Leaf Counting: Multiple Scale Regression and Detection Using Deep CNNs

Yotam Itzhaky
yotamit@post.bgu.ac.il
Guy Farjon
guyfarjon@gmail.com
Faina Khoroshevsky
bordezki@post.bgu.ac.il
Alon Shpigler
alonshp@post.bgu.ac.il
Aharon Bar Hillel
aharon.barhillel@gmail.com

Ben Gurion University of the Negev
Beer Sheva, Israel

Abstract

Visual object counting is a computer vision task relevant to a broad spectrum of problems, and specifically to the phenotyping domain. We propose two novel deep learning approaches for the visual object counting task, demonstrating their efficiency on the CVPPP 2017 Leaf Counting Challenge dataset. The first method performs counting via direct regression, predicting the count value using multiple scale representations of the image and using a novel fusion technique to combine the multi-scale predictions. In the second method, we count after predicting and aggregating all the leaf center points. Experimental results show that both our algorithms outperform last year's CVPPP challenge winners, while our second pipe also provides additional information of the leaf center points with a 95% average precision.

1 Introduction

Counting objects is relevant to a broad spectrum of computer vision problems, and specifically is prevalent in the phenotyping domain. One example of this kind of problem is counting humans or cars in very crowded scenes[19, 28]. The ability to estimate the number of humans in a crowded scene or vehicles in a traffic congestion can be used to improve security in stadiums, or monitor and improve traffic congestion problems [22]. Another type of counting related tasks involves counting the exact number of objects in an image with a high precision estimation, a common requirement in a variety of agricultural problems. Knowing the exact number of fruits, flowers, and trees helps farmers making better decisions on cultivation practices, plant disease prevention, and the size of harvest labor force [24].

In this work, we focus on the task of leaf counting, using a dataset of tobacco and Arabidopsis plants images [3, 21], but the techniques we developed could also be used in other counting applications. There are several possible approaches to this task. Convolutional

Neural Networks (CNNs) have dramatically improved the state-of-the-art in visual object recognition and detection in recent years [15, 26]. Thus, a simple idea is to perform counting by detection, given a dataset where each object is annotated by a bounding box. One can choose among several successful detection algorithms [18, 25, 26] to detect the leaves in the image and then apply an additional regression layer to produce the final counting estimation. With this approach we gain additional information to counting - the leaves' exact image locations. This information may be valuable for applications that require further processing of the leaves, like detecting diseases or examining the leaf's morphological structure. However, using a detector for the counting task requires obtaining a very high detection accuracy, and imperfect detection may lead to low accuracy for the counting task. It also requires annotating each leaf in each image with a bounding box.

Approaches requiring less annotation effort are to use dot annotations, marking the object center, or just image-level counts. Dot annotations are simpler to obtain than bounding boxes, but often require additional cues in order to be used in complex tasks [2]. If only image level counts are available, the counting task can be addressed by a direct regression approach, where the regressor is optimized solely to output count estimations [5, 20]. The natural choice for the regressor is an adapted deep CNN. The advantages of such approach are the simplicity of the algorithm, hence the ability to train it from smaller sample size, and the alleviation of the annotation burden. In this paper, we develop and compare two approaches for leaf counting: a direct regression pipe and a detection based method.

To the best of our knowledge, the best results reported of the leaf counting CVPPP challenge datasets [8, 21, 27]) are obtained with a direct regression approach [5, 31]. Adopting a similar pipe, we suggest two improvements to this method. The first is regressing the number of leaves from multiple image scales, thus accounting for cases of small and large leaves, using a Feature Pyramid Network (FPN) [32]. The second is fusion of the multiple estimates from the multiple image scales based on an estimated confidence variance. The regressor's variance is estimated from the deep network as an additional output to the main regressed value (the number of leaves). We examine several techniques for fusing the multiple estimators, including min-variance selection and a Maximum Likelihood Estimation (MLE) solution. The usage of multiple scales and the novel fusion techniques provide accuracy improvements over the current state of the art.

The second pipe proposed in this work involves counting by detection/density estimation. While detection usually relies on bounding box annotation, here we adjust such an algorithm to the simpler "leaf center" point annotations. To cope with the minimal annotation, Our detection pipe is based on regression of a "heat map" — a map with Gaussians of predefined parameters placed upon the annotated leaf centers, similar to other density estimation methods [16]. The density map estimated then goes through a spatial softmax layer for non-maxima suppression, followed by a global sum layer to get the count estimate. This detection based estimate is fused with direct regression, in a network driven by two losses: detection based and regression based. We show that this pipe also outperforms the current methods, and additionally provides the ability to detect individual leaves' locations, using the information given by the detector.

