Pseudo-label Generation for Agricultural Robotics Applications

Thomas A. Ciarfuglia*†, Ionut Marian Motoi†, Leonardo Saraceni†, Daniele Nardi
Department of Computer Science, Management and Automation Engineering (DIAG)
Sapienza University of Rome, Italy
{ciarfuglia, saraceni, motoi, nardi}@diag.uniroma1.it

Abstract

In the context of table grape cultivation there is rising interest in robotic solutions for harvesting, pruning, precision spraying and other agronomic tasks. Perception algorithms at the core of these systems require large amounts of labelled data, which in this context is often not available.

In this work, we propose a semi-supervised solution to reduce the data needed to get state-of-the-art detection and segmentation of fruits in orchards. We present the case of table grape vineyards in southern Lazio (Italy) since grapes are a difficult fruit to segment due to occlusion, color and general illumination conditions. We consider the concrete scenario where the source labelled data is wine grape, while the target data is table grape, with considerable covariate shift. Starting from a simple video input, our method generates first bounding box labels, leveraging the structure from motion information, then segmentation masks, using the same weakly generated bounding box labels and a refining step based on Grabcut.

This system is able to produce labels that considerably reduce the covariate shift from source to target data and that requires very limited data acquisition effort. Comparisons with State-of-the-art supervised solutions show how our methods are able to train new models that achieve high performances with few labelled images, with very simple labelling.

1. Introduction

Robotic applications in agriculture target some critical problems such as the lack of manpower for physically demanding tasks, or the reduction of chemicals in the environment with precision spraying, to name a few examples. Each of these applications relies on a perception system able to detect or segment the objects in the scene with performances good enough to reliably accomplish the task at hand. In the last decade, the main robotic perception advances are related to deep learning techniques (see for example [5, 13]). However, the availability of labelled data is one of the factors that limit the greater diffusion of these approaches. In particular, the need for data is related to the sources of variability that are commonly found in the field: uneven distribution of vegetation, intra-species variability, illumination, occlusion and clutter. From a technical point of view, all these aspects translate to covariate shifts and a lack of labelled samples. In this context, it is difficult to collect a good amount of labelled images that catch the actual distribution variability. For this reason, methods that help a robotic system to collect field data and unsupervisedly use them to bridge the covariate shift become desirable.

We explore methods that could help in training detection and instance segmentation algorithms with few labelled data. We explicitly consider the case where a small amount of labelled data from a similar cultivation has been collected and labelled (the source dataset) but is not enough to get acceptable detection and segmentation performances on a different vineyard, with consistent covariate distribution shift (target dataset). Our test target data are table grapes cultivated in Aprilia, southern Lazio, while our source dataset is the wine grape dataset presented in [19]. Our contribution consists in a combination of weakly and semi supervised techniques to increase the performance of detection and segmentation algorithms, while dealing with the lack of labelled data and the covariate shift problem. We also compare with the State-of-the-art on the example application of yield estimation. In the following, we give an overview of the work done in detection for yield estimation, and at the end of the Section we summarize the contribution of this work.

2. Related Works

In this work, we focus on the methods that rely on vision sensors, such as simple RGB cameras, since they are readily available and relatively cheap. In this context, a number of studies are relevant to our discussion. Bellocchio et al. [2]

*Corresponding author
†The authors contributed equally to the work.

present an olive counting solution that is explicitly trained with weak labels and consistency losses. This work is close to ours for the focus on working on data with minimal labelling, however, it is based on simple direct fruit counting, which can lead to huge errors in cases where self occlusion is typical. While many early works [12, 22], and some recent ones [15], use handcrafted features, most of the recent approaches use representation learning [1,2,24]. When moving into these kinds of techniques, data availability is a key issue to avoid overfitting. To have a general view of this issue, Koirala et al. [8] present an overview of Deep Learning methods applied to fruit detection, pointing out the critical role of data availability, and recommending the use of public data and benchmarks to compare results.

