2 Supplementary shows that this model performs very well. Every nearby vine was recognized, and even distant vines were detected. In addition, the recognition was stable and consistent. Figure 5 shows some examples of vine trunk detection from vineyards in Istria that were not included in either the training set or in the validation set. A look at the graphs in Figure 3 before and after filtering the dataset confirms that our filtering approach had a significant positive impact on model performance.

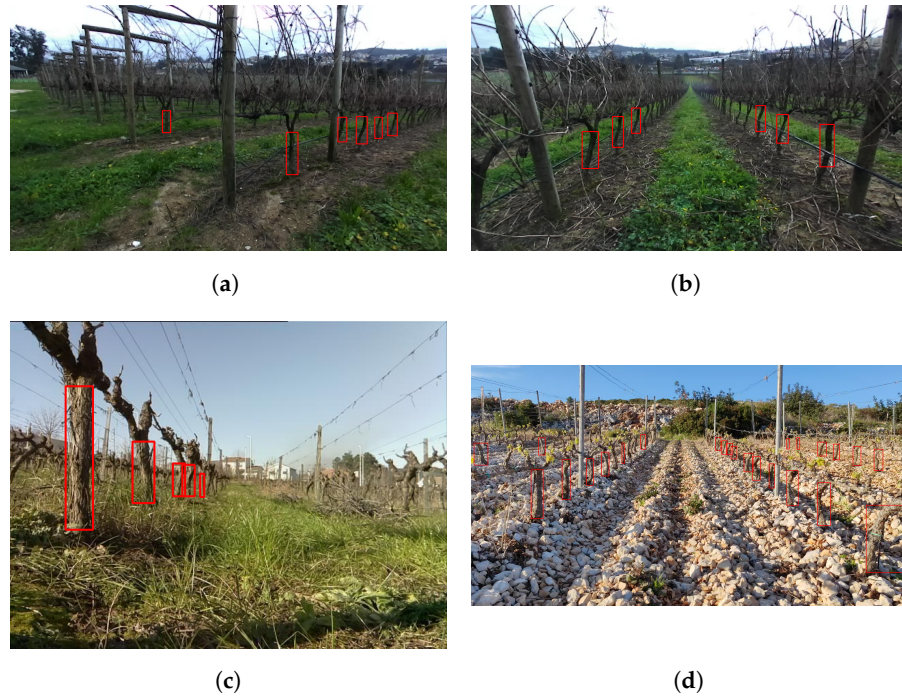


Figure 2. Vine trunk detection dataset sample. The images in (a–c) are samples from VineSet, and the image in (d) is a sample of data from the Korčula vineyard. Red rectangles indicate labeled vines.

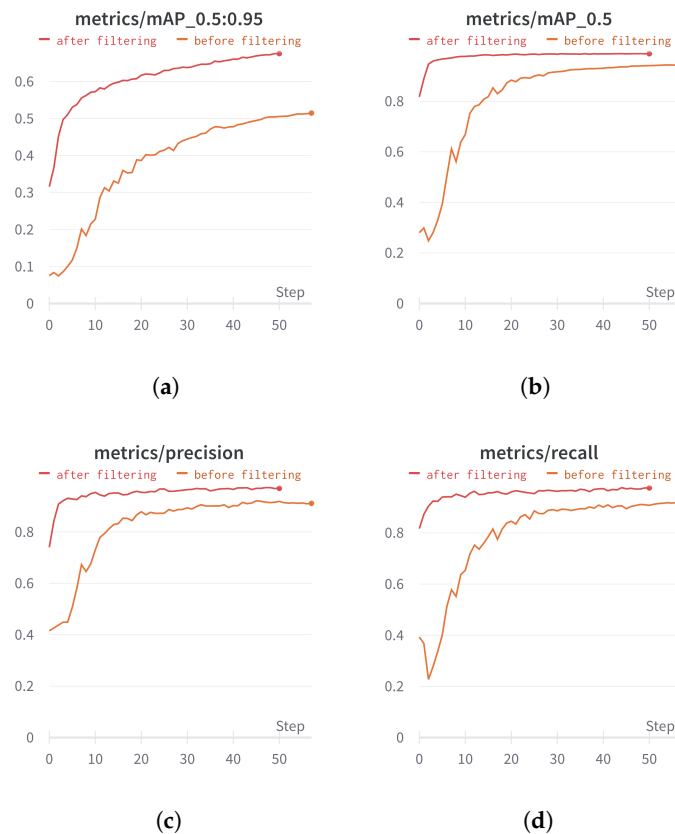


Figure 3. Validation training metrics for the YOLO model before and after dataset filtering. We can see that in (a) the mAP_{0.5:0.95}, which represents the mean average precision of the IOU metric over a 0.5–0.95 range for thresholds, our most important metric, was improved by about 17% after we filtered the dataset. (b) shows the mean average precision on the fixed threshold of 0.5. (c) shows precision or how often a model is correct then predicting the target class. (d) shows recall or what proportion of target classes was identified correctly.

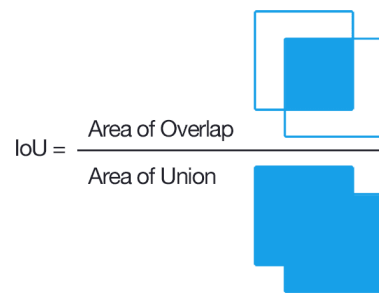


Figure 4. Representation of the IOU metric. The figure shows two bounding boxes: the model-prediction bounding box and the ground-truth bounding box. The overlapping region is the intersect and the combined region is the union.



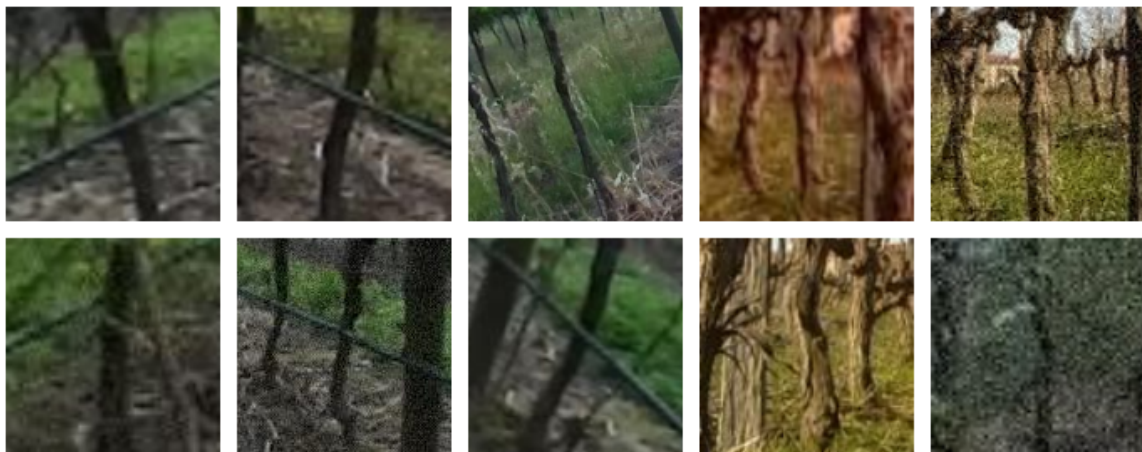
Figure 5. Sample YOLO detection in a vineyard in Istria that is not present in the original dataset. Colored rectangles show individual detections, and the text above the rectangles shows the class (which is always “trunk”) and the detection confidence in a range from 0 to 1.

