Weakly Supervised Fruit Counting for Yield Estimation using Spatial Consistency

Enrico Bellocchio†, Thomas A. Ciarfuglia†, Gabriele Costante†, and Paolo Valigi

***Abstract*—Fruit counting is a fundamental component for yield estimation applications. Most of the existing approaches address this problem by relying on fruit models (*i.e.,* by using object detectors) or by explicitly learning to count. Despite the impressive results achieved by these approaches, all of them need strong supervision information during the training phase. In agricultural applications, manual labelling may require a huge effort or, in some cases, it could be impossible to acquire fine-grained ground truth labels. In this paper, we tackle this problem by proposing a weakly supervised framework that learns to count fruits without the need for task-specific supervision labels. In particular, we devise a novel CNN architecture that requires only a simple image level binary classifier to detect whether the image contains instances of the fruits or not and combines this information with image spatial consistency constraints. The result is an architecture that learns to count without task-specific labels (*e.g.,* object bounding boxes or the multiplicity of fruit instances in the image). The experiments on three different varieties of fruits (*i.e.,* olives, almonds and apples) show that our approach reaches performances that are comparable with SotA approaches based on the supervised paradigm.**

***Index Terms*—Agricultural Automation, Computer Vision for Other Robotic Applications, Deep Learning in Robotics and Automation, Robotics in Agriculture and Forestry, Visual Learning.**

I. Introduction

AMONG the multitude of agricultural processes that draw the attention of computer science and robotics researchers, an important role is certainly played by yield estimation. An accurate estimation of the yield of a culture facilitates the farmer in planning for harvesting operations and crop sales. However, the standard practice to yield estimation often relies only on coarse measurements and direct inspection, a practice that has high costs and low accuracy. To address this problem, we have recently witnessed a widespread adoption of camera-equipped robots in agricultural fields. The use of automated vehicles combined with SotA computer vision techniques [1], [2] achieves the benefits of both reducing costs and increasing yield estimation accuracy.

While it is possible to reduce the cost and density of yield sampling with the use of autonomous robots to collect images of the orchard [3], [4], [5], the actual fruit counting still remains a challenging task. This is mainly related to the inherent difficulty of extracting high level concepts (*i.e.,* fruits) from raw images due to image anomalies, background clutter, scale variations and occlusions, to name a few. While the detection of some crop varieties is easier due to their shapes or colors (*e.g.,* tomatoes or apples) [6], [7], there are fruit species that can be very difficult to distinguish from background or foliage, such as olives or almonds.

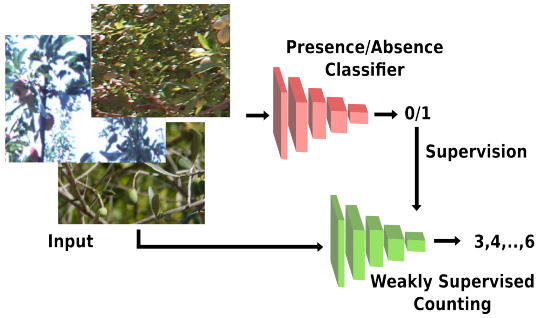


Fig. 1: Our weakly supervised counting network. The weak supervision comes in the form of a simple binary presence classifier, that requires less data and much simpler labelling to train than any fully supervised method.

In order to build more reliable systems to achieve fruit counting, many vision-based strategies have been proposed by the research community. Most of the existing approaches exploit three fundamental paradigms: 1) counting by using specific object detectors [4], [8], [2], 2) counting by estimating density maps [9] or 3) counting by explicitly training an object counter [6], [7]. Despite the impressive results presented in these works, to be trained, all of the proposed strategies require detailed supervision, in the form of object bounding boxes, density maps or instance multiplicity. This procedure is very burdensome and error prone and, in many agricultural scenarios, it may be impossible to acquire a sufficient number of labelled samples to achieve consistent performance that are robust to image noise or other forms of covariate shift.

Driven by the previous considerations, in this work, we aim to remove the need for explicit instance or density labelling. This is achieved by proposing a weakly supervised deep architecture that relies only on an image level binary classifier, *i.e.,* the sole supervision label that is needed is whether the image contains instances of the fruit or not (see Figure 1). Since this information by itself is not sufficient to allow the network to learn to count, we propose a novel objective function that imposes consistency between the image level classifier predictions computed at different spatial locations and scales. Our approach is related to what is done in the field of representation learning [10] where counting tasks are used to learn general image representation, but by the introduction of the consistency loss on the weak binary classifier we

Enrico Bellocchio, Gabriele Costante, Thomas A. Ciarfuglia and Paolo Valigi are with the Department of Engineering, University of Perugia, Perugia, Italy {enrico.bellocchio, gabriele.costante, thomas,ciarfuglia, paolo.valigi}@unipg.it

† The authors equally contributed to the work.

manage to force the network to learn the actual object we are interested in. To the best of our knowledge, this is the first attempt to devise a weakly supervised object counting strategy for yield estimation applications. The experiments demonstrate that the proposed approach reaches performance that are comparable to fully supervised baselines with respect to three different fruit species, namely apples, almonds and olives.

II. Related Work

The introduction of computer vision techniques in agriculture, often referred to as agrovision [1], has considerably grown in recent decades. In this context, yield estimation has gained particular attention and the latest works have shown impressive results in different scenarios, such as apple orchards [5], vineyards [3] and mango orchards [11], [4].

Although in the majority of the agrovision researches the problem of yield estimation is devised as a fruit detection task, it can be framed as a more generic object counting one and addressed either indirectly by using object detectors or explicitly with architectures that learn to count. The latter usually set up a regression problem to directly infer the number of object instances in the image or to estimate density maps.

Many effective approaches for object counting have been proposed in other contexts, such as crowd counting [12], [13] or biological cell counting [9], [14]. Thus, the following sections cover the SotA works that focus on the more general object counting problem, with particular attention to agrovision applications, and give an outline of our contributions.

1) Counting by detection: The earliest paradigm adopted to estimate the number of object instances in an image is probably counting by detection. The idea behind it is straightforward: counting is performed by summing the object instances found by a class-specific object detector. This strategy has been initially developed and widely adopted in crowd counting applications, where a people detector [15] is used to count the number of people in a scene. Detection is performed both monolithically [16], [17], [18] and with part-based approaches [19], [20], [21]. The latter were introduced in an attempt to address high density crowd scenes, where monolithic approaches (*i.e.,* that rely on full body detectors) fail due to the high number of occlusions.