The rest of the paper is organized as follows: we review related work Section 2, present our algorithm at length in Section 3, present results in Section 4 and discuss conclusions and further work in Section 5.

2 Related work

Our contribution relates concretely to detection-based and direct regression counting, but we rely on recent advances in multiple-scale image processing and uncertainty measurement in deep models for our main contributions.

Counting via detection/density estimation: In 2010, Lempitsky and Zisserman [16] introduced an object counting method based on object density map regression. Their approach appeared to perform well, even in high density and high occlusion cases. Following this work, several authors [2, 23] used random forest regression for object density estimation. Zhang et al. [34] were among the first to apply CNN-based methods to the crowd counting task. They proposed to learn a CNN, based on AlexNet architucture [14], by alternating training on two objective functions: direct count and density estimation. Cohen et al. [4] suggested predicting in each location a local count of items in the area surrounding it. This leads to an over-counting, which is later accounted for. Lu et al. [20] applied a similar method to outdoors counting of maize tassels. Finally, the most similar work to ours is the recent [1], where a counting network is trained using both a direct regression loss and a density estimation loss. Unlike our work, they use the regression loss only as a regularizer for the direct regression pipe, but it is not used for getting a count estimate.

Counting via direct regression: The majority of published works in the leaf counting domain adopted a direct count approach [5, 8, 31]. The winners of the 2015 CVPPP challenge [8] proposed a learning approach for counting leaves in rosette plants. They used a supervised regression model to relate image-based descriptors, which are learned in an unsupervised manner, to leaf counts. Rahnemoonfar and Sheppard [24] introduced CNN to their tomato counting model. They re-trained the Inception-ResNet CNN architucture [30] as a regression model, with the ability to skip connections between layers. In addition, they trained their network solely on generated synthetic data, and tested on real images, suggesting a solution to data hungry models. Teimuri et al. [31] referred to leaf counting as a classification task, predicting leaf numbers as labels, using the Inception-v3 architecture [29]. Dobrescu et al. [5] suggested a regression model based on ResNet-50 architecture [9], training on multiple leaf datasets to produce a more generalized model, with excellent results. None of those methods used multiple-scale image representation or uncertainty measurement in their models, as we did in our direct regression pipe.

Multiple scale approaches: The importance of processing the image in multiple scales for detection was well acknowledged for non-deep methods [6]. Successful introduction of a multi-resolution feature pyramid into deep detection models was recently done in the Feature Pyramid Network (FPN) [17], where top down and lateral connections are added to create a pyramid of equivalent feature maps at multiple resolutions. We rely on this contribution in our suggested method.

Uncertainty and ensemble fusion: Fusion of the results received from multiple deep models is often done to improve accuracy, where the simple fusion of averaging the results is often employed [5, 14]. Better fusion requires an estimation of the uncertainty for the multiple predictors. In [11] it was suggested to measure network regression uncertainty by minimizing a log-normal density loss with an input-dependent variance estimate. We make use of this technique here for obtaining variance estimates, which are then used for informed fusion.

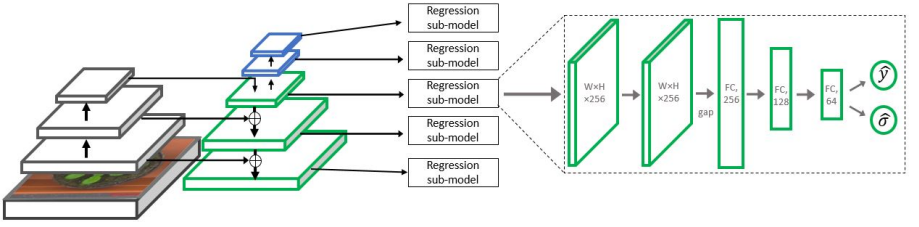

Figure 1: The direct regression architecture. The left side is the backbone network with the FPN. On the right is the regression sub-models attached to P_i level of the FPN.

3 Method

3.1 Direct regression for counting

In this section, we propose a deep CNN based architecture to infer the number of leaves in a given image, using direct regression. For this mission we adopt the FPN architecture in [17] as a backbone, and following the work of [18], create a sub-model on top of each pyramid scale. The fusion of the results given by the multiple predictors is done using a novel approach, based on thier uncertainty estimates.

