Leaf Counting Without Annotations Using Adversarial Unsupervised Domain Adaptation

Mario Valerio Giuffrida
University of Edinburgh
v.giuffrida@ed.ac.uk

Andrei Dobrescu
University of Edinburgh
A.Dobrescu@ed.ac.uk

Peter Doerner
University of Edinburgh
Peter.Doerner@ed.ac.uk

Sotirios A. Tsaftaris
University of Edinburgh
The Alan Turing Institute
S.Tsaftaris@ed.ac.uk

Abstract

Deep learning is making strides in plant phenotyping and agriculture. But pretrained models require significant adaptation to work on new target datasets originating from a different experiment even on the same species. The current solution is to retrain the model on the new target data implying the need for annotated and labelled images. This paper addresses the problem of adapting a previously trained model on new target but unlabelled *images. Our method falls in the broad machine learning problem of domain adaptation, where our aim is to reduce the difference between the source and target dataset (domains). Most classical approaches necessitate that both source and target data are simultaneously available to solve the problem. In agriculture it is possible that source data cannot be shared. Hence, we propose to update the model without necessarily sharing the data of the training source to preserve confidentiality. Our major contribution is a model that reduces the domain shift using an unsupervised adversarial adaptation mechanism on statistics of the training (source) data. In addition, we propose a multi-output training process that (i) allows (quasi-)integer leaf counting predictions; and (ii) improves the accuracy on the target domain, by minimising the distance between the counting distributions on the source and target domain. In our experiments we used a reduced version of the CVPPP dataset as source domain. We performed two sets of experiments, showing domain adaptation in the intra- and inter-species case. Using an Arabidopsis dataset as target domain, the prediction results exhibit a mean squared error (MSE) of* 2.3. *When a different plant species was used (Komatsuna), the MSE was* 1.8.

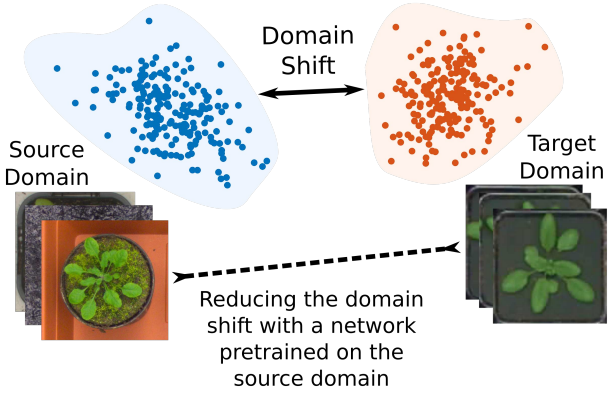

Figure 1. Domain shift representation: two datasets consisting of the same semantic objects have different representation. To reduce the domain shift, we considered the following domain adaptation scenario. a neural network is pretrained to perform leaf counting in a dataset (source domain). The trained model is given to someone who wants to use it with their data. The model is fine tuned on the target data, using adversarial domain adaptation. Our model does not require direct access to source data, but only their image representation (features).

1. Introduction

Plant phenotyping focuses on the characterisation of plants by analysing visual traits. The large-scale analysis of plants is intractable when performed manually, as it is time-consuming and error-prone [9]. Image-based plant phenotyping can help reduce effort in train extraction, but requires the development of robust algorithms to obtain accurate predictions [36]. Machine learning has been recent.ly employed to tackle plant phenotyping problems, such as plant segmentation [20], plant stress assessment [31], plant image synthesis [12, 44], leaf segmentation [26, 27, 29, 41, 44],

and leaf counting [1, 7, 10, 11, 16].

The power of machine learning stems on the ability to learn a function $f(\cdot)$ that maps the data $x \in \mathcal{X}$ to desired values $y \in \mathcal{Y}$. Typically, a dataset $\mathcal{D} = \{\mathcal{X}, \mathcal{Y}\}$ comes with data and annotations and the function f can learn a (non-linear) relationship between these two sets: this is known as *supervised learning*. However, it is known that when trained on a source dataset may not work with adequate precision on unseen target data. This is typically known as the generalisation problem. When the distributions between the data sets differ this is known as the domain shift problem. In Figure 1, we show a graphical representation of such a problem: two datasets representing the same semantic objects (in this case, plants), are mapped into two different clusters in the feature space. The distance between these two clusters is known as domain shift.