In the context of agrovision, fruit detectors have been developed with a wide variety of strategies, from classification by using low-level keypoint extractions [3], [22] to segmentation and detection [23], [5], [24]

The advent of Convolutional Neural Networks (CNNs) has considerably improved the robustness of detectors. CNN-based detectors have been successfully used in the context of supervised fruit counting for yield estimation [2]. The work presented in [4], uses Faster R-CNN [25] to detect fruits. The detection module is integrated with a navigation system and a LiDAR component to associate fruits to corresponding trees to perform yield estimation. Similarly, in [8] the authors use Faster R-CNN to detect fruits in apple, almond and mango orchard data. [26] propose a multi-modal extension of Faster R-CNN to combine colour (RGB) and Near-Infrared (NIR) information to achieve fruit detection. The approach introduced by [27] has two components: a fruit detection and ripeness estimation system inspired by [26] and a tracking by detection system to compute the quantity of fruits.

The limitation of these approaches is the cost of training the detector and, possibly, the lack of robustness to covariate shifts of the detector itself. These considerations have driven the researchers towards approaches that explicitly learn to count object instances in an image.

2) Counting by regression: Counting by regression aims to directly map visual features extracted globally or locally from image patches to the count of object instances. Hence, the model explicitly learns to count, which results in more robust and precise estimations.

Similarly to counting by detection strategies, regression based approaches have been deeply studied in crowd counting applications [13], achieving very promising results. In agrovision, [6] and [7] are among the most significant works that design solutions to the yield estimation problem by following the supervised regression paradigm. [6] proposes a modified version of Inception-ResNet CNNs to output the fruit count. Furthermore, they show that the CNN can be trained by using synthetic images, easing the labelling effort required. In [7] the authors devise a count architecture composed of three fundamental parts. The first is responsible for detecting candidate fruit blobs using a fully convolutional network. The second, trained as a liner regression problem, uses a CNN to count the number of fruits in each blob.

3) Counting by density estimation: Regressing on the global count provides an improved robustness with respect to occlusions and background clutter, but spatial information is inevitably discarded. Recently, different image descriptors that take into account global or local spatial information have been proposed in the context of image retrieval and classification [28], [29], [30], [31], [32], [33]. However, these descriptors are not specifically designed for object counting applications that, on the other hand, could benefit from approaches able to learn features that are jointly spatial-aware and well-suited for the counting task. Thus, density map estimation approaches have been proposed to incorporate the spatial correlation during the learning process. This approach is introduced by [9]. By learning object density maps, the benefit obtained is twofold: they avoid learning the more complex detection task while keeping spatial information. This approach has been extended by [34], [35], [14] and further improved by more recent works based on CNNs, such as [36], [37] and [38]. While these approaches have been successfully applied in the context of crowd or biological cell counting, to the best of our knowledge no works have been proposed for fruit counting applications.

4) Unsupervised and weakly supervised counting: Non-supervised approaches for fruit counting and yield estimation are an unexplored area. In the context of crowd counting, the work proposed by [39] and [40] avoid supervised learning by exploiting motion dependencies across sequential frames in a video stream. In [39] the Kanade-Lucas-Tomasi (KLT) tracker is used to track features. Afterwards, clustering is performed to estimate the number of people in the scene. Similarly, in [40] low-level features are tracked and clustered

with Bayesian clustering techniques. However, although these strategies follow the unsupervised paradigm, they rely on the uniformity of motion fields and work only with contiguous sequences of image frames. This assumption are often too restrictive in yield estimation applications.

A very recent and promising strategy has been proposed by [41] in the context of instance segmentation application. They devise a weakly supervised approach that exploits peak response maps to identify class segments in the image. It is then straightforward to perform object counting by simply enumerating the estimated instance segments. However, this approach is not specifically designed for object counting tasks and, thus, as shown in the experimental section, it achieves lower performance when compared to ours.

5) Contribution: To the best of our knowledge, in the literature there are currently no works that propose a weakly supervised approach for still image object counting tasks. All the previous approaches rely on bounding box labels [4], [8], [27], object instance count labels [6], [7] or density maps [13] as supervision signals.

In this work, we give a new twist to the fruit counting problem by removing the strong assumption of having labelled data samples to train the object count models. The only requirement is a simple image level binary classifier that predicts whether the image contains instances of the fruit or not, *i.e.,* there is no need to provide the number of objects in the image or bounding box labels or density maps during training. To achieve this, we build an objective function that combines the classifier output at different locations and scales of the image with a spatial consistency term. This forces the model to learn to count without the need for task specific supervision signals.

An extensive set of experiments on different fruit varieties (olives, almonds and apples) shows how the proposed weakly supervised approach is able to achieve a performance similar to its fully supervised counterparts and to SotA approaches. Furthermore, two new datasets containing images of olive and almond groves with ground truth information are released for comparisons and further researches.

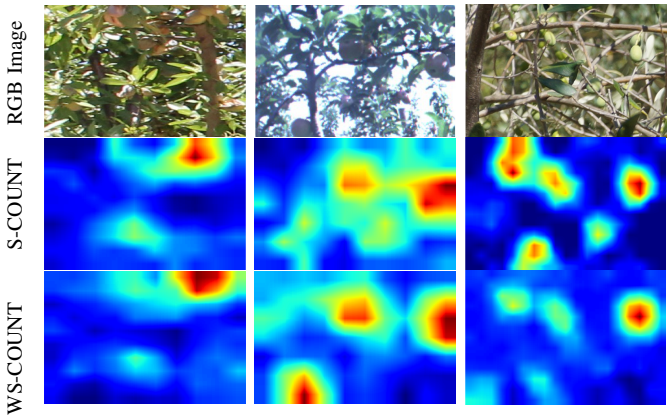
III. Proposed Approach

This section describes the proposed weakly supervised framework for fruit counting. First, a fully supervised architecture that learns to regress the total fruit count in an image by using task specific labels (*i.e.,* the multiplicity of fruit instances) is introduced. The network, referred to as S-COUNT (supervised counting) is trained in an end-to-end fashion and acts as one of the supervised baselines.