3.1.1 FPN backbone

Our architecture includes representation of the image in several resolutions (octaves) and inference of the number of leaves in the given image for each of the representations. This method is based on the FPN [17] architecture, which improves multi-scale detection for several algorithms [10, 18], showing significant gains over using one resolution only. This architecture can be modified, as we present in this work, to a direct regression model.

The FPN method takes a single-scale image of an arbitrary size as input, and outputs a feature pyramid of proportionally sized maps at multiple levels, in a fully convolutional fashion. This process is independent of the backbone convolutional architectures, and in this work, similarly to [17], we used *ResNet-50* [9] as the backbone. The construction of the pyramid involves bottom-up connections, top-down pathways, and lateral connections, that generate equivalent feature maps at multiple scales (see figure 1, left). We construct a pyramid with levels P_3 through P_7, where i indicates pyramid level (P_i has resolution 2^i lower than the input). As in [17], all pyramid levels have $C = 256$ channels, with details of the pyramid generally follow [17] with a few modest differences, based on [18]. Each level of the pyramid is better suited for counting objects at a certain scale range.

3.1.2 Count regression sub model

On top of each pyramid level a regression sub-model is introduced including two 3×3 convolutional layers with 256 output maps, followed by Global Average Pooling (GAP), flattening the maps to a compact 256×1 representation. Similarly to [5], this vector is fed into two fully connected layers in decreasing sizes, 128 and 64 respectively, except in our architecture this lead to an output layer including two neurons. These two output neurons are estimates of the mean and variance of the expected number of leaves (see 3.1.3 for details). The architecture is summarized in figure 1. As in the RetinaNet [18] architecture, all the

regressor sub-models share their weights, so effectively the same regressor is applied to each pyramid level.

We hence use 5 different image pyramid levels for gaining 5 different leaf count estimators. Fusing these into a single estimator is better done when uncertainty estimates are available. We describe how such estimates are obtained in 3.1.3 and how they are fused in 3.1.4.

3.1.3 Modeling the Uncertainty of Each Sub-Model

Following [12], our regression network has two outputs, $\hat{y}$ and $\hat{\sigma}^2$, corresponding to the expected leaf count and its variance. The training data is a labeled sample of images $\{I_i, y_i\}_{i=1}^N$ with y_i the leaf count label. We train the network to minimize the following loss:

$$L_r = \sum_{i=1}^{N} \frac{1}{2\hat{\sigma}_i^2} ||\hat{y}_i - y_i||^2 + \frac{1}{2} \log \hat{\sigma}_i^2 \tag{1}$$

where $\hat{y}_i = \hat{y}(I_i), \hat{\sigma}_i = \hat{\sigma}(I_i)$ are the outputs of the regressor over image i, in a specific scale. Note that this loss is the log of a Gaussian density (up to a constant), but with the mean and variance being input-dependent estimates. This means that in image space regions where the regression is not accurate (i.e. where large deviations occur between $\hat{y}$ and the truth y), a larger $\hat{\sigma}^2$ is inferred to reduce the loss. The opposite happens in areas of accurate prediction. In practice, followed by [12], we train the network to predict the more numerically stable $s = \log \hat{\sigma}^2$, so we minimize $\sum_{i=1}^N \frac{1}{2} \exp(-s_i) \cdot ||\hat{y}_i - y_i||^2 + \frac{1}{2} s_i$ to obtain the variance estimates.

3.1.4 Fusion of the Sub-Models' Predictions

Given a variance estimate for each count prediction, new fusion options are possible. Keeping an input image index i fixed (and hence omitted), denote the multiple estimators for this image by $(\hat{y}^j, \hat{\sigma}^j)_{j=1}^J$. One intuitive way would be to choose the model with the lowest degree of uncertainty, i.e. the one with the lowest predicted variance for the specific input image:

$$\hat{y} = \hat{y^k}, \quad with \quad k = argmin(\hat{\sigma^j}) \tag{2}$$

While this choice is intuitive, we know that the MLE for the consolidation of Gaussian distributions is different. Specifically, the MLE has the closed form solution:

$$\hat{y} = \frac{\sum_{j=1}^J \frac{1}{(\hat{\sigma^j})^2} \hat{y^j}}{\sum_{j=1}^J \frac{1}{(\hat{\sigma^j})^2}} \tag{3}$$

It should be noted that the second method assumes observation independence between sub-models' results. This assumption is not fulfilled (given that each sub-model receives input as a representation of the same image at a different resolution), and in practice the fusion method given by equation 2 performed better and was used as default in our experiments.