When looking specifically at robotics in agriculture, there are numerous contributions, both in single and multi robot scenarios. Halstead et al. [5] present one of such systems where they detect red peppers with a camera mounted on an AGV. The authors also estimate ripeness by considering ripeness stages as different classes. Among the robotic solutions for yield estimation, there are a few that use multi robot setup, in particular mixing different types of robots, such as Unmanned Ground Vehicles (UGVs) and Unmanned Aerial Vehicles (UAVs) as in [16]. A more recent example of this kind of system is given in [14] by Pretto et al., where the multi robot approach is explored in depth together with the use of multi spectral cameras to detect and monitor both crop and weeds. Since in these works data collection is an issue, [4] extends data augmentation by using GANs. In the specific case of wine and table grape applications, an early approach to grape detection for yield estimation has been presented by Nuske et al. [12]. In their work, a robotic solution able to work at night with controlled lighting is presented. However, most of the early approaches to the detection of grapes are based on handcrafted features and geometrical considerations, as for example in the work by Skrabanek and Mejerik in [22]. Here the authors use HOG descriptors together with a Support Vector Machine to build a white wine grape detector. An approach that builds on these early results and data is the one presented by Pérez-Zavala et al. [15]. The authors use again handcrafted features (HOG, FRST and LBP) to feed an SVM based detector, and use geometrical considerations to separate self-occluding grape bunches that show some robustness to color and illumination variability. The yield estimation task is then a result of the computation of the number of berries detected. Another approach to grape yield estimation that is based on geometrical considerations is the one by Liu et al. [11], where the detection is done on the early stage buds that shoot from the branches in an unsupervised fashion only by using Gaussian fitting. All these approaches, while effective, are difficult to adapt to other situations and cultures, since they would require some context specific tuning and adaptation performed by and expert. A more modern approach to table grape detection and segmentation is the work of Santos *et al*. [19], which uses Mask R-CNN [6] trained on a custom dataset of a few hundred images. While the dataset shows low variability, the fact that the test set has the same distribution makes the approach effective. With regard to our work, we decided to use this dataset as a source dataset to demonstrate the effectiveness of our approach.

2.1. Contribution:

Supervised solutions to detection and tracking have been proposed many times and work well, but the generality of these solutions is naturally impaired by the limited data each solution is trained on. Even if for the majority of the most common fruits a supervised solution with a relative amount of labelled data exists (*e.g*. apples [1], grapes [19], tomatoes [10]), every time we want to train a new detection or segmentation network for the same fruit in a different field, there is considerable covariate shift due to different sensor and environmental characteristics, illumination conditions and fruit intra-species variability. We propose to tackle this problem by using a pseudo-label generation system based on existing source data and unlabelled data for target field that can be used to train detection and segmentation algorithms fine-tuned on the target dataset. We do this using only an input as simple as a single video of the vineyard and automatically extract both bounding boxes and segmentation masks. We work on this problem using the case of table grapes, for which there is some labelled data available (the wine grape dataset from Santos et al. [19]), but not enough to work on different varieties (*e.g*. Pizzutello instead of Cabernet), cultivated with slightly different techniques (*canopy* structure instead of standard trellis structure). With this in mind, the specific pseudo labelling strategies we propose are of two kinds:

- Automatic generation of bounding boxes for objects contained in consecutive video frames, based on a starting estimate and 3D structure geometrical considerations. We show that, by leveraging a simple initial labelling - which could be manual or automatic - and information from feature matching and structure from motion, we are able to generate new labelled data that greatly increase the performance of the detector.

- Pseudo mask generation for instance segmentation: we show how, starting from a simple bounding box - which could be the one automatically generated in the previous step - it is possible to use a segmentation network together with a refining strategy to generate new mask labels.