2.2. Creation of the Data Set

After obtaining a trained model for recognizing vine trunks, we could use it to separate all the vines from our data into individual images. YOLOv5 was run on the entire VineSet as well as on annotated and unannotated footage from Korčula collected as part of the HEKTOR project. Each detection rectangle was converted into a square by setting the shorter side of the rectangle to the value of the longer side. Squares that crossed the image boundary during this process were discarded to avoid black columns in the dataset. All squares were then cropped from their respective images and scaled to a resolution of 128×128 pixels. The reasons for setting the resolution to 128×128 are manifold. The main

reason is the storage space required for the data. At a resolution of, say, 268×268 pixels, the dataset would be four times as large, which would make it much more unwieldy and training much slower. If we keep the amount of data small, we can create prototypes, test ideas, and train much faster.

Using this method, 37,913 images were collected, each containing a centered vine trunk: 34,218 from VineSet with the sample shown in Figure 6a and 3695 from Korčula with the sample shown in Figure 6b. We can see that some images contained more vine trunks, but we were only interested in the vine trunks in the center of the image; i.e., we only wanted the model to determine the segmentation of the centered vine trunk because other vine trunks, if detected, would have their own images and their own segmentations. If the model were to segment all vine trunks in an image, it would have to implicitly perform object detection, which is not its task. By letting the model determine only the segmentation of the centered vine trunk, we simplify its task considerably. Next, a random sample of 300 images was manually annotated using the online application RoboFlow [73], which provides a set of tools for annotating data for machine learning purposes. In our case, we used the autoselect tool, which allows the user to create binary masks with just a few clicks using basic computer vision methods, significantly speeding up the process.



(a)



(b)

Figure 6. Sample of cut-out images of vine trunks from (a) VineSet and (b) the Korčula vineyard using YOLO detection. We can see that each image has a centered vine trunk. It does not matter that there are other trunks in the image as we were only focused on the centered one.

2.3. SSL-ELN

The method used to create our dataset was a semi-supervised learning method called Semi-Supervised Semantic Segmentation with Error Localization Network (SSL-ELN), which was introduced in [45]. With this method, we can train a semantic segmentation decoder with a small set of annotated data and a large set of unannotated data. The SSL-ELN method is based on two segmentation networks: the student network, which is the final model we train, and the teacher network, which is used to generate pseudo-labels. Pseudo-labels are a technique commonly used in semi-supervised learning to apply artificial labels to unlabeled data based on the prediction of the model. The student network is trained with the teacher's pseudo-labels in two ways: through self-learning and contrastive learning. Since models often show limited performance when learning with small amounts of labeled data, the use of pseudo-labels can improve the model's ability to generalize but also carries risks due to their inaccuracy. To mitigate these risks, the SSL-ELN method introduces the Error Localization Network (ELN), which acts as an auxiliary component for identifying potentially inaccurate pseudo-labels in unlabeled images and is integrated with self-learning and contrastive learning approaches, ensuring performance improvements. The ELN helps the model to focus on areas with a high probability of error in segmentation, reducing the damage that inaccurate pseudo-labels can cause. In addition, the ELN is trained with a special learning strategy that simulates different and highly probable errors that may occur during segmentation, improving the model's adaptability and generalization. The benefits of the SSL ELN method have been tested with popular segmentation datasets such as PASCAL VOC 2012 [32] and Cityscapes [31], where the method outperformed the existing best approaches with most test settings [45].

When we tried to train this model, we ran into the same problem again: lack of data. Specifically, it was necessary to split our data into a training set and a validation set in order to evaluate the performance of the model. Considering that we only had 300 annotated images, a standard split of 80% for training and 20% for validation would leave us with only 60 images in the validation set. Due to the small validation set, we could not effectively evaluate the model, and if we increased the validation set, we would have to reduce the training set and risk significantly limiting the effectiveness of the method. As can be seen in Figure 7, the validation metric saturated to about 85% very quickly after the start of training and provided no more information after that. For this reason, we could not determine when we should stop training to avoid overfitting. We did not know whether the last iteration of training was better or worse than the current one.

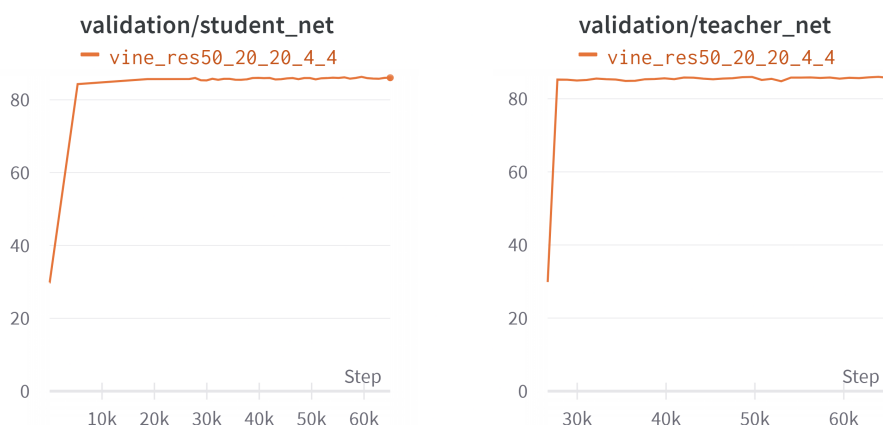


Figure 7. Graph of the metrics on the validation set during the training of the SSL-ELN model. Since our validation set is too small, our validation metrics become saturated shortly after training begins and provide no additional information about the model's ability to generalize.

The solution to this problem is to train the model “blindly”; i.e., to let the model train significantly longer than the validation metrics suggest, as shown in Figure 7. This ensures

that overfitting actually occurs. The usual approach in machine and deep learning is to stop training when the loss in the validation set reaches a minimum or the accuracy of the validation set reaches a maximum to avoid overfitting, but due to the lack of validation data, this was not possible. When we tested this overtrained model on unlabeled test data not included in the training set, we saw that the model mainly predicted noise and did not generalize, as shown in Figure 8. However, when we tested this overfitted model with unlabeled training data, we saw that the model had learned to segment the vines from the training data. These results directly indicated that overfitting had indeed occurred. In Figure 9, examples of the model’s results with the unlabeled training set are shown. Since the goal of this step was not to train a decoder that accurately segments the vines, it did not matter that the model did not generalize because we could process the entire unlabeled dataset with this model to produce an annotated dataset for semantic segmentation.

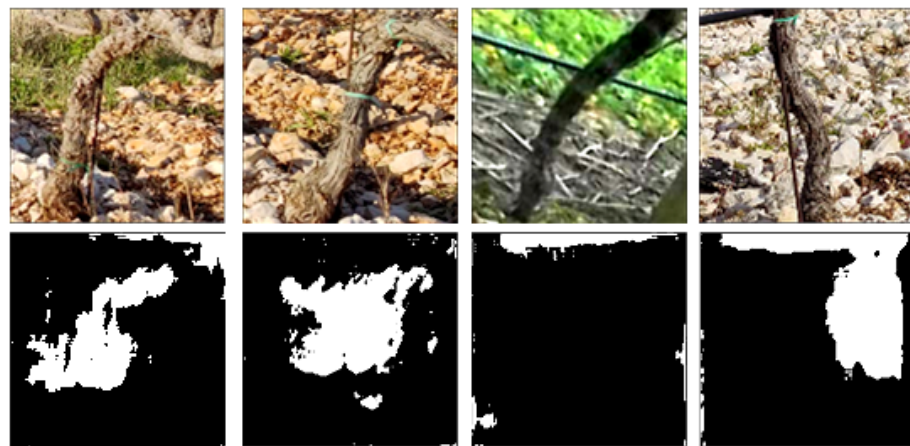


Figure 8. An example of the output of the SSL-ELN method using data from the validation set. The upper row shows the input images and the lower row shows the corresponding binary images generated by the SSL-ELN method. We can see that the decoder generated noise and that the model did not generalize.