3.2 Counting by detection of key-points

Our second approach of solving the given task is to first detect the leaves and only then, count them. The CVPPP dataset includes leaf centers, but no bounding box annotations (see 4.1 for further details). Hence, we are teaching a network to find leaf centers. To achieve this task, we first create a heat map, with a two-dimensional Gaussian of fixed width k placed around each leaf center. k was carefully chosen so that Gaussians of nearby leaf centers

will not overlap. A network was trained to predict this heat map, and then use this map to regress the number of leaves. Like in section 3.1, we start from the FPN with a backbone of $ResNet-50$. We take the middle pyramid level P_3 as our input representation, and create a detection sub-network over it to regress the heat map. A count regression sub-network then accept the heat map as input and produce its final count prediction.

3.2.1 Detection sub-network

The detection sub-model is a small Fully Convolutional Network (FCN), containing four 3×3 ReLU convolutional layers with 256 filters each.

The predicted heat map is actually an additional 3×3 ReLU convolutional layer, including a single filter. Although the final heat map is predicted following all four convolutional layers, each of those layers predict the heat map independently, and is guided using the same loss as the final loss. The architecture is summarized in figure 2.

Variables $\hat{p}$ and p are the prediction and target heat map values for a single pixel in a single image, respectively (dropping the image and the pixel indices for convenience). The loss minimized in training is a weighted smooth-L1 loss L_d, with $w = 0.1$:

$$L_d = \begin{cases} (1-w) \cdot P_{loss} & \text{, where p} > 0 \\ w \cdot P_{loss} & \text{,otherwise} \end{cases}, P_{loss} = \begin{cases} |\hat{p}-p| - \frac{1}{2} & \text{, where } |\hat{p}-p| \geq 1 \\ \frac{1}{2}(\hat{p}-p)^2 & \text{,otherwise} \end{cases} \tag{4}$$

The weighting keeps the total weight of positive pixels in the heat map high, and directs the optimization towards accurate regression of these values, rather than the pixels with the value zero, which outnumber them considerably. The total loss minimized is the sum of L_d over all pixels and images.

3.2.2 Counting sub-network

Given an estimated heat map, we can find the number of Gaussian centers. Ideally, this number will be the estimated leaf count. However, assuming imperfect detection, it is preferable not to use it directly as an estimator, but as a collaborator to the regression. To properly count the leaves, we would like each Gaussian to be shrinked to a delta function before summing. Hence we apply the following non-maxima suppression procedure to the estimated heat map $P \in R^{N \times M}$:

$$Q = MaxPool(P,(k,k)), \quad P' = P \cdot exp(-\beta \cdot |P-Q|) \tag{5}$$

Using spatial softmax function, a pixel in the output map P' keeps a value close to its original value in P only if it was the highest value of P in a $k \times k$ neighborhood. We chose k to be the same value used in the heat map creation, so a single pixel is expected to remain active for a leaf-sized Gaussian.

Spatial softmax keeps mostly the pixels which are local maxima, but it does not remove noise, so P' still contains small values in non-center pixels. We hence apply a smooth step function (sigmoid) layer of the form $y(x) = \frac{1}{1+e^{-\beta(x-t)}}$ to keep only values greater than some threshold t, to get the detection map $\hat{D}$, containing the estimated leaf centers as its active pixels. A global sum pooling layer applied to $\hat{D}$ then gives us a single number $C_{\hat{D}}$ - the detection-based estimate for the leaf count.

Beyond the detection-based estimate $C_{\hat{D}}$, Additional features may be used for the final count regression. We apply a GAP operation to the fourth convolutional layer, obtaining a 256-dimensional feature vector V_4. The final count estimator $\hat{C}$ is obtained by applying linear regression in the final layer, .i.e. $\hat{C} = W \cdot [V_4, C_{\hat{D}}]$ where $[V_4, C_{\hat{D}}]$ is a concatenated 257 features vector. We train the final counter to minimize an $L1$ loss function.