Afterwards, Section III-B describes how S-COUNT is extended to exploit the weakly supervised paradigm. This network is referred to as WS-COUNT.

A. End-to-End Supervised Counting

The S-COUNT network is trained as a fully supervised regression model. It directly outputs a single number that represent the total fruit count for a given image. Each training instance is composed by an image-label pair $(\mathcal{I}, y)$, where $\mathcal{I}$ is an RGB image and $y \in \mathbb{R}$ represents the ground truth number of the object instances in the image.



(a) Almonds (b) Apples (c) Olives

Fig. 2: Response maps of the S-COUNT and WS-COUNT architectures. The first row depicts the input images, while the second and the third show one of the eight output maps of the 1×1 convolutional layer of the S-COUNT and WS-COUNT architecture, respectively. It can be observed that the networks assign higher value to image locations with high probability of fruits.

The feature extraction part of S-COUNT is ResNet101 [42] architecture without the final fully connected layer (see Figure 3). The output of the last 3×3 convolutional layer of ResNet101, which has 2048 feature maps, is fed into a 1×1 convolution with 8 filters. The purpose of these filters is to learn output response maps, which are then usd by the final fully connected layer to regress the instance count (see Figure 2). We experimentally observed that its response maps have greater values in regions that most likely contain fruit instances. Although S-COUNT is not a classification network, this is expected since, in order to count the number of instances in $\mathcal{I}$, the network needs to be able to somehow locate them before counting. The choice to use eight filters and not a single one provides the network with the capability to learn multiple modalities to represent the object and to improve the overall accuracy. However, as shown in [43], learning too many maps can lead to a performance drop due to overfitting, hence, we do not ask the network to learn more than eight of them.

All the layers in the network, except for the output one, have ReLU activations and batch normalization layers to ease the optimization during training. The loss function that is optimized is a standard Mean Squared Error (MSE) loss.

B. Weakly Supervised Counting

As stated in the previous section, the S-COUNT network relies on task-specific supervision labels $y^{(i)}$ that encode the fruit instance multiplicity in each image. Ideally, we want to remove this label and have a network that is able to learn from the images what and how to count. Stated like this the problem is ill-posed, because the network should have at least a slight hint about what to learn. For this reason, a more complex counting network is introduced. This network is a multi-branch counting CNN (MBC-CNN) that operates on different image sub-windows at different levels. More precisely, it works on three scales, the whole image, the image divided into

quadrants, and the image divided into 16 parts (Figure 3). For each level the network is applied to each tile and tries to regress the number of fruits in the tile. Since the correct number of fruits in the image is not used as a supervisory signal, we impose the constraint that at each scale the total count regressed on the corresponding tiles must be consistent with the total count of other levels. As shown in [10], this is enough to force the network to learn to count something. However, using only this approach, it is not possible to be sure that the network will learn to count fruits.

To overcome this, the weakly supervised framework is introduced. Consider a training set $\mathcal{T} = \{(\mathcal{I}^{(0)}, c^{(0)}), (\mathcal{I}^{(1)}, c^{(1)}), ..., (\mathcal{I}^{(N)}, c^{(N)})\}$ where N is the total number of training samples and $c^{(i)} \in \{0, 1\}$ indicates whether the image contains instances of the fruit or not. The information encoded by $c^{(i)}$ is clearly "weaker" than the precise instance count $y^{(i)}$, but it is easier to collect and less prone to human counting errors.

Since these labels cannot be used naively to train a counting network, we introduce an image level binary classifier, which will be referred to as PAC (Presence-Absence Classifier), and use it to train the actual counting network. The key intuition, in addition to the counting consistency, is to force consistency between the output of each counting branch and the prediction of the PAC. If the classifier predicts the presence of object instances, the counter should output a number greater than zero. Conversely, when the absence of fruits is estimated, the count must be zero.

Similarly to [10], WS-COUNT uses consistency checks on counting at different levels to force the network to learn a concept of objectness. However, in [10] there is no constraint on what to count, so the network learns some vague concept of objectness. Instead, our approach exploits the binary classifier to allow the counting network to learn the correct concept. While the supervision given by the PAC classifier is weak and noisy, it is enough, together with the count consistency constraints, to learn the counting task.

The rest of this section provides the details about the PAC and the MBC-CNN networks and describes the objective functions used to train the overall WS-COUNT framework.

1) Presence-Absence Classifier: The Presence-Absence Classifier architecture is similar to the S-COUNT one presented in Section III-A, *i.e.,* it has a ResNet101-based feature extractor followed by a 1×1 convolutional layer with eight response map. Differently from S-COUNT, the response map layer output is not fed into a series of fully connected layers. Instead, the PAC network uses the two-stage pooling layer proposed by [43], which is composed of a class-wise pooling operation with a spatial pooling one. The former combines the eight maps from the last 1×1 convolutional layer into a final one that summarizes their contribution. The latter is used to average the spatial information into a single final value $\hat{c}^i$. This value is then provided as input to a sigmoid function to predict the presence or the absence class associated to the image $\mathcal{I}^{(i)}$. The PAC network is trained with a standard binary cross-entropy (BCE) loss.

2) Multi-Branch Counting CNN: The MBC-CNN operates on sub-windows extracted from the image at three different scales: the first one is the full image while the second and the third are obtained by dividing $\mathcal{I}^{(i)}$ into 4 and 16 non-overlapping sub-windows, respectively. Each sub-window is processed by a CNN-branch with the same {ResNet101→Response Map Layer→Fully Connected Layers} structure as S-COUNT and outputs a single positive real value $o^{(j)}$ that represents the estimation of the count instances in that sub-window. The feature extraction part is shared between the levels, while the fully connected predictor is different for each level. By observing Figure 2 it is clear that, similarly to S-COUNT, the response map layer of MBC-CNN is able to localize fruit instances in the image. To allow the MBC-CNN to learn to count, as introduced in the previous sections, we set up two objective terms: a classifier consistency $\mathcal{L}_{PAC\text{-}C}$ loss and a spatial consistency $\mathcal{L}_{SP\text{-}C}$ loss. The first is responsible for constraining the counts of each branch to be coherent with the presence-absence prediction of the classifier (which is trained separately):

$$\mathcal{L}_{PAC\text{-}C} = -\sum_{i=0}^{N}\sum_{k=0}^{2}\sum_{j=0}^{2^{2k}} \hat{c}^{(i,k,j)} \log(g(o^{(i,k,j)})) + (1 - \hat{c}^{(i,k,j)}) \log(1 - g(o^{(i,k,j)})) \quad (1)$$

where $\hat{c}^{(i,k,j)}$ and $o^{(i,k,j)}$ are the PAC prediction and the estimated count for the sub-window j at level k, respectively, and $g(x) = \sigma(\alpha x - 0.5)$ is a shifted and scaled (by α) sigmoid function that maps $o^{(i,k,j)}$ to $\{0, 1\}$.