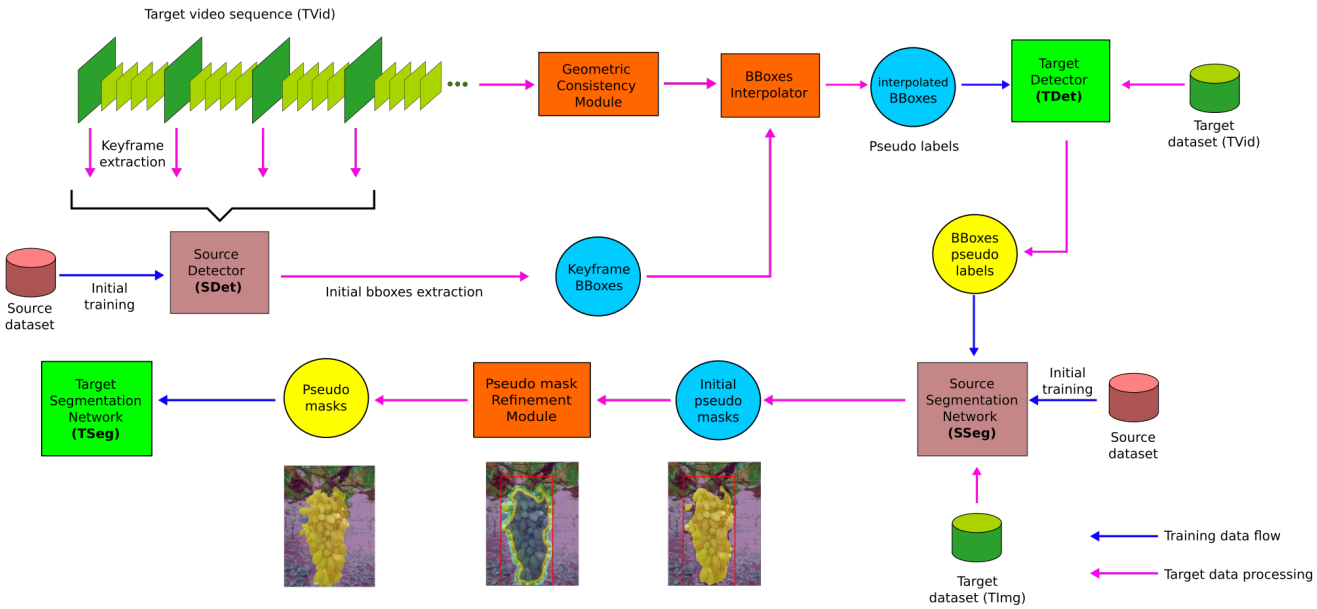

Figure 1. This figure shows the complete system architecture. The inputs are a source dataset and a video collected on the field by the robot or a farmer. The SDet and SSeg are the initial detection and segmentation networks trained only on the source dataset. All the intermediate computing blocks are depicted in orange, while the intermediate outputs are in blue circles. Both the pseudo bounding boxes and pseudo masks produced are depicted in yellow, while the detection and segmentation networks trained on these new labels (TDet and TSeg) are depicted in green.

3. System Architecture

In this Section, we explain the general architecture of the system, the implemented algorithms, and the data used. Section 3.1 describes the global system architecture and how the different components are interconnected. Section 3.2 briefly illustrates the datasets, while the following Sections go into more detail on each subsystem.

3.1. System Overview

Figure (1) presents an overview of the proposed system. The inputs are a source dataset available from a similar task and a video collected on the target field. These data sources are used to train the initial detection and segmentation models, namely the Source Detector Network (SDet) and the Source Segmentation Network (SSeg). An initial prediction is done on some keyframes to obtain bounding boxes using a high confidence threshold to avoid false positives. For simplicity, the keyframes are selected starting form the first frame and then skipping two frames until the next keyframe. The predictions on the remaining frames are made through a Geometric Consistency block (GC block), exploiting feature extraction and association. The geometric information is used to interpolate the position of the bounding boxes on the remaining frames, generating new bounding boxes (pseudo labels) that are used to re-train the detector (TDet) in a more effective way for the target set, using only automatically annotated data. TDet can then be used to refine the bounding box labels of the input video.

The lower branch generates pseudo masks to train an instance segmentation network. As mentioned earlier, SSeg is trained only on the source data, and it is not able to produce high-quality segmentation masks on the Target Data. The mask estimates can be greatly improved by providing the network with additional information. This is done by specifying the bounding box region in which the instance should be segmented. This cue can either come from the upper branch of the architecture or other sources such as manual labelling. The Pseudo-masks Refinement Module further improves the initial pseudo-masks as will be described in Section 3.4.2. Finally, the refined pseudo-masks are used to train the TSeg Network, whose results are shown in Section 4.2.

The joint experiments of the whole system, trained using only predictions given by the detector on images and videos, and tested for instance segmentation performance are given in Section 4.2.

3.2. Dataset

We used two kinds of data for this work. The target data is represented by the Pizzutello table grape variety commonly cultivated in southern Lazio, Italy. The collected images are of two kinds: the first ones are videos recorded using a mid-range cellphone camera that simulates a data col-

lection operation that could be easily performed by a robot or a farmer. We collected videos moving along the vineyard (*i.e*. tangential to the rows), without any requirement on distance from the fruits or height from the ground. In this work, we use HD (1280x720) videos at 10Hz with a total of 1469 frames. We call this target video dataset TVid. In addition, we collected static images of Black Pizzutello. This dataset consists of 134 images of 3000x4000 resolution collected with the same cellphone camera used for the videos. All the images have been labelled for detection (bounding boxes), while a small subset (50 images) has also been labelled for instance segmentation and used for the validation and test of the algorithms described in this Section. We call this image dataset TImg. Together, these datasets (TVid and TImg) constitute our Target Dataset (TD).

As mentioned earlier, we work under the hypothesis that a small amount of labelled data of the same fruit exists, but it has a considerable covariate shift with respect to the target distribution. In this work, our source data is the one presented by Santos *et al*. [19].

3.2.1 Detection and Segmentation Network Architectures

The generation of the pseudo labels is based on pre-trained networks for detection and segmentation (SDet and SSeg). Those are required to automatically provide pseudo labelled data to train new networks that in the end will perform better on the target dataset.

The initial bounding box estimate is computed by SDet and we chose YOLOv5 [17] as the base detector for this stage, for two reasons. The first is the computational speed that allows for real-time detection. While for the pseudo label generation pipeline this is not a requirement, the final detector TDet is meant to be used on a robot on the field in real-time, and with limited computational resources. In addition, a two-stage detector could increase the performances of the final TDet and TSeg networks, but the pipeline gives very good performances even without using the best solution for each of the sub-modules. For the pseudo mask generation, we use the Mask R-CNN [6] architecture, since it allows for having a bounding box input that can become a cue for the segmentation mask as will be described in Section 3.4.1.