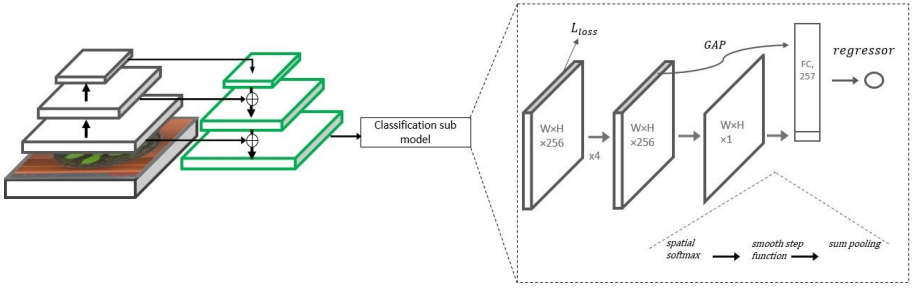

Figure 2: The detector architecture. The left side is the backbone network with the FPN. On the right is the detection sub-model which includes four 3×3 ReLU convolutional layers and at the top - the regression head.

4 Experiments and results

4.1 Experimental setup

Evaluation metrics: For the regression evaluation, we used the metrics provided by the workshop organizers (see [5]) - the average distance and the L_1 distance between estimated and true count, and the fraction of full agreement between the two. For estimation of the detection performance we use a recall-precision curve and the Average Precision (AP) metric- the area under the curve. In order to determine if a leaf detection (an active point in $\hat{D}$) is a hit or a miss we use the criterion introduced in Percentage of Correct Keypoints (PCK) computation [33]. With PCK, a leaf detection point is considered a hit if the distance between it and a ground truth leaf center is lower than $\alpha \cdot max(w,h)$ where (w,h) are the width and height of the bounding box surrounding the entire object (plant). Like in [33] we used $\alpha = 0.1$ to test our detection performance.

Data sets: The LCC data includes 4 datasets - A1, A2, A3, and A4. A1, A2, and A4 contains images of the Arbidopsis plant and A3 dataset contains images of young Taboco plants. Datasets A1, A2, and A3 are relatively small containing 128, 31, and 27 images respectively, while A4 is bigger with 624 images. Since the datasets are relatively small, we applied random transformations including rotation, vertical and horizontal flips and scaling. The images' size varies between 500×530 and 2448×2048, so we resized them to $800 \times W$ where W is chosen to keep the original image aspect ratio.

Training procedure: For the training results we train and test on each dataset separately with four fold cross validation following [5], and report average results over the folds. For the test results we trained a single model over the accumulated dataset, with 80% of the Ac data used for model fitting and 20% as a validation set for the early stopping criterion. We did not use the foreground plant mask in either of our model, and used the leaf centroid information in the detection pipe only. We used ADAM [13] as optimizer, with a learning rate of $1e^{-5}$.

4.2 Results

We consider first the train and test counting results of the two approaches we suggest, and compare these results to the state of the art. Next, we present experiments showing a breakdown of the elements contributing to each approach, and the detection results of the detection-based pipe.

Dataset	DiC			|DiC|			Agreement [%]		
	MSR	D+R	[5]	MSR	D+R	[5]	MSR	D+R	[5]
A1	0.02	0.08	-0.81	0.59	0.64	0.94	49	47	25
A2	1.28	0.52	-2.38	2.05	1.24	2.38	20	24	38
A3	-0.16	-0.42	-0.57	1.05	1.45	1.43	14	11	14
A4	0.03	0.12	0.1	0.67	0.68	0.91	47	47	35

Table 1: Cross validation results on seperate datasets. The presented algorithms are Multiple Space Regression (MSR, see section 3.2), Detection+ Regression (D+R, section 3.1), and the direct regression results of [5] for comparison

	DiC			|DiC|			Agreement [%]			MSE		
	MSR	D+R	Ref [5]	MSR	D+R	Ref [5]	MSR	D+R	Ref [5]	MSR	D+R	Ref [5]
A1	-0.27(1.21)	0.12(1.11)	-0.39(1.17)	0.70(1.02)	0.73(0.84)	0.88(0.86)	57.6	45.5	33.3	1.48	1.21	1.48
A2	-0.22(0.67)	0.44(0.73)	-0.78(1.64)	0.44(0.53)	0.44(0.73)	1.44(1.01)	55.6	66.7	11.1	0.44	0.67	3.00
A3	-0.04(1.61)	-0.27(1.14)	0.13(1.55)	1.14(1.12)	0.84(0.80)	1.09(1.10)	32.1	33.9	30.4	2.54	1.34	2.38
A4	0.15(1.08)	0.45(0.97)	0.29(1.10)	0.76(0.78)	0.75(0.76)	0.84(0.76)	42.9	39.9	34.5	1.19	1.14	1.28
A5	0.10(1.22)	0.26(1.05)	0.25(1.21)	0.84(0.90)	0.76(0.77)	0.90(0.85)	41.7	39.6	33.2	1.50	1.17	1.53
All	0.07(1.22)	0.26(1.05)	0.19(1.24)	0.83(0.90)	0.76(0.77)	0.91(0.86)	42.3	39.9	32.9	1.49	1.17	1.56

Table 2: Results of the Multiple Scale Regressor (MSR) and the Detection+Regression approach (D+R) on the held-out test set, with comparison to the winner of the 2017 CVPPP challenge [5].