Here, we are interested in a simple, but very practical, scenario. How can we take a pretrained model (trained on annotated source data) and optimise it on an unseen target dataset, assuming we do *not* have access to the labels in this new dataset? We are particularly interested in the case that the optimisation on the target domain assumes no direct access to the source dataset. This is a scenario particularly prevalent now, due to concerns of confidentiality between data holders. Even if we want to share models, we may not be able to share source data and annotations.

In this paper, we propose a method that aims to improve performance of leaf counting models in unseen scenarios, using adversarial domain adaptation to reduce the domain shift between two datasets (c.f. Figure 1). Inspired by [38], we propose a method to learn to count leaves on a new target dataset, without providing the actual labels (no annotations). We start from a network pretrained on a dataset, known as source domain ($\mathcal{D}_S$). Using adversarial learning, we can learn leaf counting on a new dataset (target domain $\mathcal{D}_T$), using the knowledge the network has already learnt when pretrained on the source domain. The purpose of adversarial learning is to reduce the domain shift between source and target datasets.

The contribution of this paper are multi-fold:

- We apply unsupervised domain adaptation in a plant phenotyping problem, for the first time (to the best of our knowledge). Once a deep network is pretrained to perform leaf count in a known source dataset, our method optimises the network parameters to perform leaf counting in an unseen dataset without having access to the labels.
- During the domain adaptation, the source datasets are not required (only their image representation is needed). In case our model would be pretrained on private data, restricted-access images do not need to be disclosed, ensuring data confidentiality.
- Instead of global regression [7, 10, 16], we propose a multi-output regression, which brings the following benefits: (i) allows (quasi-)integer leaf counting, by constraining the network with a properly designed regulariser; and (ii) allows to learn a distribution of the predicted leaf count. The actual leaf counting is obtained using the *SumLayer*, which sums all the contributions of the multi-output regression layer.
- We impose a prior distribution over the leaf counting prediction, as the multi-output regression allows to learn a distribution over the counts. This is achieved by minimising the Kullback-Leibler divergence between the predicted and a prior distribution .

The reminder of this paper is structured as follows. Section 2 discussed the related works. The proposed method is described in Section 3. Then, Section 4 shows experimental results demonstrating the effectiveness of our approach. Then, Section 5 concludes the paper.

2. Related Works

As of now, deep neural networks are the state-of-the-art of machine learning models, as they have been demonstrated to work well for plant phenotyping problems as well [37], such as leaf segmentation [26, 27, 41, 44] and leaf counting [1, 7, 10, 16] show extraordinary performance. Although deep learning has been demonstrated to perform well on the leaf counting, still lacks of generalisation on unseen datasets. One way to solve the generalisation problem is to perform domain adaptation: a network is pretrained to perform leaf counting on a dataset (source domain), then the learnt knowledge is adapted to work on another dataset (target domain).

A lot of research has been devoted on the domain adaptation problem. Firstly, we discuss related works in domain adaptation (a recent survey can be found in [6]). Then, we will describe related works utilising domain adaptation and, more general transfer learning, in plant phenotyping. Although domain adaptation is a special case of transfer learning, they are different problems.

Following the taxonomy in [6], we focus to solve domain adaptation in the homogeneous case, where source $\mathcal{D}_S$ and target $\mathcal{D}_T$ domains are *similar* (their representations are in the same space, but are far due to the domain shift).