3.3. Geometric Consistency Block

The strategy used to generate the pseudo bounding boxes consists in the association of the grape instances across subsequent frames. This is achieved by extracting geometrical correspondences in the frames of a video stream exploiting epipolar geometry. In particular, a Structure from Motion (SfM) algorithm has been used through COLMAP [20, 21], which is a self-contained software that extracts sparse features from the frames and then operates a sequential search through the whole video to match the features extracted. The key point of this system is that the correspondences are used to triangulate unique 3D points by minimizing the 3D to 2D reprojection error. Hence, despite the sparse nature of the problem, the computational cost increases exponentially with the number of frames. To give an idea of the computational cost, to run COLMAP on videos composed of 600 full HD frames on a machine equipped with an Intel-Core i7 3.4 GHz, an Nvidia GTX 950m and 16 GB of RAM requires about 5 hours of computation. While these computational times could be reduced by implementing ad hoc solutions, this is not relevant for our purposes, since the pseudo label generation is a process intended to be run offline.

3.3.1 Bounding Box Interpolation and Pseudo Label Generation

Figure 2 shows how the bounding box interpolation process works to generate the pseudo labels.

The process starts by predicting the bounding boxes at frame i using the detector SDet, and then, using the GC Block, associates the 2D features inside the prediction with those contained in a subsequent frame $i+n$. Due to the camera movement under the vineyard canopy, both the illumination conditions and the position of the grapes in frame $i+n$ will be different from frame i. Consequently, the number of matched features and their position will be different. To draw the new boxes in frame $i+n$, we exploited the hypothesis that the camera is slowly moving and that the motion is tangential to the direction of the vineyard. This allows us to assume that the size (and shape) of the new bounding boxes are the same as the predicted ones, and only their position changed. Therefore, the position of the new bounding box is updated by letting the center of the box coincide with the center of gravity of the features matched in frame $i+n$, as shown in Figure 2a.

3.4. Pseudo Masks generation for Instance Segmentation

After addressing the bounding box detection problem, we focus on some field operations that need a more precise detection, *e.g*. quantitative yield estimations, or harvesting. Instance segmentation is better suited to address these problems, but it generally requires pixel-level labels. Ideally, we would like to have pixel-perfect masks to train a segmentation network, but Bellocchio et al. [2] demonstrated that this is not necessary, since a minimal labelling signal (*e.g*. presence or absence of an object in an image) is sufficient for the task network to learn some representations that are similar to masks of the object of interest. In this case, we leverage both an external cue (the bounding box) and the segmentation network itself to produce pseudo masks that

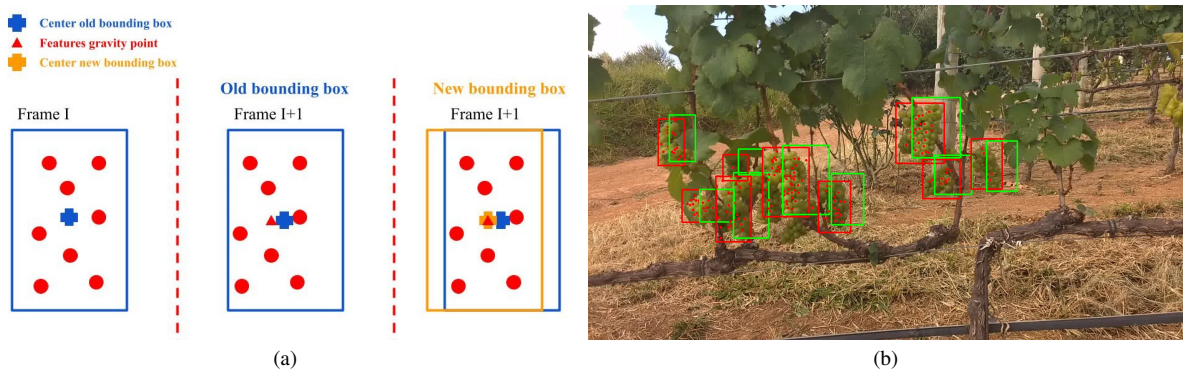

Figure 2. The bounding box interpolation process. a) Shows the updating principle: the bounding box (blue) predicted in frame i is moved to frame $i+n$; while the size remains the same, the position of the new bounding box (orange) is updated by computing the new center of gravity of the features extracted and making it coincide with the box center. b) Pseudo-labels generated by means of the SfM algorithm: the green boxes are the predictions produced by SDet at frame i transposed in the current one $(i+n)$, while the red boxes are interpolated ones according to the features matched (represented as red points).

are able to greatly increase the TSeg performances. This is achieved by giving the external bounding box to the Mask R-CNN network at inference time, instead of using the one produced by its own detector. In addition, following the example of [7], we add the information of an external algorithm (Grabcut) to refine the label. This mitigates the confirmation bias that TSeg could suffer from when learning from its own generated masks.