The $\mathcal{L}_{SP\text{-}C}$ loss aims to impose coherency among the sum of sub-window counts of the three different levels:

$$\mathcal{L}_{SP\text{-}C} = \sum_{i=0}^{N}\sum_{k=0}^{2}\sum_{l=k+1}^{2} \left|\left| \left(\sum_{j=0}^{2^{2k}} o^{(i,k,j)}\right) - \left(\sum_{m=0}^{2^{2l}} o^{(i,l,m)}\right) \right|\right|_2^2 \quad (2)$$

where i is the index associated with the current training image $\mathcal{I}^{(i)}$, while k and l iterate over the three different scales. $o^{(i,j,k)}$ is the object instance estimate of the counting CNN for the sub-window k at level j of image $\mathcal{I}^{(i)}$ (similarly for $o^{(i,m,l)}$).

The two losses are combined in the final objective function that is optimized during training:

$$\mathcal{L}_{MBC\text{-}CNN} = \mathcal{L}_{PAC\text{-}C} + \beta \cdot \mathcal{L}_{SP\text{-}C} \quad (3)$$

The main hyperparameters are α and β parameters and can be found through cross validation. Since the multi-branch networks produce three estimates (one per scale level) that are slightly different, it is possible to choose the final output value as a specific level count or as the average of the three predictions. By cross-validating this aspect, we found that the average of predictions gives the best performance.

IV. Experiments

A. Datasets

The proposed WS-COUNT framework is evaluated with respect to three different fruit species: apples, almonds and olives (see Figure 2). For apples and almonds we use the datasets provided by [8]. We also gathered and labelled a brand new almond dataset to evaluate the performance under

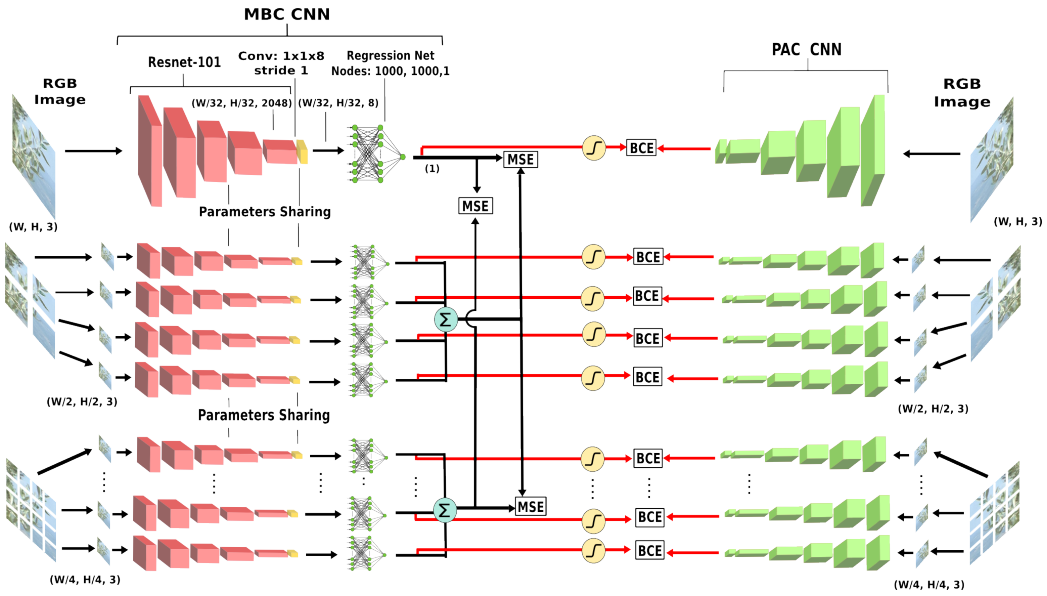


Fig. 3: The figure shows the core architectural components of our WS-COUNT network. MBC-CNN (on the left) processes the input image at three different scales (1, $1/2$ and $1/4$) and outputs the estimated fruit counts for each image tile. To this end, the image tiles are fed into a Resnet-101 feature extractor (red blocks) that ends with a 1×1 convolutional layer (yellow block) with 8 response maps. These maps are then flattened and processed by 3 fully connected layers that output the final count estimate. The feature extractor part shares its parameters across each branch, while the fully connected layers are specific for each image scale. The count estimate of each branch is summed with the other counts at the same scale and the results is forced to be similar with those at the other scales by using a Mean Squared Error (MSE) loss. The weak supervision signal is provided by the PAC (green blocks on the right), that processes the same image sub-blocks of the MBC-CNN network and provides a binary label encoding whether there are fruits in the image block or not. Count consistency is achieved by comparing the PAC outputs with the MBC-CNN count estimates with a Binary Cross Entropy (BCE) loss. To convert the continuous values of the MBC-CNN into $\{0, 1\}$ labels, we fed them into a sigmoid functions.

different conditions and scenarios. Furthermore, since there are currently no olive datasets available in the research community, we also collected and manually labelled a set of images of olive trees. Both the almond and the olive datasets are made available to the community on the accompanying web page.

The apple and the almond datasets from [8] are composed of 1115 RGB images with a resolution of 308×202 pixels and 555 images with 300×300 pixels, respectively. The images in both datasets are provided with manually annotated bounding boxes around the fruits. The count label for each image are directly derived from the bounding box lists.