2.1. Domain Adaptation

With the emergence of big data and machine learning, domain adaptation has become an important area of study, as a way to lessen the burden of acquiring annotations [23]. With the increasing prevalence of deep learning, numerous domain adaptation approaches have been proposed for many visual tasks, such as object recognition [15], image

classification [38], and face recognition [30]. A first domain adaptation approach to transfer knowledge from a source to a target dataset is the *finetuning*: a network is trained on $\mathcal{D}_S$ on a given task; this pretrained network is used as starting point for $\mathcal{D}_T$. Although it is a successful technique, fine-tuning is supervised and requires an annotated target domain. In [24], the authors present how fine-tuning can be performed in an unsupervised fashion in an image retrieval problem. However, they rely on reconstructed 3D models of the data to make a dataset of positive and negative examples, which are then used to fine tune the network.

When labels for the target domain $\mathcal{Y}_T$ are not available, the typical approach is to match the feature space of the target domain with the space learnt from the source domain. Let Φ_S be the representation space of the source domain, $\Phi_S = \{\phi_S(x) \mid x \in \mathcal{X}_S\}$, and Φ_T the representation space of the target domain, the task that domain adaptation algorithms solve is to reduce the distance between Φ_S and Φ_T:

$$\min_{\Phi_{\{S,T\}}} d(\Phi_S, \Phi_T). \tag{1}$$

A common approach regularly deployed is to break down the problem into two steps. The first one is to train a supervised deep neural network, and the second one is to optimise domain shift. Different choices of the function $d(\cdot, \cdot)$ lead to different methods. In [18], the authors minimise the *maximum mean discrepancy* between source and target feature space. In [33], the *correlation alignment* (CORAL) [32] is used to minimise the covariance between the two domains. In [38], the authors propose the *Adversarial Discriminative Domain Adaptation* (ADDA) to minimise the domain shift using adversarial loss [13]. In this case, a discriminator network is responsible to differentiate between features extracted from the source and the target domain, while the feature extractor tries to make the target feature space as similar as the source domain. A similar approach is the *Domain-Adversarial Neural Network* (DANN) [8], where the authors propose a model that solves two tasks at the same time: (i) classification in the source domain; and (ii) domain classification (source vs. target). The network is optimised such that the features are discriminative for the task, but are non-discriminative for the domain classification. Therefore, if the domain classifier cannot differentiate between features coming form the source or target domain, it means the domain gap has been reduced.

All of the aforementioned approaches solve the domain adaptation problem in a classification task, whereas primarily leaf counting is a regression task. A recent work addressing domain adaptation on regression task is described in [4]. The authors propose a convex optimisation framework for sample re-weighting. Although their approach can be used on the regression task, it assumes the availability of source data during the domain adaptation and it still lacks of end-to-end learning integration.

Therefore, we decided to build upon the ADDA paradigm [38] for this paper for the following reasons: (i) it is a general framework using adversarial learning (ideally, any neural network can be used); (ii) it does not require access to the source dataset when the network is optimising for the target domain (only the image representation Φ_S are required); (iii) the learning from source and target are separated (useful when the model needs to be deployed); (iv) it is unsupervised in the target domain (no expert manual annotations are required).

2.2. Domain Adaptation in Plant Phenotyping

Although the problem of domain shift in plant phenotyping was recently emphasised in [39], domain adaptation in plant phenotyping has not been extensively explored and investigated. In [35], a plant classification problem is proposed, where a GoogLeNet is pretrained on ImageNet, which is then fine tuned on the CVPPP dataset (Arabidopsis vs Tobacco). A similar approach is followed in [7, 10], where a ResNet-50 is pretrained on ImageNet, then fine tuned to predict leaf counting. In particular, in [10] the fine-tuning is used to perform leaf counting in other datasets. Given a network pretrained on only the Arabidopsis images in the CVPPP dataset (A1, A2, and A4), the network is fine tuned on other datasets (e.g., Tobacco A3).

Although these can be seen as solving the domain adaptation problem in plant phenotyping, all of them require labels on the target domain. Therefore, we are the first to present an algorithm to perform unsupervised domain adaptation in plant phenotyping applied to leaf count.