3.4.1 Pretrained Segmentation Networks

For our SSeg network, we decided to use Mask R-CNN [6] instead of a specific segmentation network architecture, such as U-Net, since in the first case is possible to rewire the segmentation sub-network in order to use external bounding boxes as a cue. In particular, Mask R-CNN is wired differently at inference time from training time, since the bounding boxes predicted by the detection head are directly fed to the mask head. We leverage this feature to integrate the positional information coming from ground truth or estimated bounding boxes. As we mentioned, bounding box labelling is much cheaper than segmentation labelling. In addition, it is possible to automatically produce these labels by using a system such as the one described before. By doing so, we use the bounding boxes as a sort of attention mechanism, guiding the network to where each instance is actually present. An example of the difference of pseudo masks produced by standard Mask R-CNN and the rewired version is given in Figure 3

3.4.2 Pseudo Mask Refining Block

To refine the pseudo masks, an external source of information is used. Some earlier works explored this aspect, such as [7]. Our approach refines the initial masks by using simple computer vision techniques that work on different principles from the convolutional filters contained in SSeg. In particular, our strategy relies mainly on the Grabcut algorithm, an iterative segmentation technique introduced in [18]. It represents the image as a graph where foreground and background pixels are modeled as Gaussian Mixture Models and have to be separated iteratively by cuts to the graph edges. We used the OpenCV [3] implementation where it is possible to initialize the algorithm with the pseudo mask defining four pixel categories, *i.e.* sure foreground, sure background, probable foreground and probable background. The pseudo mask is used as probable foreground. Dilation is applied to the pseudo mask for a number of iterations proportional to the smallest dimension of the reference bounding box to obtain the probable background. Erosion is applied for the same number of iterations to obtain the sure foreground, while the rest is set to sure background. A sample of the effects of Grabcut is shown in Figure 4.

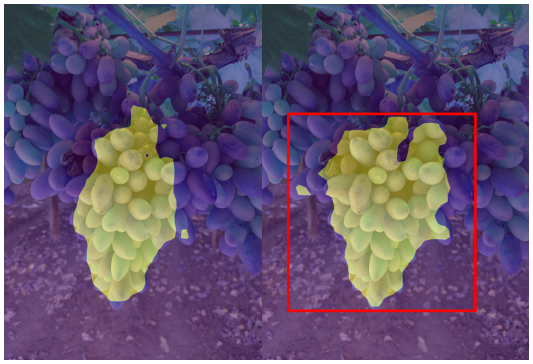

Figure 3. Pseudo-masks generated without (left) and with (right) the bounding box attention mechanism for the same grape cluster.

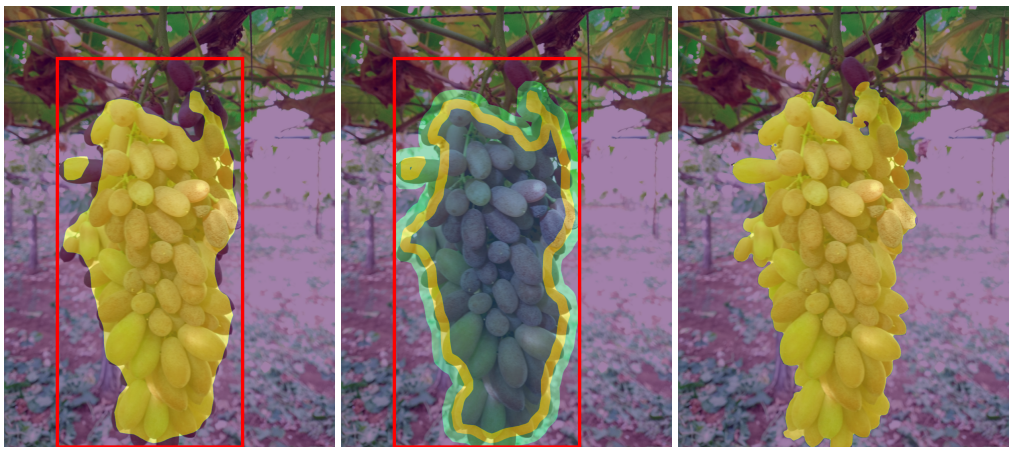

Figure 4. Examples of image refined with Grabcut. The color of the overlay defines the pixel as sure foreground (blue), probable foreground (yellow), probable background (green) or sure background (purple).