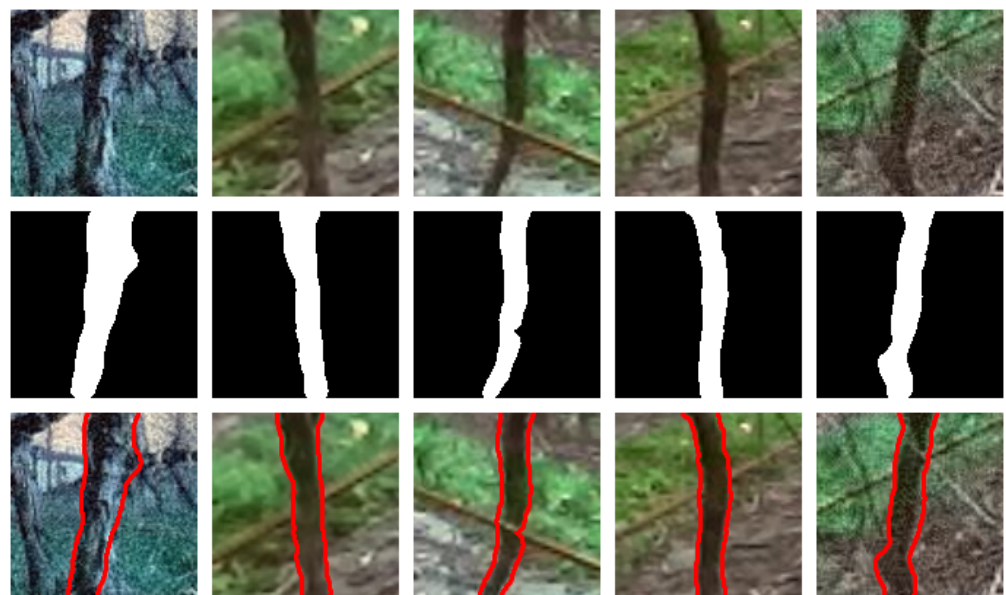


Figure 9. An example of the output of the SSL-ELN method using data from the training set. The first row shows the input images and the row in the middle shows the corresponding binary images

generated by the SSL-ELN method. The lowest row shows the outline of the output for the input image for the purpose of visualization. We can see that the model indeed learned the training set and could segment it effectively.

2.4. Dataset Filtering

Visual inspection of the generated dataset revealed a number of problems. Some examples consisted entirely of noise, and some had accurate detection and noise. We wanted to filter out examples that contained significant amounts of noise or random detections, as these were false negatives that could potentially have a negative impact on the performance of the model. A very primitive but effective method for filtering out noise data in this case would be to look at the number of white pixels in the image. Figure 10 shows a histogram of the number of white pixels in the labels of the entire dataset. We can see that there is a clear jump around the value 1200. If we isolate the examples with less than 1200 white pixels (Figure 11), we can see that the vast majority of the examples were actually noise. The problem arose from very thin vine trunks that also had a small number of white pixels but were correctly labeled. The number of such examples was so small that it was practical to manually isolate these examples and leave them in the set.

Another way to filter out noise data would be to analyze the annotations according to the number of contours they contain. The histogram in Figure 12 shows that most images contained one or two contours. The red line represents a standard deviation from the average and shows us that images with one or two contours made up the majority of the set. Therefore, we can consider data that fell outside of the standard deviation as exceptions. If we isolate all images that have three or more contours (Figure 13), we can see that they all contain significant amounts of noise. We can discard such images. About 2500 examples were discarded using these methods. The fact that these steps require human intervention to filter the data is not ideal but is only a minor shortcoming. The filtering methods automatically isolate candidates that should be discarded, and a human can tell at a glance whether an image is random noise or a valid segmentation. Compared to the manual annotation of 37,913 images, this is a considerable saving in time.

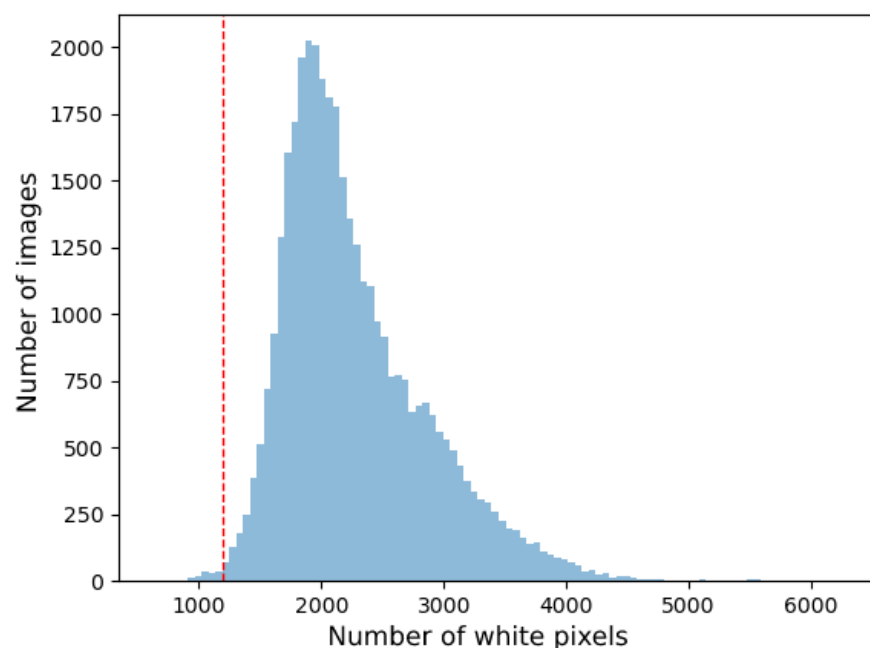


Figure 10. Histogram of the number of white pixels in the entire generated dataset with the threshold value marked in red below which the data were discarded. The threshold value of 1200 was chosen because a sudden increase in the number of white pixels could be seen in the histogram.

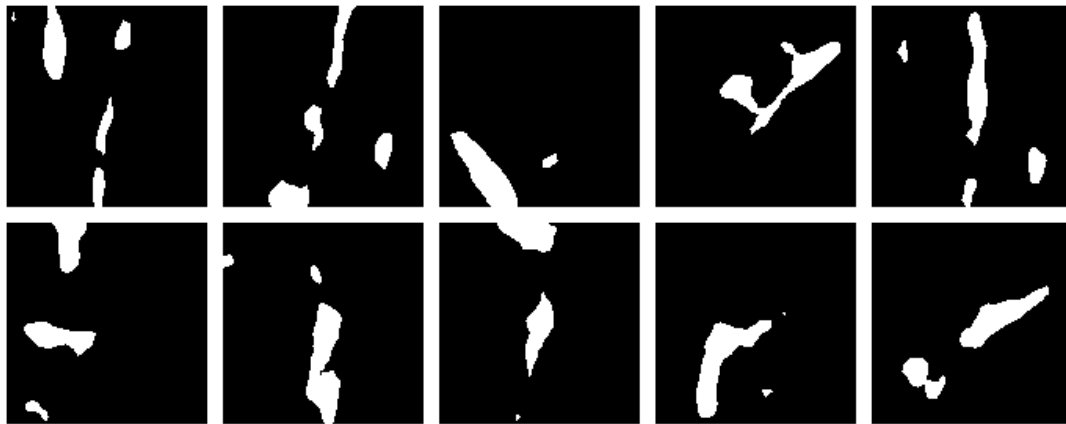


Figure 11. Example of SSL-ELN-generated annotations with 1200 white pixels or less. Some remnants of labeled vine trunks can be seen, but the images cannot be considered valid annotations due to noise and random white blobs in the images.