3. Proposed Method

The methodology we propose is split into three steps, as displayed in Figure 2. First, we pretrain a neural network as in [7] to achieve leaf counting, where we extend the model via the multi-output regression task and SumLayer. Then, we detail how we can perform domain adaptation from one dataset to another, using adversarial training. Lastly, we describe how the trained model can be finally used to obtain leaf counts in the target domain.

3.1. Pretraining

The model used to be trained for the regression task consists of two subnetworks: (i) the feature extractor; and (ii) the leaf counting network (LCN in Figure 2(1)). Following the successful performance shown in [7, 10], we use a ResNet-50 [14] as a feature extractor (network architecture details are in [7, 14]). The output feature vector of the ResNet-50 has a dimension of 2048, which is provided to the Leaf Counting Network (LCN) (details of the architecture are provided in Table 1).

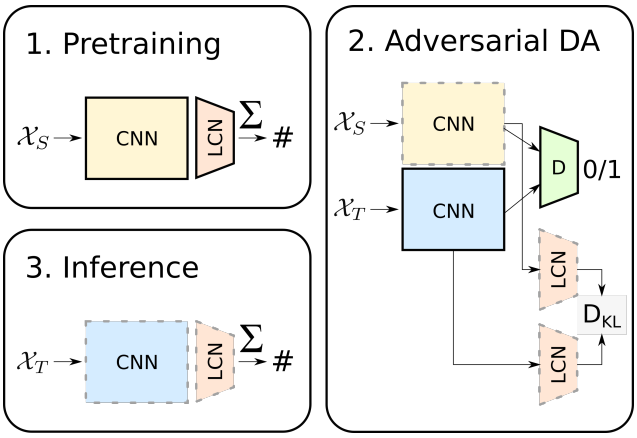

Figure 2. Outline of our proposed method, based on [38]: (i) pretraining using supervised learning on the source domain $\mathcal{X}_S$; (ii) unsupervised adversarial domain adaptation (DA) using the features from the source domain Φ_S and the unlabelled data of the target domain $\mathcal{X}_T$; (iii) inference in the target domain. Solid lines indicate (sub-)networks which weights are optimised, whereas faded dashed lines indicate (sub-)networks with frozen parameters. The greek letter Σ indicates the presence of the SumLayer.

Layer	Name	Input Size	Output Size	Activation
Dense	D1	2048	1024	ReLU
Dense	D2	1024	512	ReLU
Dense	D3	512	50	Sigmoid
Sum	O	50	1	-

Table 1. Architecture details of the Leaf Counting Network.

3.1.1 Multi-output layer for integer counting

The layer D3 in the LCN (c.f. Table 1) is a fully-connected layer of size m using the sigmoid $l = \sigma(x)$ as activation function. Thus, the output l of the D3 layer is in the range of $[0, 1]$. We can constrain the network to output numbers very close to either '0' or '1' and thus achieve (quasi-)integer counting. To achieve this, we constrain the derivative $\nabla_x \sigma(x)$ of the sigmoid function, as the derivative of a function is directly related to its steepness. Observing Figure 3, the first derivative of the sigmoid function reaches its minimum (thus the least steep points) when the function approaches $\pm\infty$. Hence, we add a constraint to the learning process penalising for the first derivative of $\sigma(x)$, encouraging the network to minimise it as follows:

$$\mathcal{L}_{\text{int}} = \lambda_{\text{int}} \|\nabla_x \sigma(x)\|_2 = \lambda_{\text{int}} \|\sigma(x)(1 - \sigma(x))\|_2^2.$$

We set in our experiments $\lambda_{\text{int}} = 0.02$.

3.1.2 SumLayer

As said in the previous section, the output of the D3 layer is a sequence of m numbers between $[0, 1]$, but it does not provide the actual leaf counting. We obtain the actual leaf counting by simply summing all the nodes in the D3 layer. Formally, the output of the network $\hat{y}$ is computed as:

$$\hat{y} = \sum_{k=1}^{m} l_k. \tag{2}$$

A visual representation of the SumLayer is shown in Figure 4: the multi-output regression provides a sequence values in $[0, 1]$ (sigmoid activation function), then they are summed to obtain the final leaf counting prediction.