In Section 4.2 we show how this refinement method performs compared to the baseline.

4. Experimental Results

4.0.1 Training details

Detector networks: Training were performed on the Nvidia DGX-1 Station since it offers an appropriate computational power. All the training runs had 300 epochs with a batch size of 4 and the patience parameter for early stopping set at 30 epochs. The learning rate (lr) strategy used was "one cycle" [23], with initial $lr = 0.01$ and final $lr = 0.001$. The optimizer is SGD, with momentum 0.937 and weight decay 5×10^{-4}. The training required about 3 hours. All the detection models were pre-trained on the MS COCO dataset [9] and then fine-tuned on the source and target datasets. In order for the detector to generalize towards different scenarios, the 242 training images of the source set provided by [19] were augmented using random crop, random contrast, Gaussian blur, Gaussian noise and horizontal flip. During the experiments, the augmentations were applied offline randomly four times, generating 726 augmented images.

Segmentation networks: The implementation of Mask R-CNN we chose is Detectron2 [25], using ResNet 101 as backbone network. Again, the experiments were performed on the NVidia DGX cluster. The training started from the MS COCO weights, then was fine-tuned on the source and target dataset. For all the training, common data augmentation was performed, by applying Gaussian blur, Gaussian noise, random changes in brightness and contrast, pixel dropout, random flip, and random crop. In addition, the trainings were executed using a learning rate of 0.001, weight decay of 0.0001 and a momentum of 0.9. Each training proceeded for a maximum of 100 epochs, but early stopping was used while monitoring the segmentation AP on the validation set of the table grape dataset, with a patience value of 20.

4.0.2 Covariate shift experiments

To appreciate the covariate shift between the source dataset WGISD and the target dataset, we tested both SDet and SSeg on the source and target dataset. Table 1 shows a comparison of the results for the detection tasks obtained by SDet on the WGISD test split and on our TImg, using the MS COCO metrics [9]. The results draw attention to a severe case of covariate shift between the two datasets, especially in the spatial distribution of the instances present. This is shown by the fact that the precision remains almost the same, while the recall decreases.

Dataset	$Precision$	$Recall$	$mAP_{0.5}$	$mAP_{0.5:0.95}$
WGISD	89.10	83.18	89.68	62.50
T_{img}	83.20	55.60	68.30	42.70

Table 1. Results obtained by the detector trained using only the source dataset, on WGISD dataset and T_{img}.

In the same way, we give an idea of the initial performance gap of SSeg in Table 2. We performed data augmentation on the WGISD dataset, in particular crop and resize to mitigate the difference in scale with the TD, nonetheless in all metrics there is a marked difference in performance.

4.1. Pseudo bounding boxes generation experiments

In this section we describe the results of using the generated bounding boxes to train TDet. We performed preliminary experiments to identify which YOLOv5 variant to use, and the results showed that the models with a large number of parameters offer a minimal performance increase on

Dataset	Task	AP	AP_{50}	AP_{75}
WGISD	Detection	53.40	87.02	57.36
	Segmentation	53.60	89.44	55.41
TSeg	Detection	32.65	60.40	30.37
	Segmentation	32.88	65.40	34.77

Table 2. Evaluation of a Mask R-CNN model on the masked test set of the WGISD (27 images) and TSeg dataset (20 images) using some of the COCO metrics.

detection, compared with the lightweight versions. This is due to the small quantity of training data provided by the source dataset, reinforcing the need for a semi-supervised approach to labelling.

Table 3 shows the difference in performance on the target set (TImg) between the detector trained only on the source data (SDet) and on the pseudo labels generated from the videos (TVid). It is possible to see that the $mAP_{0.5}$ increased by **8%** even though the frames of the videos have a different distribution compared with the target images, due to the different process followed to collect them.

Table 3. Comparison of the detector trained only with source data (SDet) and with the pseudo labels generated from the videos (TDet) and tested on the target data (TImg).

Model	*Precision*	*Recall*	mAP@0.5	mAP@0.5 : 0.95
SDet	0.90	0.56	0.69	0.46
TDet	0.98	0.68	0.77	0.47

A qualitative example of how the detection improves using the pseudo labels mechanism is given in Figure 5, where the same image is shown but with the detection performed by SDet (on the right), and by TDet (on the left) which is trained using the pseudo labels. It is possible to see that not only the bounding boxes are generally tighter around the instances but also that more grapes are detected, meaning that the proposed method improves both precision and generalization.

Since TImg and TVid does not have the same distribution, we also applied the TDet on the test data from TVid. The results are shown in Table 4, where it is possible to see that the network trained with the pseudo labels gained **10%** in $mAP_{0.5}$ compared to the one trained without them. In this case the increase is higher due to the minimal covariate shift between TVid test and training data.