The novel almond dataset (AlmondISAR) has been collected by flying a DJI Phantom 4 drone over a cultivation of almond trees. First, 17 high resolution (5472×3078 pixels) images depicting the full almond tree canopy are extracted. Afterwards, these images have been cropped to generate 1158 images at 300×300 pixel of resolution. To build the olive dataset (OlivesISAR), 31 high resolution images with 5456×3632 pixels are collected by using a high quality camera. The images capture the full olive tree shape. Three of them are kept apart to run the entire tree shape (see Section IV-D), while the remaining 28 were used to train the count models. Similarly to the almond dataset, the latter ones are processed by randomly picking smaller tiles of 606×403 pixels to obtain a dataset of 1402 images. The bounding box labelling was performed by using [44]. It is important to mention that the bounding box information is only used to train the approach described in [8] (see Section IV-B) for comparison, while our approach needs only the fruit instance counts in the image. All the train-test splits are reported in Tables II.

(a) Almonds (b) Apples (c) Olives

Fig. 4: Sample images of the three fruit species (Almonds (4a), Apples (4b) and Olives (4c)) used in the experiments.

B. Baselines

To prove the effectiveness of our approach, we compare it against six different baselines. As stated before, to the best of our knowledge all the SotA approaches for fruit counting are fully supervised.

As the fully SotA supervised baselines we use the approaches proposed by [8] and [6]. The former is trained on bounding-box instance labels and uses the Faster-RCNN object detector to count the fruit instances in an image. Conversely, the latter takes advantage from a modified Inception-ResNet CNN to learn a model that regresses directly the fruit counts. In the following, these approaches are referred to as RCNNCount and DeepCount, respectively. Furthermore, WS-COUNT is compared against the weakly-supervised strategy proposed by [41] (which will be referred to as PRM), that exploits peek response maps to perform instance-level segmentation. We simply count the detected peaks to estimate the total number of fruits in the image.

In addition to this, we consider also two supervised networks trained in an end-to-end fashion by using the instance counts as supervision signal. The first one is the S-COUNT

architecture presented in Section III-A.

Since WS-COUNT exploits a multi-branch structure, we decided to compare it also against a multi-branch version of S-COUNT (which we named MBS-COUNT) to provide a fair comparison. Each branch of MBS-COUNT, similarly to the MBC-CNN architecture, processes one of the sub-windows and outputs a count estimate. We train it by optimizing the following loss:

$$\mathcal{L}_{MBS\text{-}COUNT} = \mathcal{L}_{MB\text{-}MSE} + \mathcal{L}_{SP\text{-}C} \tag{4}$$

where

$$\mathcal{L}_{MB\text{-}MSE} = \sum_{i=0}^{N} \sum_{k=0}^{2} \left|\left| \left(\sum_{j=0}^{2^{2k}} o^{(i,j,k)} \right) - y^{(i)} \right|\right|_2^2 \tag{5}$$

is the loss that enforces the network to keep the sum of the sub-window count estimates close to the ground truth count label, while $\mathcal{L}_{SP\text{-}C}$ is the spatial-consistency loss defined in (2). It worth noting that, unlike the objective defined for WS-COUNT, in this case we rely on the supervision labels $y^{(i)}$.

Finally, since PAC is used as a source of supervision, we asked ourselves whether simply counting the binary prediction of the presence-absence classifier (PAC) could provide good estimates. For this reason, the binary classifier is applied at the finer scale (16 tiles) and, for each tile, the sum of the fruit detections ($0-1$ signals) is counted. This last baseline helps us to understand whether the WS-COUNT network is able to perform better than the supervisory PAC network.

C. Implementation Details

The implementation of the S-COUNT, MBS-COUNT, PAC and WS-COUNT architectures is based on the Pytorch framework. For each architecture, one model for each fruit dataset is trained. ResNet101 is pretrained on ImageNet [45] and fine-tuned with a smaller learning rate (*i.e.,* 10^{-6}) with respect to the top layers (see Table I for further implementation details).

For the implementation of RCNN-COUNT [8], the Tensorflow implementation of the Faster-RCNN object detector architecture is used. To provide a comparison independent from the feature extraction module, we decide to use the same backbone as in our networks (*i.e.,* ResNet101), instead of VGG16 [46]. The object detection network is initialized on the COCO object detection dataset [47] and then fine-tuned on the fruit sets after performing a net surgery process to reduce the number of classes from 80 (*i.e.,* the COCO number of categories) to 1. Since the authors do not provide the code for DeepFruit, we follow the indications on the paper [6] to implement it. Conversely, we used the code provided by [41] to train their PRM network in our scenarios. We only replaced the ResNet50 feature extractor in their code with the 101 version (the same used in our network) to achieve a fair comparison.

All the aforementioned architectures are optimized by using a Nvidia Tesla K-40 GPU with 12 GB of VRAM. All the newtorks are optimized with SGD, except for DeepCount that is trained with the Adam optimizer as suggested by the authors in [6]. The code and the data are available at the accompanying web page [1].

[1]https://isar.unipg.it/

	Net	Batch	Learning rate	Epochs	Training Time	Test Time (s/img)
OlivesISAR	S-COUNT	16	10^{-4}	100	8h	0.026
	MBS-COUNT	5	10^{-5}	200	65h	0.11
	MB-PAC-Only	16	10^{-2}	20	1h	0.09
	WS-COUNT	3	10^{-5}	200	65h	0.057
	RCNNCount [8]	1	3×10^{-4}	100	12h	0.60
	DeepCount[6]	15	10^{-3}	60	45m	0.021
	PRM[41]	16	10^{-2}	50	80m	1.19
Apples[8]	S-COUNT	64	10^{-4}	100	2h	0.007
	MBS-COUNT	16	10^{-5}	200	10h	0.069
	MB-PAC-Only	64	10^{-2}	20	1h	0.026
	WS-COUNT	16	10^{-5}	200	10h	0.024
	RCNNCount [8]	1	3×10^{-4}	100	12h	0.47
	DeepCount[6]	15	10^{-3}	60	45m	0.005
	PRM[41]	16	10^{-2}	50	40m	0.52
Almonds[8]	S-COUNT	32	10^{-4}	100	2h	0.01
	MBS-COUNT	14	10^{-5}	200	10h	0.073
	MB-PAC-Only	32	10^{-2}	20	1h	0.032
	WS-COUNT	12	10^{-5}	200	10h	0.027
	RCNNCount [8]	1	3×10^{-4}	100	12h	0.44
	DeepCount[6]	15	10^{-3}	60	15m	0.008
	PRM[41]	16	10^{-2}	50	30m	0.083
AlmondsISAR	S-COUNT	32	10^{-4}	100	3h	0.01
	MBS-COUNT	14	10^{-5}	200	12h	0.073
	MB-PAC-Only	32	10^{-2}	20	2h	0.032
	WS-COUNT	12	10^{-5}	200	12h	0.027
	RCNNCount [8]	1	3×10^{-4}	100	13h	0.44
	DeepCount[6]	15	10^{-3}	60	20m	0.008
	PRM[41]	16	10^{-2}	50	40m	0.083