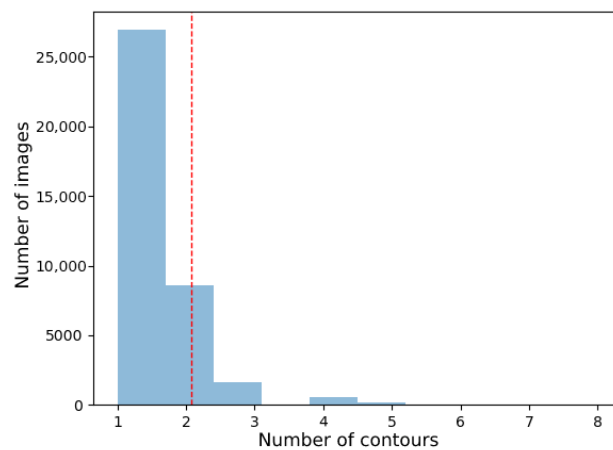


Figure 12. Histogram of the number of contours in the generated dataset. It can be seen that most of the images had one or two contours. The red line represents a standard deviation from the mean and shows us that images with one or two contours made up the majority of the set.



Figure 13. Examples of generated annotations with three or more contours. There are some remnants of labeled vine trunks, but the images cannot be considered valid annotations due to noise and random blobs present in the images.

2.5. Semantic Segmentation Training

Given the large number of available semantic segmentation models, selecting the optimal architecture for our topic was a challenging task. Among the most important criteria that helped us select the most suitable model were accuracy, prediction speed, and memory requirements; i.e., the amount of VRAM the model occupied on the graphics card. We had to select the desired pre-trained encoder and decoder architecture. To find out which model best fulfilled these criteria, we had to run experiments with different combinations of architectures and decoders.

The publicly available project called Segmentation models with pre-trained backbones in PyTorch [74] provided us with a very simple API to train and run different semantic segmentation models and allowed us to try out different pre-trained encoders. We considered the following model architectures: UnetPlusPlus [16], MANET [17], DeepLabV3Plus [13], PAN [20], Link [18], FPN [75], and PSP [19]. The pre-trained encoders we tried were *mobilenet_v2* [25] and *efficientnet-b0* [26]. These encoders were chosen for their small size and high efficiency, which makes them suitable for mobile robot applications. *Efficientnet-b0* has only four million parameters, while *mobilenet_v2* has only two million parameters.

The evaluation was carried out by grid search using combinations of encoders and decoders. We had a total of 12 combinations and trained one after the other. The data were split into a training set of 80% and a validation set of 20%. Each training operation took 40 epochs with a mini-batch size of 32 and a learning rate of 0.0001 using the Adam optimizer [76]. The Adam optimizer is a stochastic gradient descent method based on adaptive estimates of first- and second-order momentum that achieves faster convergence than standard methods. The training uses the so-called dice loss [77]. The dice loss is a more manageable version of the IOU metric. It is useful for semantic segmentation because it normalizes the overall class size in the image. In ordinary binary cross-entropy loss, segmentations that occupy more space in the image are implicitly considered more important, while smaller segmentations tend to be ignored. Normalization by segmentation size gives a more accurate measure of similarity between prediction and label. The dice loss is represented by the following equation

$$DICE = 1 - \frac{2 \sum_i^N p_i g_i}{\sum_i^N p_i^2 + \sum_i^N g_i^2} \quad (1)$$

where p_i is the prediction of the pixel i and g_i is the ground truth; i.e., the label of the pixel i . All training and evaluations were carried out with an RTX 2080Ti graphics card.

3. Evaluation of Results and Discussion

Table 2 shows the evaluation metrics after training each model. Figure 14 shows the data from the last two columns of Table 2 in a graph where the x -axis is the processing time and the y -axis is the value of the IOU metric with the validation set. It can be seen that the PSP model with both encoders was significantly faster than all other models. The PSP model with the *mobilenet_v2* encoder achieved an average processing time of 4.54 ms, which was 2.75 times faster than the Link model with the *mobilenet_v2* encoder, which was next in line. We also see that the *efficientnet-b0* was about 8.75 ms slower than the *mobilenet_v2* in all models except PSP, making *mobilenet_v2* a more suitable encoder for this application. Of course, there is always a trade-off. All models except PSP had an accuracy between 83% and 84%, while PSP achieved an accuracy of about 81%. Since the intention was to use these models in mobile robotic systems, the efficiency of the models was one of the most important metrics. A decrease in accuracy of about 3% was thus much less important than the speedup provided by the PSP model. Therefore, we can conclude that the PSP model with the *mobilenet_v2* encoder offers a compromise that makes it the most suitable model for our application.

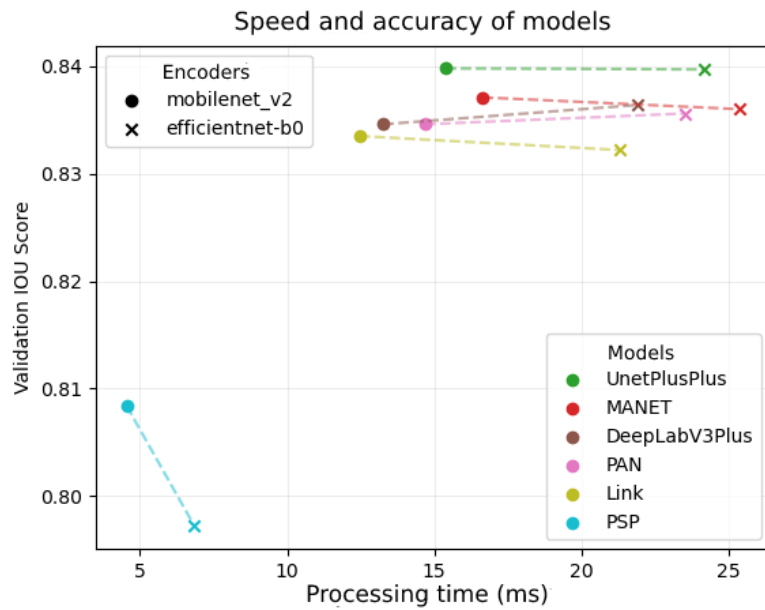


Figure 14. The data from Table 2 are displayed in a format that compares the speed of the model *x*-axis and its accuracy with validation data *y*-axis. The color represents the architecture of the model, while the shape of the dot represents the selected encoder. We can see that the efficientnet-b0 encoder significantly reduced the processing time in all models. We can also see that the PSP model was 2.7 times faster than all others, but only 3% worse.

Table 2. Evaluation metrics for the combination of model and encoder. The best values in each column are highlighted in bold. Processing time was calculated by averaging the measured time of 100 samples with a mini-batch size of 100 and expressed in milliseconds.