Results and comparison to previous work: The cross validation results on the four datasets are reported in Table 1. The results are more stable for datasets $A1$ and $A4$, which are larger. In Table 2 we present our results on the test data of the CVPPP challenge. Our two new models outperform last year's winner of the LCC challenge in almost all metrics. Specifically, in the final 'all' test set, our approaches achieve 42% (multiple scale regression) and 40% (detection+regression) count agreement, while [5] (last year's winner) achieved 33%. A more recent result on this dataset was presented by [31], which only reported the results on the combined dataset: DiC - 0.52, AbsDiC - 1.31 and count agreement - 41%. Compared to this work, we obtained better scores in the DiC and AbsDiC metrics in both our models, and a higher count agreement using the direct regression model.

Multiple Scale Regression pipe: Table 3 presents the results of our preliminary experiments on several models providing intermediate check points between a baseline similar to [5], and our multiple scale regressor. The top row present the results of a model similar to [5] which was our baseline. This model includes a $ResNet-50$ backbone, global

	Input	Loss	Description	DiC	|DiC|	Agreement [%]	MSE
1	C_5	L_2 *loss*	Baseline	-0.92	1.08	38	2.93
2	P_3	L_2 *loss*	pyramid 1 level	0.27	0.74	39	1.08
3	$P_3 - P_7$	L_2 *loss*	Avg fuse	0.34	0.76	42	1.31
4	$P_3 - P_7$	*Unc loss*	Avg fuse	0.10	0.69	44	1.04
5	$P_3 - P_7$	*Unc loss*	*min* $\hat{\sigma}$ fuse	-0.01	0.62	49	0.91
6	$P_3 - P_7$	*Unc loss*	MLE fuse	-0.05	0.60	49	0.87

Table 3: Incremental improvements over a baseline in the direct regression model, trained and validated on a single fold of A4 dataset, in out preliminary experiments. The table rows present the following: **1.** A baseline similar to [5]. **2.** Replacing the input with the middle pyramid level. **3.** Using 5 pyramid levels, averaging the results. **4.** Like 3, with log-Gaussian loss (Eq. 1). **5.** Like 4, with min-sigma fusion of the predictors (Eq. 2). **6.** Like 3, with MLE fusion (Eq. 3).

Condition	DiC	|DiC|	Agreement [%]	MSE
1. Best model	0.08	0.64	47	0.89
2. No data Aug.	0.21	0.66	46	0.93
3. No multiple losses	1.06	1.41	34	3.84
4. Detection only	0.83	1.25	27	2.88

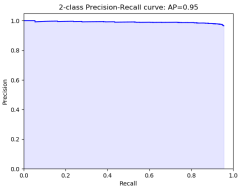

Figure 3: Left: Ablation table for the cross validation results of the detection model on the A1 dataset. **1.** The full model. **2.** Training without data augmentation. **3.** Using a single detection loss instead of four, one after each convolutional layer. **4.** Using the detection-based estimator $C_{\hat{D}}$ as the single feature for count regression. Right: The recall-precision curve, generated based on the validation set of the detection model, when trained on 80% of the Ac dataset and validated on 20% of it.

average pooling on top of C_5 convolution layer, followed by two fully connected layers in decreasing sizes and a final single neuron predictor. It obtaines count agreement of 38% and a $|DiC|$ score of 1.08, compared to 35% and 0.91 in [5] respectively. As can be seen, several steps improve the performance over this baseline: moving to pyramid features, log Gaussian loss and variance-based fusion methods. In additional experiments we found the Maximum-likelihood estimator (Eq. 3) slightly inferior to simple choice of the estimator with the lowest variance (Eq. 2), and hence the latter was used in the rest of our experiments.