TABLE I: Batch sizes, learning rates, epochs, training and test times of all the networks compared in the experiments. The training times for DeepCount and PRM are considerably lower than WS-COUNT, MBS-COUNT and S-COUNT since our networks need to process the images at three different scales.

D. Results and Discussion

To evaluate the performance of both the baseline methods and our approach, the count estimates are compared with the ground truth value by using the RMSE metric. In order to perform statistical verification of the results, we train 20 models by using different weight initialization seeds for each network. The RMSE of each model on the test set are then used to compute the means and the standard deviations. We start our discussion by commenting the RMSE obtained by each model over the whole test sets for each fruit dataset. The results are shown in the right-most column of Table II.

It can be observed that the best performance are achieved by [8]. This is to be expected, since their model is specifically trained with the most informative labels (bounding boxes on instances), which are more robust to self-occlusions and background clutter. The S-COUNT and MBS-COUNT networks give higher, but still comparable, errors with respect to [8], showing that end to end counting on total number of instances is effective. The most important result is that WS-COUNT, despite being trained in a weakly supervised manner, achieves performances that are close to our supervised baselines (S-COUNT and MBS-COUNT) and better than DeepCount [6]. It is also important to observe that the errors obtained by the MB-PAC-Only baseline are considerably higher than WS-COUNT, which proves that the combination of the classifier consistency and the spatial consistency losses gives the network a better capability to count the fruit instances. Furthermore, WS-COUNT performs significantly better than PRM [41], which proves that, despite both of them follow the weakly supervised paradigm, our approach is better suited for fruit counting.

OlivesISAR					
GT #fruits	0	1-5	6-10	11+	All
(#Train / #Test)	(653/8)	(374/64)	(176/25)	(87/7)	(1298/104)
S-COUNT	0.05±0.04	1.51±0.11	2.39±0.22	4.45±0.36	2.03±0.07
MBS-COUNT	0.51±0.35	1.11±0.27	2.58±0.38	4.35±1.05	1.91±0.21
MB-PAC-Only	0.04±0.02	1.56±0.92	3.00±0.78	7.50±1.58	2.73±0.41
WS-COUNT	0.05±0.02	1.23±0.14	2.90±0.12	6.69±0.33	2.44±0.06
RCNNCount[8]	0.0±0.0	1.15±0.07	2.29±0.13	2.42±0.81	1.57±0.18
DeepCount[6]	0.76±0.34	2.54±0.34	4.47±0.38	9.1±1.06	3.81±0.22
PRM[41]	6.82±5.58	5.71±3.29	4.25±0.62	5.41±2.92	5.95±2.08
Almonds [8]					
GT #fruits	0	1-5	6-10	11-15	All
(#Train / #Test)	(39/9)	(186/36)	(149/23)	(93/20)	(467/88)
S-COUNT	3.12±0.4	2.62±0.19	2.93±0.2	4.93±0.23	3.40±0.09
MBS-COUNT	3.83±0.94	2.48±0.31	2.12±0.46	4.85±0.44	3.24±0.19
MB-PAC-Only	2.32±1.67	1.83±1.67	3.26±0.61	6.16±1.35	3.65±0.63
WS-COUNT	1.59±0.3	1.53±0.24	3.63±0.17	6.07±0.27	3.61±0.12
RCNNCount[8]	2.08±0.29	1.36±0.1	1.48±0.17	2.86±0.44	1.90±0.13
DeepCount[6]	3.14±0.66	3.0±0.51	3.21±0.34	6.88±0.57	4.29±0.17
PRM[41]	0.69±0.23	2.2±0.34	5.82±0.58	9.86±0.54	5.75±0.41
Apples [8]					
GT #fruits	0	1-5	6-10	11-15	All
(#Train / #Test)	(56/8)	(532/59)	(351/40)	(65/4)	(1004/111)
S-COUNT	1.98±0.21	1.56±0.09	2.17±0.1	4.47±0.5	2.00±0.06
MBS-COUNT	2.21±0.54	1.37±0.17	2.02±0.25	4.42±0.66	1.88±0.12
MB-PAC-Only	1.23±3.15	1.77±2.35	2.44±1.18	5.26±1.5	2.22±0.89
WS-COUNT	0.90±0.22	1.76±0.15	2.55±0.18	4.12±0.33	2.03±0.06
RCNNCount[8]	0.61±0.23	1.29±0.04	1.24±0.2	0.70±0.7	1.22±0.12
DeepCount[6]	0.59±0.46	1.78±0.18	2.78±0.27	4.49±0.56	2.3±0.15
PRM[41]	0.14±0.19	2.19±0.09	5.38±0.18	9.35±0.19	4.02±0.11
AlmondsISAR					
GT #fruits	0	1-5	6-10	11+	All
(#Train / #Test)	(528/12)	(420/72)	(93/14)	(16/1)	(1057/99)
S-COUNT	0.36±0.11	1.46±0.07	2.9±0.18	2.91±0.83	1.69±0.06
MBS-COUNT	0.83±0.58	1.55±0.21	2.95±0.72	2.21±1.45	1.8±0.24
MB-PAC-Only	1.31±0.72	1.69±0.24	3.73±0.68	4.42±1.83	2.15±0.19
WS-COUNT	1.47±0.28	1.39±0.11	3.22±0.25	3.46±0.56	1.81±0.03
RCNNCount[8]	0.30±0.07	1.16±0.15	2.17±0.4	1.6±0.8	1.33±0.2
DeepCount[6]	1.56±0.43	2.09±0.27	5.15±0.57	7.84±1.42	2.82±0.12
PRM[41]	3.39±2.39	2.57±0.58	4.33±1.27	7.25±1.48	3.22±0.47

TABLE II: The table reports the mean and the standard deviation of the RMSE for all the test sets conditioned by the number of the fruits in the images. The second column gives results for no olives in the image, while the third, the fourth and the fifth give the results for low, medium and high number of fruits per image. The last column provides the results averaged on the whole test set.