Model	Encoder	Train Dice Loss	Train IOU Score	Valid Dice Loss	Valid IOU Score	Processing Time (ms)
UnetPlusPlus	Mobilenet_v2	0.08146	0.8495	0.08728	0.8398	15.39
MANET	Mobilenet_v2	0.08404	0.8452	0.0889	0.8371	16.65
DeepLabV3Plus	Mobilenet_v2	0.08487	0.8446	0.0909	0.8346	13.27
PAN	Mobilenet_v2	0.08388	0.8467	0.09122	0.8346	14.69
Link	Mobilenet_v2	0.08524	0.8432	0.09108	0.8335	12.46
PSP	Mobilenet_v2	0.1045	0.8121	0.1069	0.8084	4.54
UnetPlusPlus	Efficientnet-b0	0.07869	0.8543	0.08734	0.8397	24.19
MANET	Efficientnet-b0	0.08358	0.846	0.08952	0.836	25.4
DeepLabV3Plus	Efficientnet-b0	0.08452	0.8451	0.08983	0.8364	21.93
PAN	Efficientnet-b0	0.08442	0.846	0.09079	0.8356	23.53
Link	Efficientnet-b0	0.08518	0.8433	0.09183	0.8322	21.31
PSP	Efficientnet-b0	0.1128	0.7986	0.1138	0.7972	6.85

3.1. Final Vine Trunk Segmentation Process

Once we had a well-trained semantic segmentation model, we could combine the whole process by integrating all the systems we set up. The process is as follows:

1. The image is read from the video or from the camera.
2. The image is pre-processed in a way that is suitable for the YOLO model.
3. From the YOLO model, we get vine trunk detections. We memorize the positions and shapes of the detections.
4. We reshape the detections into squares and cut out the vine trunks from the image.
5. The cut-out images are segmented using the trained PSP network.
6. The segmented binary images are cut out and reshaped so that they have the same shape as the original detections.

7. The reshaped binary images are pasted over the original image with a customized alpha channel to make them transparent.

A diagram of this process is shown in Figure 15. Examples of the results of this process can be found in Figure 16 and in the Video S3 at Supplementary.

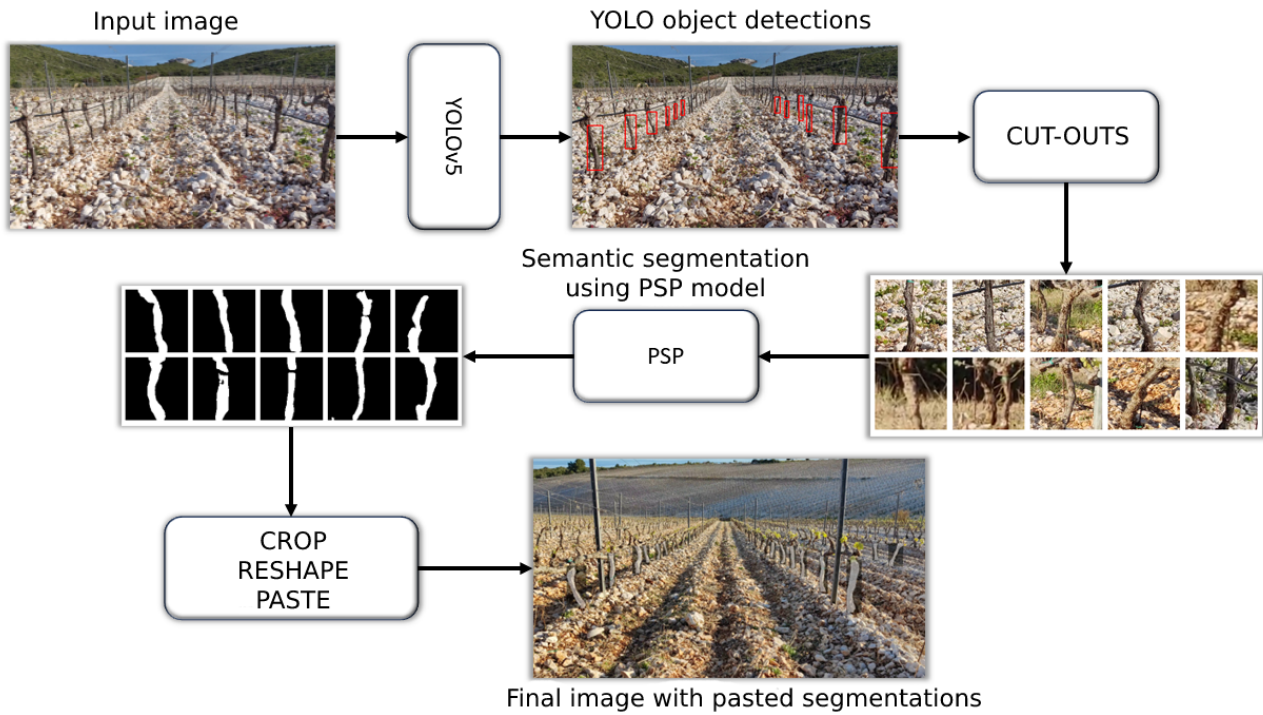


Figure 15. Diagram of the final procedure for segmenting vine trunks. The input image is pre-processed and sent through the YOLO network to recognize the vine trunks in the form of rectangles shown in red. The rectangles are then converted into squares, cut out from the original image, and resized to 128×128 pixels. The small cropped images are then sent through the PSP model, which provides binary masks of the vine trunks. These masks are then cropped so that they have the same shape as the original rectangle, resized to the same size, and inserted into the original image.



(a)

Figure 16. Cont.



(b)

Figure 16. Examples of the final results of the procedure described. (a) The example shows an image from Korčula that was not included in the original training dataset. (b) The example shows an image from Istria that was not included in any dataset. The red lines show the outlines of semantic segmentation masks generated by the PSP model using the cropped images from YOLO object detection.

3.2. Limitations and Shortcomings

We can see that some vines were not recognized, especially vines that were either far away or very thin. This was due to the limited resolution of YOLO object detection. While the images shown in Figure 16 are relatively large, the images passed through YOLO are downsampled to a resolution of 640×480 . This means that distant or very small vine trunks are simply not displayed with enough pixels to be recognized. However, this limitation does not have a negative effect on our goal, as only the vine trunks in the immediate vicinity need to be recognized for the robot to navigate and execute tasks with vines.

Although our data covered a wide range of vineyards and vine trunks, we cannot guarantee that our method is transferable to all types of environments. It is worth noting that our models were trained exclusively on data from European vineyards. Therefore, there is a plausible chance that the efficiency and accuracy of these models will decrease if they are used in vineyards that are significantly different in structure or contain different vine varieties. This means that our method may need to be fine-tuned if it is used in a significantly different environment. While this is not ideal, it is not a major drawback. As this study shows, more data from different vineyards can be added without requiring a significant amount of annotation time.

Another shortcoming of our method is the resolution of the segmentation of the vine trunks. The low-resolution segmentation models were chosen for simplicity and efficiency. While it would be trivial to increase the resolution of the data and models, this could cause the model to become so inefficient that it is no longer suitable for a mobile robot platform. This requires further research.

3.3. Discussion

The approach presented in this article is less elegant than the usual semantic segmentation approach, where the model segments the entire image at once and performs both object detection and semantic segmentation. It is better suited to low-power mobile robotic systems such as those used in autonomous robotics. In the case of a vineyard robot, it is not necessary to segment all the vine trunks at all times. Most of the time, only object detection is needed for tasks such as navigation, while semantic segmentation is only required for specific vine trunks in specific situations (e.g., for planning the robotic arm's movements).

The proposed approach adapts to this circumstance and allows the robot to semantically segment vine trunks as needed, which reduces processing time. In this way, we can use a semantic segmentation model that has a much smaller memory footprint than would be required to segment the entire image. The more challenging task of recognizing vine trunks is handled by a dedicated object detection model. With this customized approach, we achieve a more efficient use of system resources available in low-power hardware.

4. Conclusions

We have developed an approach for generating a new dataset for the semantic segmentation of vine trunks, a crucial step towards the development of autonomous robots for vineyard management. Starting from vine detection with the YOLOv5 model and continuing to semi-supervised training of semantic segmentation with the SSL-ELN method, we have shown that it is possible to reduce the need for human data annotation, which is usually a time-consuming and expensive process.