Detection+Regression pipe: Figure 3 left, shows the metric scores for several ablated model versions. The results indicate that adding multiple losses to each of the FCN sub-model convolutional layers, and applying additional 256 features to the regressor beyond $C_{\hat{D}}$, improves the detector's performance. However, the data augmentation techniques we used did not significantly contribute the model. Figure 3 right, presents the detection performance based on the AP metric, as presented in 4.1, showing the trade-off between precision and recall. The resulted AP value is 0.95, presenting both high recall and high precision values, indicating high detection accuracy (high precision) and a low false negative rate (high recall). Some examples of the detectors' performance are shown in figure 4. It can be seen from these examples, that the Gaussian heat map successfully predicts the vast majority of the leaves, and that the misses of the detector occur in cases of close leaves, high leaf occlusion and very small leaves.

5 Conclusions and future work

We presented two methods for leaf counting, suited for different levels of annotations. The first method solves leaf counting as a direct regression task. We showed that using multi scale representations of the image and predicting the leaf number using each of them improves the counting accuracy. Additionally, we suggested a novel fusion approach for the multi scale predictions. In the second approach we suggested a counting approach by leaf detection, using leaf center annotations as ground truth. Empirically, the methods show similar performance, and both provide improvements over existing art.

There are several directions which we think are likely to improve the presented results. In the direct regression method, we believe that using L_1 loss function rather than L_2 can improve the results. More specifically, we plan to change the Gaussian distribution loss (Eq. 1) to a Laplacian, as done in [12], and modify the fusion method accordingly. In the detection-

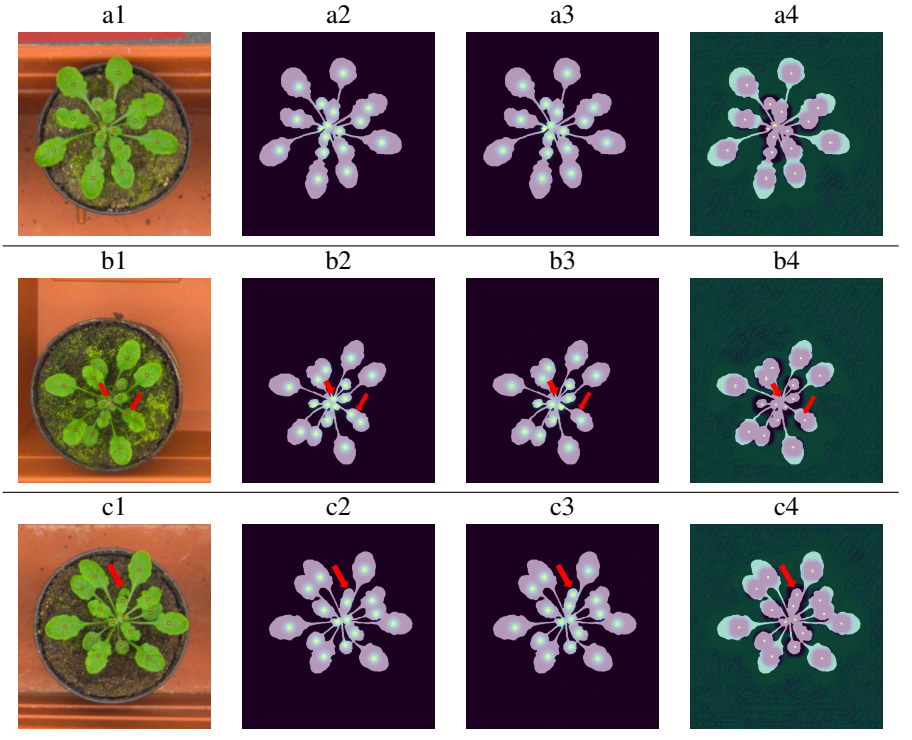

Figure 4: Leaf centers detection examples for two images. Example 'a' presents a successful detection while example 'b' presents some miss detections (pointed with red arrows). Example 'c' presents a case where we correctly found a leaf that wasn't annotated. Images a1,b1,c1 present the images with their ground truth point annotations. Images a2,b2,c2 present the generated ground truth Gaussian heat map. Images a3,b3,c3 present the model predictions of the Gaussian heat map, and images a4,b4,c4 present the Gaussian predictions after the spatial softmax operation.

based pipe, incorporating the multi-scale approach is a good direction. Finally, combining the two approaches into a single unified network is likely to provide better accuracy, though maybe at the cost of larger sample size and annotation requirements.

Acknowledgements

The authors were supported by the Generic technological R&D program of the Israel innovation authority, and the Phenomics consortium.

References

[1] Shubhra Aich and Ian Stavness. Improving object counting with heatmap regulation. *CoRR*, abs/1803.05494, 2018. URL http://arxiv.org/abs/1803.05494.