To further investigate the capability of our approach, we run experiments to evaluate the performance depending on the number of fruits in the images (see Table II).

The first aspect that can be observed is that, in general, the performance of all the approaches deteriorates as the number of fruits grows. This is mainly motivated by the increasing difficulty of handling object self-occlusions (*i.e.,* bunches of fruits). Nevertheless, the RMSEs of WS-COUNT are comparable to those of the supervised architecture and there are even cases where it outperforms the supervised baselines, *i.e.,* , when the number of fruits is lower than 6 units per image. The errors slightly increase with respect to the baselines when the number of fruits is greater than 6 units. This is to be expected, since the weakly supervised approach has no supervision signals (*i.e.,* bounding boxes or total counts) that give it clues about the exact locations and number of instances. Hence, it is more challenging for WS-COUNT to learn to distinguish self-occluded objects.

Finally, we run tests on the full resolution image to compare the performance when counting fruits on complete hi-res image shots. As explained in Section IV-A, three images, depicting considerable portions of olive tree canopy, are kept aside. The results obtained on images D-281 and D-294 (see Table III) confirm that WS-COUNT gives count estimates that are very close to the supervised baselines and, more importantly, to ground-truth. Conversely, the image D-282 seems to be more challenging since the associated errors are higher, not only for WS-COUNT, but also for the other supervised baselines (*e.g.,* RCNN-Count, whose estimate is more than 50 units lower than the ground-truth one). We attribute this performance decrease to the adverse illumination conditions (light from behind the tree). Making the approach robust to such a problem will be the object of future work.

Network	**D-281**	**D-294**	**D-282**
GT	138.0	184.0	208.0
S-COUNT	132.47±7.9	163.35±15.52	172.12±11.95
MBS-COUNT	154.52±28.35	194.24±38.69	235.11±66.54
MB-PAC-Only	167.48±67.49	258.39±78.27	98.45±40.86
WS-COUNT	164.59±19.65	221.67±22.24	109.78±14.12
RCNNCount[8]	140.2±6.01	163.4±6.41	140.8±31.12
DeepCount[6]	106.32±34.8	105.6±59.78	32.98±11.68
PRM[41]	72.38±19.82	413.35±214.64	577.23±167.6

TABLE III: Table showing the performance on entire olive plant images.

V. Conclusion

In this work, we proposed a novel weakly-supervised framework for fruit counting in agricultural applications. The WS-COUNT strategy is able to learn to count without requiring task-specific supervision labels, such as manually labelled object bounding boxes or total instance count. Instead, it exploits a simple binary presence absence classifier (PAC), trained with only image level labels, and a spatial consistency loss that imposes coherency among counting branches at different scale levels. By removing the necessity for laborious data collection processes, our approach takes an important step towards the complete automation of yield estimation systems. The experiments run on three different fruit species clearly show that our approach guarantees performances that are comparable to those of fully supervised baselines.

Although the results obtained are very promising, there are still important aspects that will be addressed in future works, such as making the algorithm more robust to self-occlusion happening when there are dense groups of fruits, or improving the accuracy on adverse illumination conditions. In addition, integrating this approach on a robotic platform requires methods to deal with multiple image shots of the same trees that naturally occur when collecting a video of an orchard and which could bias the whole orchard yield estimate.

References

[1] K. Kapach, E. Barnea, R. Mairon, Y. Edan, and O. Ben-Shahar, "Computer vision for fruit harvesting robots–state of the art and challenges ahead," *International Journal of Computational Vision and Robotics*, vol. 3, no. 1-2, pp. 4–34, 2012.

[2] A. Kamilaris and F. X. Prenafeta-Boldú, "Deep learning in agriculture: A survey," *Computers and Electronics in Agriculture*, vol. 147, pp. 70–90, 2018.

[3] S. Nuske, K. Wilshusen, S. Achar, L. Yoder, S. Narasimhan, and S. Singh, "Automated visual yield estimation in vineyards," *Journal of Field Robotics*, vol. 31, no. 5, pp. 837–860, 2014.

[4] M. Stein, S. Bargoti, and J. Underwood, "Image based mango fruit detection, localisation and yield estimation using multiple view geometry," *Sensors*, vol. 16, no. 11, p. 1915, 2016.

[5] S. Bargoti and J. P. Underwood, "Image segmentation for fruit detection and yield estimation in apple orchards," *Journal of Field Robotics*, vol. 34, no. 6, pp. 1039–1060, 2017.

[6] M. Rahnemoonfar and C. Sheppard, "Deep count: fruit counting based on deep simulated learning," *Sensors*, vol. 17, no. 4, p. 905, 2017.

[7] S. W. Chen, S. S. Shivakumar, S. Dcunha, J. Das, E. Okon, C. Qu, C. J. Taylor, and V. Kumar, "Counting apples and oranges with deep learning: A data-driven approach," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 781–788, 2017.

[8] S. Bargoti and J. Underwood, "Deep fruit detection in orchards," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 3626–3633.

[9] V. Lempitsky and A. Zisserman, "Learning to count objects in images," in *Advances in neural information processing systems*, 2010, pp. 1324–1332.

[10] M. Noroozi, H. Pirsiavash, and P. Favaro, "Representation learning by learning to count," in *IEEE International Conference on Computer Vision (ICCV)*, vol. 00, Oct. 2018, pp. 5899–5907.

[11] A. Payne, K. Walsh, P. Subedi, and D. Jarvis, "Estimating mango crop yield using image analysis using fruit at stone hardeningstage and night time imaging," *Computers and Electronics in Agriculture*, vol. 100, pp. 160–167, 2014.