4.2. Pseudo masks generation experiments

The first experiment presented compares SSeg as our baseline with the TSeg, which was trained on both the source and the static images of the target data (TImg) labelled with pseudo masks. Table 5 shows that the additional pseudo masks are able to considerably improve the perfor-

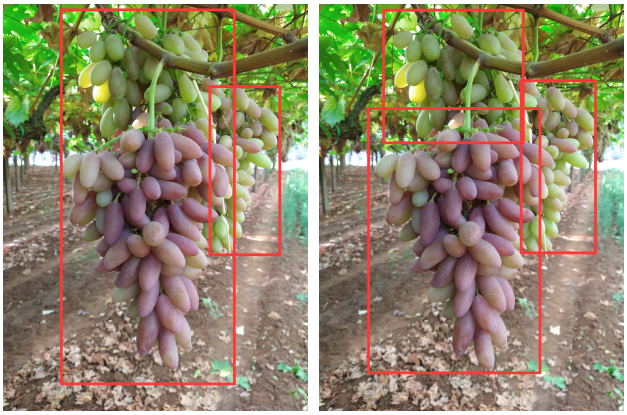

Figure 5. Examples of how the detection improves on the same image when the model is trained without the pseudo labels (left), and with the pseudo labels (right).

Table 4. Comparison of the detector trained only with source data (SDet) and with the pseudo labels generated from the videos (TDet) and tested on the target data (TVid).

Model	*Precision*	*Recall*	mAP@0.5	mAP@0.5 : 0.95
SDet	0.62	0.59	0.55	0.21
TDet	0.74	0.60	0.65	0.23

mance on the TD in terms of AP, with an improvement of over 50% on the baseline performance. In the same table we show the results obtained by TSeg trained with and without the Refining Block. The additional refinement increases the $mAP_{0.5:0.95}$ by 1.13% and the the $mAP_{0.75}$ by 4.58% with respect to TSeg trained without refinement, but decreases in $mAP_{0.75}$, showing that the refinement process is more effective at higher IoU levels, *i.e.* when the initial pseudo mask is well centred on the object.

Table 5. Comparison of the results obtained by TSeg (Mask R-CNN) on the TImg test set, trained with different datasets and pseudo-mask processing methods.

Model	$mAP_{0.5:0.95}$	$mAP_{0.5}$	$mAP_{0.75}$
Baseline	32.88	65.40	34.77
TSeg (w/o Refinement)	48.43	83.06	53.12
TSeg	49.56	81.03	57.70

The second experiment tests the effectiveness of the pseudo mask generation process when the target samples come from the videos of TVid and the labels are the pseudo bounding boxes generated by TDet, as described in Section 3.1. The test data for this experiment is the TImg test set, so the training and test ditributions, although being target data, are different. Table 6 again shows the comparison of TSeg with and without the Refining Block. Despite the fact that the video frames present many differences with respect to the target dataset, the TSeg still manages to increase the

performance by **42%** with respect to the baseline. In this case the Refinement block gives only a minimal advantage. From the values of $mAP_{0.50}$ and $mAP_{0.75}$ we deduce that the increase is due mainly to the IoU higher than 0.75.

Table 6. Comparison of the results obtained by Mask R-CNN (TSeg) on the target dataset test set, trained with the bounding boxes produced by the refined YOLO (TDet).

Model	$mAP_{0.5:0.95}$	$mAP_{0.5}$	$mAP_{0.75}$
Baseline	32.88	65.40	34.77
TSeg (w/o Refinement)	45.99	78.30	52.59
TSeg	46.66	77.38	52.22

5. Conclusions

We presented a semi-supervised pipeline to generate labelled data for detection and segmentation starting from a simple video input and leveraging geometrical considerations and the availability some source data with covariate shift with respect to the target data. We tested this method on the case of a table grape vineyard, where the source dataset is wine grape and presents considerable covariate shift with respect to the table grape. The experiments showed that the proposed system is able to remarkably increase the performances of the detection and segmentation networks on the target dataset. We stress here that we used only one video to produce the pseudo labels, but the number of videos collected could be increased to improve the accuracy of the detector. How this system can be improved in this sense will be the object of future work.