One of the main challenges was the lack of high-quality data for vineyard segmentation. Through a detailed analysis of the existing VineSet dataset, we were able to identify and address its critical shortcomings, providing a reliable basis for further model training. Furthermore, by using semi-supervised learning, we were able to train a semantic segmentation decoder with fewer annotated data than normally required, resulting in significant savings in time and resources. In conducting this research, only 300 vines were annotated by humans and we created a dataset of about 35,500 images. This represents less than 1% of the effort that would be required if all images were manually segmented.

Based on the studied dataset, we compared different architectures of semantic segmentation models to find the most suitable model for our application in mobile robotic systems. After a thorough evaluation, we concluded that the PSP model with the mobilenet_v2 encoder represented the best compromise between accuracy, speed, and memory requirements. The results of this study show that it is now possible to annotate data with fewer and fewer human work hours, contributing to further innovations in the field of machine learning.

A continuation and further development of this work would be to use the developed method to annotate full semantic segmentation images, not just image sections. This would require a much larger semantic segmentation model, which might make it unsuitable for mobile robotic systems. Furthermore, the system would then no longer be able to select the objects to be segmented, which would also make it less suitable for mobile robot applications as it would increase the processing time, but it would enable further research in the field of semantic segmentation of vineyards.

Supplementary Materials: The following supporting information can be downloaded at: <https://www.mdpi.com/article/10.3390/robotics13020020/s1>.

Author Contributions: Conceptualization, P.S.; methodology, P.S.; software, P.S.; validation, P.S.; formal analysis, P.S.; investigation, P.S.; resources, P.S.; data curation, P.S.; writing—review and editing, P.S., I.H. and Z.K.; visualization, P.S. and I.H.; supervision, I.H. and Z.K.; project administration Z.K.; funding acquisition, Z.K. All authors have read and agreed to the published version of the manuscript.

Funding: The research work presented in this article was supported by the project titled Heterogeneous Autonomous Robotic System in Viticulture and Mariculture (HEKTOR), financed by the European Union through the European Regional Development Fund—The Competitiveness and Cohesion Operational Programme (KK.01.1.1.04.0036).

Data Availability Statement: The data that support the findings of this study are available on request from the corresponding author.

Conflicts of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest. At the time of the research, all authors were employed by or associated with the Faculty of Electrical Engineering and Computing, University of Zagreb.

References

1. Heterogeneous Autonomous Robotic System in Viticulture and Mariculture (HEKTOR project)—2020–2023. Available online: <http://hektor.fer.hr/en/homepage/> (accessed on 9 April 2020).
2. Kapetanović, N.; Goričanec, J.; Vatavuk, I.; Hrabar, I.; Stuhne, D.; Vasiljević, G.; Kovačić, Z.; Mišković, N.; Antolović, N.; Anić, M.; et al. Heterogeneous Autonomous Robotic System in Viticulture and Mariculture: Vehicles Development and Systems Integration. *Sensors* **2022**, *22*, 2961. [[CrossRef](#)] [[PubMed](#)]
3. Hrabar, I.; Kovačić, Z. Localization of Mobile Manipulator in Vineyards for Autonomous Task Execution. *Machines* **2023**, *11*, 414. [[CrossRef](#)]
4. Hrabar, I.; Goričanec, J.; Kovačić, Z. Towards Autonomous Navigation of a Mobile Robot in a Steep Slope Vineyard. In Proceedings of the 2021 44th International Convention on Information, Communication and Electronic Technology (MIPRO), Opatija, Croatia, 27 September–1 October 2021; pp. 1119–1124. [[CrossRef](#)]
5. Vatavuk, I.; Vasiljević, G.; Kovačić, Z. Task Space Model Predictive Control for Vineyard Spraying with a Mobile Manipulator. *Agriculture* **2022**, *12*, 381. [[CrossRef](#)]
6. Vatavuk, I.; Stuhne, D.; Vasiljević, G.; Kovačić, Z. Direct Drive Brush-Shaped Tool with Torque Sensing Capability for Compliant Robotic Vine Suckering. *Sensors* **2023**, *23*, 1195. [[CrossRef](#)] [[PubMed](#)]
7. Khan, M.Z.; Gajendran, M.K.; Lee, Y.; Khan, M.A. Deep Neural Architectures for Medical Image Semantic Segmentation: Review. *IEEE Access* **2021**, *9*, 83002–83024. [[CrossRef](#)]
8. Yuan, R.; Xu, J.; Li, X.; Zhang, Y.; Feng, R.; Zhang, X.; Zhang, T.; Gao, S. MedSeq: Semantic Segmentation for Medical Image Sequences. In Proceedings of the 2022 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), Las Vegas, NV, USA, 6–8 December 2022; pp. 1356–1361. [[CrossRef](#)]
9. Jebamkiyous, H.H.; Kashef, R. Deep Learning-Based Semantic Segmentation in Autonomous Driving. In Proceedings of the 2021 IEEE 23rd Int Conf on High Performance Computing & Communications; 7th Int Conf on Data Science & Systems; 19th Int Conf on Smart City; 7th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys), Haikou, China, 20–22 December 2021; pp. 1367–1373. [[CrossRef](#)]
10. Tong, L.; Song, K.; Tian, H.; Man, Y.; Yan, Y.; Meng, Q. SG-Grasp: Semantic Segmentation Guided Robotic Grasp Oriented to Weakly Textured Objects Based on Visual Perception Sensors. *IEEE Sens. J.* **2023**, *23*, 28430–28441. [[CrossRef](#)]
11. Terreran, M.; Antonello, M.; Ghidoni, S. Boat Hunting with Semantic Segmentation for Flexible and Autonomous Manufacturing. In Proceedings of the 2019 European Conference on Mobile Robots (ECMR), Prague, Czech Republic, 4–6 September 2019; pp. 1–8. [[CrossRef](#)]
12. Chen, L.C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; Yuille, A.L. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 834–848. [[CrossRef](#)] [[PubMed](#)]
13. Chen, L.C.; Zhu, Y.; Papandreou, G.; Schroff, F.; Adam, H. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. In *Computer Vision—ECCV 2018, Proceedings of the 15th European Conference, Munich, Germany, 8–14 September 2018*; Part VII; Springer: Berlin/Heidelberg, Germany, 2018; pp. 833–851. [[CrossRef](#)]
14. Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 3431–3440. [[CrossRef](#)]
15. Noh, H.; Hong, S.; Han, B. Learning Deconvolution Network for Semantic Segmentation. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 1520–1528. [[CrossRef](#)]
16. Zhou, Z.; Rahman Siddiquee, M.M.; Tajbakhsh, N.; Liang, J. UNet++: A Nested U-Net Architecture for Medical Image Segmentation. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support, Proceedings of the 4th International Workshop, DLMIA 2018, and 8th International Workshop, ML-CDS 2018, Granada, Spain, 20 September 2018*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 3–11. [[CrossRef](#)]
17. Fan, T.; Wang, G.; Li, Y.; Wang, H. MA-Net: A Multi-Scale Attention Network for Liver and Tumor Segmentation. *IEEE Access* **2020**, *8*, 179656–179665. [[CrossRef](#)]
18. Chaurasia, A.; Culurciello, E. LinkNet: Exploiting encoder representations for efficient semantic segmentation. In Proceedings of the 2017 IEEE Visual Communications and Image Processing (VCIP), St. Petersburg, FL, USA, 10–13 December 2017; IEEE: Piscataway, NJ, USA, 2017. [[CrossRef](#)]
19. Zhao, H.; Shi, J.; Qi, X.; Wang, X.; Jia, J. Pyramid Scene Parsing Network. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 6230–6239. [[CrossRef](#)]
20. Li, H.; Xiong, P.; An, J.; Wang, L. Pyramid Attention Network for Semantic Segmentation. *arXiv* **2018**, arXiv:1805.10180.
21. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [[CrossRef](#)] [[PubMed](#)]
22. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Columbus, OH, USA, 23–28 June 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 580–587. [[CrossRef](#)]
23. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature Pyramid Networks for Object Detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 936–944. [[CrossRef](#)]