[12] C. C. Loy, K. Chen, S. Gong, and T. Xiang, "Crowd counting and profiling: Methodology and evaluation," in *Modeling, Simulation and Visual Analysis of Crowds*. Springer, 2013, pp. 347–382.

[13] V. A. Sindagi and V. M. Patel, "A survey of recent advances in cnn-based single image crowd counting and density estimation," *Pattern Recognition Letters*, vol. 107, pp. 3–16, 2018.

[14] J. P. Cohen, G. Boucher, C. A. Glastonbury, H. Z. Lo, and Y. Bengio, "Count-ception: Counting by fully convolutional redundant counting," in *Computer Vision Workshop (ICCVW), 2017 IEEE International Conference on*. IEEE, 2017, pp. 18–26.

[15] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 4, pp. 743–761, 2012.

[16] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1. IEEE, 2005, pp. 886–893.

[17] B. Schiele, E. Seemann, and B. Leibe, "Pedestrian detection in crowded scenes," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 01, 06 2005, pp. 878–885.

[18] O. Tuzel, F. Porikli, and P. Meer, "Pedestrian detection via classification on riemannian manifolds," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 10, pp. 1713–1727, 2008.

[19] B. Wu and R. Nevatia, "Detection and tracking of multiple, partially occluded humans by bayesian combination of edgelet based part detectors," *International Journal of Computer Vision*, vol. 75, no. 2, pp. 247–266, 2007.

[20] M. Li, Z. Zhang, K. Huang, and T. Tan, "Estimating the number of people in crowded scenes by mid based foreground segmentation and head-shoulder detection," in *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*. IEEE, 2008, pp. 1–4.

[21] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010.

[22] S. Sengupta and W. S. Lee, "Identification and determination of the number of immature green citrus fruit in a canopy under different ambient light conditions," *Biosystems Engineering*, vol. 117, pp. 51–61, 2014.

[23] I. Sa, C. McCool, C. Lehnert, and T. Perez, "On visual detection of highly-occluded objects for harvesting automation in horticulture." ICRA, 2015.

[24] K. Yamamoto, W. Guo, Y. Yoshioka, and S. Ninomiya, "On plant detection of intact tomato fruits using image analysis and machine learning methods," *Sensors*, vol. 14, no. 7, pp. 12 191–12 206, 2014.

[25] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[26] I. Sa, Z. Ge, F. Dayoub, B. Upcroft, T. Perez, and C. McCool, "Deepfruits: A fruit detection system using deep neural networks," *Sensors*, vol. 16, no. 8, p. 1222, 2016.

[27] M. A. Halstead, C. S. McCool, S. Denman, T. Perez, and C. Fookes, "Fruit quantity and ripeness estimation using a robotic vision system," *IEEE Robotics and Automation Letters*, 2018.

[28] N. Ali, K. B. Bajwa, R. Sablatnig, and Z. Mehmood, "Image retrieval by addition of spatial information based on histograms of triangular regions," *Computers & Electrical Engineering*, vol. 54, pp. 539–550, 2016.

[29] N. Ali, K. B. Bajwa, R. Sablatnig, S. A. Chatzichristofis, Z. Iqbal, M. Rashid, and H. A. Habib, "A novel image retrieval based on visual words integration of sift and surf," *PloS one*, vol. 11, no. 6, p. e0157428, 2016.

[30] B. Zafar, R. Ashraf, N. Ali, M. Iqbal, M. Sajid, S. Dar, and N. Ratyal, "A novel discriminating and relative global spatial image representation with applications in cbir," *Applied Sciences*, vol. 8, no. 11, p. 2242, 2018.

[31] B. Zafar, R. Ashraf, N. Ali, M. Ahmed, S. Jabbar, K. Naseer, A. Ahmad, and G. Jeon, "Intelligent image classification-based on spatial weighted histograms of concentric circles," *Computer Science and Information Systems*, vol. 15, no. 3, pp. 615–633, 2018.

[32] N. Ali, B. Zafar, F. Riaz, S. H. Dar, N. I. Ratyal, K. B. Bajwa, M. K. Iqbal, and M. Sajid, "A hybrid geometric spatial image representation for scene classification," *PloS one*, vol. 13, no. 9, p. e0203339, 2018.

[33] B. Zafar, R. Ashraf, N. Ali, M. Ahmed, S. Jabbar, and S. A. Chatzichristofis, "Image classification by addition of spatial information based on histograms of orthogonal vectors," *PloS one*, vol. 13, no. 6, p. e0198175, 2018.

[34] B. Xu and G. Qiu, "Crowd density estimation based on rich features and random projection forest," in *Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on*. IEEE, 2016, pp. 1–8.

[35] Y. Wang and Y. Zou, "Fast visual object counting via example-based density estimation," in *Image Processing (ICIP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 3653–3657.

[36] C. Zhang, H. Li, X. Wang, and X. Yang, "Cross-scene crowd counting via deep convolutional neural networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 833–841.

[37] D. Onoro-Rubio and R. J. López-Sastre, "Towards perspective-free object counting with deep learning," in *European Conference on Computer Vision*. Springer, 2016, pp. 615–629.

[38] M. Marsden, K. McGuinness, S. Little, and N. E. O'Connor, "Resnetcrowd: A residual deep learning architecture for crowd counting, violent behaviour detection and crowd density level classification," in *Advanced Video and Signal Based Surveillance (AVSS), 2017 14th IEEE International Conference on*. IEEE, 2017, pp. 1–7.

[39] V. Rabaud and S. Belongie, "Counting crowded moving objects," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1. IEEE, 2006, pp. 705–711.

[40] G. J. Brostow and R. Cipolla, "Unsupervised bayesian detection of independent motion in crowds," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1. IEEE, 2006, pp. 594–601.

[41] Y. Zhou, Y. Zhu, Q. Ye, Q. Qiu, and J. Jiao, "Weakly supervised instance segmentation using class peak response," in *CVPR*, 2018.

[42] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[43] T. Durand, T. Mordan, N. Thome, and M. Cord, "Wildcat: Weakly supervised learning of deep convnets for image classification, pointwise localization and segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[44] S. Bargoti, "Pychet labeller - an object annotation toolbox." 2016. [Online]. Available: https://github.com/sbargoti/pychetlabeller

[45] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[46] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[47] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.