Aknowledgments: This work has been supported by the European Commission under the grant agreement number 101016906 – Project CANOPIES

References

[1] Suchet Bargoti and James P Underwood. Image segmentation for fruit detection and yield estimation in apple orchards. *Journal of Field Robotics*, 34(6):1039–1060, 2017. 2

[2] Enrico Bellocchio, Thomas A. Ciarfuglia, Gabriele Costante, and Paolo Valigi. Weakly supervised fruit counting for yield estimation using spatial consistency. *IEEE Robotics and Automation Letters*, 4(3):2348–2355, 2019. 1, 2, 4

[3] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000. 5

[4] Mulham Fawakherji, Ciro Potena, Ibis Prevedello, Alberto Pretto, Domenico D. Bloisi, and Daniele Nardi. Data augmentation using gans for crop/weed segmentation in precision farming. In *2020 IEEE Conference on Control Technology and Applications (CCTA)*, pages 279–284, 2020. 2

[5] Michael Halstead, Christopher McCool, Simon Denman, Tristan Perez, and Clinton Fookes. Fruit quantity and ripeness estimation using a robotic vision system. *IEEE Robotics and Automation Letters*, 3(4):2995–3002, 2018. 1, 2

[6] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 2, 4, 5

[7] Anna Khoreva, Rodrigo Benenson, Jan Hosang, Matthias Hein, and Bernt Schiele. Simple does it: Weakly supervised instance and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 5

[8] Anand Koirala, Kerry B. Walsh, Zhenglin Wang, and Cheryl McCarthy. Deep learning – method overview and review of use for fruit detection and yield estimation. *Computers and Electronics in Agriculture*, 162:219–234, 2019. 2

[9] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 6

[10] Guoxu Liu, Joseph Christian Nouaze, Philippe Lyonel Touko Mbouembe, and Jae Ho Kim. Yolo-tomato: A robust algorithm for tomato detection based on yolov3. *Sensors*, 20(7), 2020. 2

[11] Scarlett Liu, Steve Cossell, Julie Tang, Gregory Dunn, and Mark Whitty. A computer vision system for early stage grape yield estimation based on shoot detection. *Computers and Electronics in Agriculture*, 137:88–101, 2017. 2

[12] Stephen Nuske, Kyle Wilshusen, Supreeth Achar, Luke Yoder, Srinivasa Narasimhan, and Sanjiv Singh. Automated visual yield estimation in vineyards. *Journal of Field Robotics*, 31(5):837–860, 2014. 2

[13] C. Potena, D. Nardi, and A. Pretto. Fast and accurate crop and weed identification with summarized train sets for precision agriculture. In *Proc. of the 14th International Conference on Intelligent Autonomous Systems (IAS-14)*, 2016. 1

[14] Alberto Pretto, Stephanie Aravecchia, Wolfram Burgard, Nived Chebrolu, Christian Dornhege, Tillmann Falck, Freya Veronika Fleckenstein, Alessandra Fontenla, Marco Imperoli, Raghav Khanna, Frank Liebisch, Philipp Lottes, Andres Milioto, Daniele Nardi, Sandro Nardi, Johannes Pfeifer, Marija Popovic, Ciro Potena, Cedric Pradalier, Elisa Rothacker-Feder, Inkyu Sa, Alexander Schaefer,

Roland Siegwart, Cyrill Stachniss, Achim Walter, Wera Winterhalter, Xiaolong Wu, and Juan Nieto. Building an aerial–ground robotics system for precision farming: An adaptable solution. *IEEE Robotics Automation Magazine*, 28(3):29–49, 2021. 2

[15] Rodrigo Pérez-Zavala, Miguel Torres-Torriti, Fernando Auat Cheein, and Giancarlo Troni. A pattern recognition strategy for visual grape bunch detection in vineyards. *Computers and Electronics in Agriculture*, 151:136–149, 2018. 2

[16] Maryam Rahnemoonfar and Clay Sheppard. Real-time yield estimation based on deep learning. In J. Alex Thomasson, Mac McKee, and Robert J. Moorhead, editors, *Autonomous Air and Ground Sensing Systems for Agricultural Optimization and Phenotyping II*, volume 10218, pages 59 – 65. International Society for Optics and Photonics, SPIE, 2017. 2

[17] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement, 2018. 4

[18] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. " grabcut" interactive foreground extraction using iterated graph cuts. *ACM transactions on graphics (TOG)*, 23(3):309–314, 2004. 5

[19] Thiago T. Santos, Leonardo L. de Souza, Andreza A. dos Santos, and Sandra Avila. Grape detection, segmentation, and tracking using deep neural networks and three-dimensional association. *Computers and Electronics in Agriculture*, 170:105247, 2020. 1, 2, 4, 6

[20] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 4

[21] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 4

[22] Pavel Skrabanek and Filip Majerík. Simplified version of white wine grape berries detector based on svm and hog features. In Radek Silhavy, Roman Senkerik, Zuzana Kominkova Oplatkova, Petr Silhavy, and Zdenka Prokopova, editors, *Artificial Intelligence Perspectives in Intelligent Systems*, pages 35–45, Cham, 2016. Springer International Publishing. 2

[23] Leslie N. Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates, 2018. 6

[24] Shaohua Wan and Sotirios Goudos. Faster r-cnn for multi-class fruit detection using a robotic vision system. *Computer Networks*, 168:107036, 2020. 2

[25] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 6