[2] Carlos Arteta, Victor Lempitsky, and Andrew Zisserman. Counting in the wild. In *European conference on computer vision*, pages 483–498. Springer, 2016.

[3] Jonathan Bell and Hannah M. Dee. Aberystwyth leaf evaluation dataset, November 2016. URL https://doi.org/10.5281/zenodo.168158.

[4] Joseph Paul Cohen, Geneviève Boucher, Craig A. Glastonbury, Henry Z. Lo, and Yoshua Bengio. Count-ception: Counting by fully convolutional redundant counting. *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 18–26, 2017.

[5] Andrei Dobrescu, Mario Valerio Giuffrida, and Sotirios A Tsaftaris. Leveraging multiple datasets for deep leaf counting. In *Computer Vision Workshop (ICCVW), 2017 IEEE International Conference on*, pages 2072–2079. IEEE, 2017.

[6] P DollÃąr, R Appel, S Belongie, and P Perona. Fast feature pyramids for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36 (8):1532–1545, 2014.

[7] Luca Fiaschi, Ullrich Köthe, Rahul Nair, and Fred A. Hamprecht. Learning to count with regression forest and structured labels. *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pages 2685–2688, 2012.

[8] Mario Valerio Giuffrida, Massimo Minervini, and Sotirios A. Tsaftaris. Learning to count leaves in rosette plants. In *BMVA press*, 2016.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[10] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2980–2988. IEEE, 2017.

[11] A Kendall and Y Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[12] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in neural information processing systems*, pages 5574–5584, 2017.

[13] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[15] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553): 436, 2015.

[16] Victor S. Lempitsky and Andrew Zisserman. Learning to count objects in images. In *NIPS*, 2010.

[17] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, volume 1, page 4, 2017.

[18] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2999–3007, 2017.

[19] Shunqiang Liu, Sulan Zhai, Chenglong Li, and Jin Tang. An effective approach to crowd counting with cnn-based statistical features. In *Smart Cities Conference (ISC2), 2017 International*, pages 1–5. IEEE, 2017.

[20] Hao Lu, Zhiguo Cao, Yang Xiao, Bohan Zhuang, and Chunhua Shen. Tasselnet: counting maize tassels in the wild via local counts regression network. *Plant methods*, 13 (1):79, 2017.

[21] Massimo Minervini, Andreas Fischbach, Hanno Scharr, and Sotirios A Tsaftaris. Finely-grained datasets for image-based plant phenotyping. *Pattern recognition letters*, 81:80–89, 2016.

[22] Daniel Onoro-Rubio and Roberto J López-Sastre. Towards perspective-free object counting with deep learning. In *European Conference on Computer Vision*, pages 615–629. Springer, 2016.

[23] Viet-Quoc Pham, Tatsuo Kozakaya, Osamu Yamaguchi, and Ryuzo Okada. Count forest: Co-voting uncertain number of targets using random forest for crowd density estimation. *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 3253–3261, 2015.

[24] Maryam Rahnemoonfar and Clay Sheppard. Deep count: fruit counting based on deep simulated learning. *Sensors*, 17(4):905, 2017.

[25] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[26] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[27] H Schaar, M Minervini, A Fischbach, and S.A.Tsaftaris. Annotated image datasets of rosette plants. In *European Confernce On Computer Vision*, 2014.

[28] Santi Seguí, Oriol Pujol, and Jordi Vitria. Learning to count with deep object features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 90–96, 2015.

[29] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826, 2016.

[30] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, 2017.

[31] Nima Teimouri, Mads Dyrmann, Per Rydahl Nielsen, Solvejg Kopp Mathiassen, Gayle J Somerville, and Rasmus Nyholm Jørgensen. Weed growth stage estimator using deep convolutional neural networks. *Sensors*, 18(5):1580, 2018.

[32] Yuanpu Xie, Fuyong Xing, Xiangfei Kong, Hai Su, and Lin Yang. Beyond classification: structured regression for robust cell detection using convolutional neural network. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 358–365. Springer, 2015.

[33] Yi Yang and Deva Ramanan. Articulated human detection with flexible mixtures of parts. *IEEE transactions on pattern analysis and machine intelligence*, 35(12):2878–2890, 2013.

[34] Cong Zhang, Hongsheng Li, Xiaogang Wang, and Xiaokang Yang. Cross-scene crowd counting via deep convolutional neural networks. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 833–841, 2015.