24. Jocher, G.; Stoken, A.; Borovec, J.; NanoCode012; ChristopherSTAN; Changyu, L.; Laughing; tkianai; Hogan, A.; lorenzomamma; et al. Ultralytics/yolov5: v3.1—Bug Fixes and Performance Improvements, Zenodo. 2020. Available online: <https://zenodo.org/records/4154370> (accessed on 18 January 2024).
25. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4510–4520. [[CrossRef](#)]
26. Tan, M.; Le, Q. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In Proceedings of the 36th International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; Chaudhuri, K., Salakhutdinov, R., Eds.; Volume 97, pp. 6105–6114.
27. Wang, W.; Dai, J.; Chen, Z.; Huang, Z.; Li, Z.; Zhu, X.; Hu, X.; Lu, T.; Lu, L.; Li, H.; et al. InternImage: Exploring Large-Scale Vision Foundation Models with Deformable Convolutions. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 17–24 June 2023; pp. 14408–14419. [[CrossRef](#)]
28. Fang, Y.; Wang, W.; Xie, B.; Sun, Q.; Wu, L.; Wang, X.; Huang, T.; Wang, X.; Cao, Y. EVA: Exploring the Limits of Masked Visual Representation Learning at Scale. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 17–24 June 2023; pp. 19358–19369. [[CrossRef](#)]
29. Li, C.; Li, L.; Geng, Y.; Jiang, H.; Cheng, M.; Zhang, B.; Ke, Z.; Xu, X.; Chu, X. YOLOv6 v3.0: A Full-Scale Reloading. *arXiv* **2023**, arXiv:2301.05586.
30. Shinya, Y. USB: Universal-Scale Object Detection Benchmark. In Proceedings of the 33rd British Machine Vision Conference 2022, BMVC 2022, London, UK, 21–24 November 2022; BMVA Press: Newcastle, UK, 2022. [[CrossRef](#)]
31. Cordts, M.; Omran, M.; Ramos, S.; Rehfeld, T.; Enzweiler, M.; Benenson, R.; Franke, U.; Roth, S.; Schiele, B. The Cityscapes Dataset for Semantic Urban Scene Understanding. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 3213–3223. [[CrossRef](#)]
32. Everingham, M.; Gool, L.V.; Williams, C.K.I.; Winn, J.M.; Zisserman, A. The Pascal Visual Object Classes (VOC) Challenge. *Int. J. Comput. Vis.* **2010**, *88*, 303–338. [[CrossRef](#)]
33. Geiger, A.; Lenz, P.; Urtasun, R. Are we ready for autonomous driving? The KITTI vision benchmark suite. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 3354–3361. [[CrossRef](#)]
34. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. *Microsoft COCO: Common Objects in Context, Proceedings of the Computer Vision—ECCV 2014, Zurich, Switzerland, 6–12 September 2014*; Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T., Eds.; Springer International Publishing: Cham, Switzerland, 2014; pp. 740–755. [[CrossRef](#)]
35. Alonso, I.; Sabater, A.; Ferstl, D.; Montesano, L.; Murillo, A.C. Semi-Supervised Semantic Segmentation with Pixel-Level Contrastive Learning from a Class-wise Memory Bank. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 10–17 October 2021; pp. 8199–8208. [[CrossRef](#)]
36. Chen, L.C.; Lopes, R.G.; Cheng, B.; Collins, M.D.; Cubuk, E.D.; Zoph, B.; Adam, H.; Shlens, J. Naive-Student: Leveraging Semi-Supervised Learning in Video Sequences for Urban Scene Segmentation. In *Proceedings of the Computer Vision—ECCV 2020, Glasgow, UK, 23–28 August 2020*; Vedaldi, A., Bischof, H., Brox, T., Frahm, J.M., Eds.; Springer International Publishing: Cham, Switzerland, 2020; pp. 695–714. [[CrossRef](#)]
37. He, R.; Yang, J.; Qi, X. Re-distributing Biased Pseudo Labels for Semi-supervised Semantic Segmentation: A Baseline Investigation. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 10–17 October 2021; pp. 6910–6920. [[CrossRef](#)]
38. Kalluri, T.; Varma, G.; Chandraker, M.; Jawahar, C. Universal Semi-Supervised Semantic Segmentation. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 5258–5269. [[CrossRef](#)]
39. Ke, R.; Aviles-Rivero, A.I.; Pandey, S.; Reddy, S.; Schönlieb, C.B. A Three-Stage Self-Training Framework for Semi-Supervised Semantic Segmentation. *IEEE Trans. Image Process.* **2022**, *31*, 1805–1815. [[CrossRef](#)] [[PubMed](#)]
40. Ke, Z.; Qiu, D.; Li, K.; Yan, Q.; Lau, R.W.H. Guided Collaborative Training for Pixel-Wise Semi-Supervised Learning. In *Computer Vision—ECCV 2020, Proceedings of the 16th European Conference, Glasgow, UK, 23–28 August 2020*; Part XIII; Springer: Berlin/Heidelberg, Germany, 2020; pp. 429–445. [[CrossRef](#)]
41. Lai, X.; Tian, Z.; Jiang, L.; Liu, S.; Zhao, H.; Wang, L.; Jia, J. Semi-supervised Semantic Segmentation with Directional Context-aware Consistency. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021; pp. 1205–1214. [[CrossRef](#)]
42. Li, D.; Yang, J.; Kreis, K.; Torralba, A.; Fidler, S. Semantic Segmentation with Generative Models: Semi-Supervised Learning and Strong Out-of-Domain Generalization. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021; pp. 8296–8307. [[CrossRef](#)]
43. Mendel, R.; de Souza, L.A.; Rauber, D.; Papa, J.P.; Palm, C. Semi-supervised Segmentation Based on Error-Correcting Supervision. In *Proceedings of the Computer Vision—ECCV 2020*; Vedaldi, A., Bischof, H., Brox, T., Frahm, J.M., Eds.; Springer International Publishing: Cham, Switzerland, 2020; pp. 141–157. [[CrossRef](#)]
44. Mittal, S.; Tatarchenko, M.; Brox, T. Semi-Supervised Semantic Segmentation with High- and Low-Level Consistency. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *43*, 1369–1379. [[CrossRef](#)] [[PubMed](#)]

45. Kwon, D.; Kwak, S. Semi-Supervised Semantic Segmentation with Error Localization Network. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 9957–9967. [[CrossRef](#)]
46. Hyun Cho, J.; Mall, U.; Bala, K.; Hariharan, B. PiCIE: Unsupervised Semantic Segmentation using Invariance and Equivariance in Clustering. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021; pp. 16789–16799. [[CrossRef](#)]
47. Van Gansbeke, W.; Vandenhende, S.; Georgoulis, S.; Van Gool, L. Unsupervised Semantic Segmentation by Contrasting Object Mask Proposals. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 10–17 October 2021; pp. 10032–10042. [[CrossRef](#)]
48. Ouali, Y.; Hudelot, C.; Tami, M. Autoregressive Unsupervised Image Segmentation. In *Computer Vision—ECCV 2020, Proceedings of the 16th European Conference, Glasgow, UK, 23–28 August 2020*; Part VII; Springer: Berlin/Heidelberg, Germany, 2020; pp. 142–158. [[CrossRef](#)]
49. Ahn, J.; Cho, S.; Kwak, S. Weakly Supervised Learning of Instance Segmentation with Inter-Pixel Relations. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 2204–2213. [[CrossRef](#)]
50. Ahn, J.; Kwak, S. Learning Pixel-Level Semantic Affinity with Image-Level Supervision for Weakly Supervised Semantic Segmentation. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4981–4990. [[CrossRef](#)]
51. Chen, L.; Wu, W.; Fu, C.; Han, X.; Zhang, Y. Weakly Supervised Semantic Segmentation with Boundary Exploration. In *Computer Vision—ECCV 2020, Proceedings of the 16th European Conference, Glasgow, UK, 23–28 August 2020*; Part XXVI; Springer: Berlin/Heidelberg, Germany, 2020; pp. 347–362. [[CrossRef](#)]
52. Zhang, D.; Zhang, H.; Tang, J.; Hua, X.S.; Sun, Q. *Causal Intervention for Weakly-Supervised Semantic Segmentation, Proceedings of the Advances in Neural Information Processing Systems, Virtual, 6–12 December 2020*; Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2020; Volume 33, pp. 655–666.
53. Huang, Z.; Wang, X.; Wang, J.; Liu, W.; Wang, J. Weakly-Supervised Semantic Segmentation Network with Deep Seeded Region Growing. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7014–7023. [[CrossRef](#)]
54. Kwak, S.; Hong, S.; Han, B. Weakly Supervised Semantic Segmentation Using Superpixel Pooling Network. In Proceedings of the AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 4–9 February 2017; Volume 31. [[CrossRef](#)]
55. Sun, G.; Wang, W.; Dai, J.; Van Gool, L. *Mining Cross-Image Semantics for Weakly Supervised Semantic Segmentation, Proceedings of the Computer Vision—ECCV 2020, Glasgow, UK, 23–28 August 2020*; Vedaldi, A., Bischof, H., Brox, T., Frahm, J.M., Eds.; Springer International Publishing: Cham, Switzerland, 2020; pp. 347–365. [[CrossRef](#)]
56. Wang, X.; You, S.; Li, X.; Ma, H. Weakly-Supervised Semantic Segmentation by Iteratively Mining Common Object Features. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 1354–1362. [[CrossRef](#)]
57. Hoffman, J.; Tzeng, E.; Park, T.; Zhu, J.Y.; Isola, P.; Saenko, K.; Efros, A.; Darrell, T. CyCADA: Cycle-Consistent Adversarial Domain Adaptation. In Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; Dy, J., Krause, A., Eds.; Volume 80, pp. 1989–1998.
58. Kang, G.; Wei, Y.; Yang, Y.; Zhuang, Y.; Hauptmann, A. Pixel-Level Cycle Association: A New Perspective for Domain Adaptive Semantic Segmentation. In *Advances in Neural Information Processing Systems*; Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2020; Volume 33, pp. 3569–3580.
59. Li, Y.; Yuan, L.; Vasconcelos, N. Bidirectional Learning for Domain Adaptation of Semantic Segmentation. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 6929–6938. [[CrossRef](#)]
60. Tsai, Y.H.; Hung, W.C.; Schuster, S.; Sohn, K.; Yang, M.H.; Chandraker, M. Learning to Adapt Structured Output Space for Semantic Segmentation. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7472–7481. [[CrossRef](#)]
61. Tsai, Y.H.; Sohn, K.; Schuster, S.; Chandraker, M. Domain Adaptation for Structured Output via Discriminative Patch Representations. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1456–1465. [[CrossRef](#)]
62. Zou, Y.; Yu, Z.; Vijaya Kumar, B.V.K.; Wang, J. *Unsupervised Domain Adaptation for Semantic Segmentation via Class-Balanced Self-training, Proceedings of the Computer Vision—ECCV 2018*; Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y., Eds.; Springer: Cham, Switzerland, 2018; pp. 297–313. [[CrossRef](#)]
63. Aguiar, A.S.; Magalhães, S. Grape Bunch and Vine Trunk Dataset for Deep Learning Object Detection. Zenodo. 2021. Available online: <https://zenodo.org/records/5139598> (accessed on 18 January 2024).
64. Kirillov, A.; Mintun, E.; Ravi, N.; Mao, H.; Rolland, C.; Gustafson, L.; Xiao, T.; Whitehead, S.; Berg, A.C.; Lo, W.Y.; et al. Segment Anything. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Paris, France, 2–3 October 2023; pp. 4015–4026.

65. Shi, P.; Qiu, J.; Abaxi, S.M.D.; Wei, H.; Lo, F.P.W.; Yuan, W. Generalist Vision Foundation Models for Medical Imaging: A Case Study of Segment Anything Model on Zero-Shot Medical Segmentation. *Diagnostics* **2023**, *13*, 1947. [[CrossRef](#)] [[PubMed](#)]
66. Zhang, C.; Liu, L.; Cui, Y.; Huang, G.; Lin, W.; Yang, Y.; Hu, Y. A Comprehensive Survey on Segment Anything Model for Vision and Beyond. *arXiv* **2023**, arXiv:2305.08196.
67. Rangnekar, A.; Kanan, C.; Hoffman, M. Semantic Segmentation with Active Semi-Supervised Learning. In Proceedings of the 2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), Waikoloa, HI, USA, 2–7 January 2023; pp. 5955–5966. [[CrossRef](#)]
68. Souly, N.; Spampinato, C.; Shah, M. Semi Supervised Semantic Segmentation Using Generative Adversarial Network. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 5689–5697. [[CrossRef](#)]
69. French, G.; Laine, S.; Aila, T.; Mackiewicz, M.; Finlayson, G. Semi-supervised semantic segmentation needs strong, varied perturbations. In Proceedings of the British Machine Vision Conference, BMVC, London, UK, 7–10 September 2020.
70. Liu, S.; Zhi, S.; Johns, E.; Davison, A.J. Bootstrapping Semantic Segmentation with Regional Contrast. In Proceedings of the Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, 25–29 April 2022.
71. Zhou, Y.; Xu, H.; Zhang, W.; Gao, B.; Heng, P.A. C3-SemiSeg: Contrastive Semi-Supervised Segmentation via Cross-Set Learning and Dynamic Class-Balancing. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Virtual Event, 11–17 October 2021; pp. 7036–7045.
72. Wang, C.; Bochkovskiy, A.; Liao, H. YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors. In Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 17–24 June 2023; IEEE Computer Society: Los Alamitos, CA, USA, 2023; pp. 7464–7475. [[CrossRef](#)]
73. Dwyer, B.; Nelson, J. RoboFlow (Version 1.0). 2022. Available online: <https://roboflow.com> (accessed on 18 January 2024).
74. Iakubovskii, P. Segmentation Models with Pretrained Backbones in PyTorch. Available online: https://github.com/qubvel/segmentation_models.pytorch (accessed on 18 January 2024).
75. Kirillov, A.; He, K.; Girshick, R.; Dollár, P. A Unified Architecture for Instance and Semantic Segmentation. *arXiv* **2017**, arXiv:2112.04603.
76. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. In Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, 7–9 May 2015; Conference Track Proceedings; Bengio, Y., LeCun, Y., Eds.; 2015. [[CrossRef](#)]
77. Sudre, C.H.; Li, W.; Vercauteren, T.; Ourselin, S.; Cardoso, M.J. Generalised Dice Overlap as a Deep Learning Loss Function for Highly Unbalanced Segmentations. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*; Springer International Publishing: Cham, Switzerland, 2017; pp. 240–248. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.