We can use the output of Equation (2) to train the network using the mean squared error as loss function [7, 10]:

$$\mathcal{J}(y, \hat{y}) = \frac{1}{2} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2. \tag{3}$$

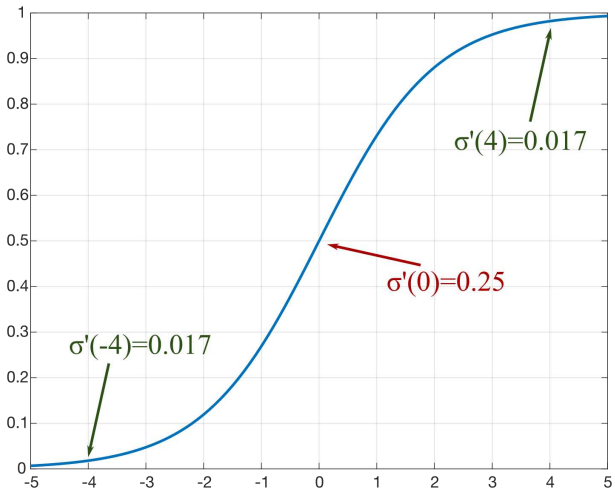

Figure 3. Sigmoid activation function and some derivatives. The function reaches its steepest point at $x = 0$ (red derivative). The function is flatter when $x \to \pm\infty$ (green derivatives).

3.2. Adversarial Learning

Originally proposed in [13], the main concept of adversarial learning is to train a network, called *generator*, to produce data as similar as possible to the train data. Specifically, let $P(s)$, $\forall s \in \mathcal{X}_s$, be the probability of the source space, a network $G(t; \Theta)$, $\forall t \in \mathcal{X}_t$, parametrised by the set Θ, learns a distribution $P(G(t; \Theta))$, such that $P(G(t; \Theta)) = P(s)$.

The parameters Θ are optimised with the help of another network $D(x; \Psi)$, called *discriminator*, which has the task to classify whether the input data x come from the source or target distribution. Adversarial networks take their name from the fact that the generator competes with the discriminator in a zero-sum game: G has the task to produce data as similar as the training set, while the discriminator's task

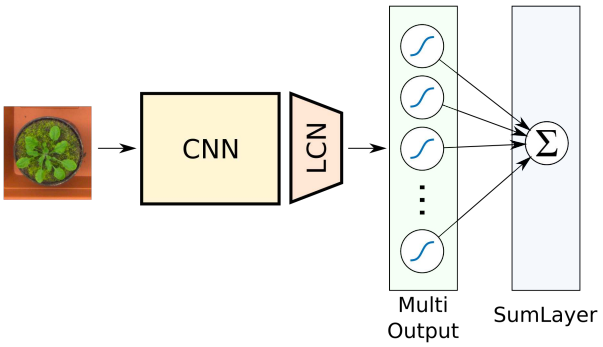

Figure 4. Graphic representation of the multi-output regression and the SumLayer. The output of the network is a total number of leaves and can be trained using MSE loss function.

is to tell what is real and what is generated (fake). In our case, the generator network is the ResNet-50, which takes an image as input and provides its representation as output. The role of the discriminator is to be able to differentiate features coming from the source and target domain. By doing that, the generator is encouraged to output features extracted from the images in the target domain as similar as the features extracted from the images in the source domain.

The parameters of generator and discriminator are alternately updated, optimising the following objectives:

$$\min_{\Psi} \quad \mathbb{E}_{s\sim\mathcal{X}_S}\left[\mathcal{F}(D(G(s;\Theta_s);\Psi),1)\right] + \mathbb{E}_{t\sim\mathcal{X}_T}\left[\mathcal{F}(D(G(t;\Theta_t);\Psi),0)\right], \tag{4}$$

and

$$\min_{\Theta_t} \quad \mathbb{E}_{t\sim\mathcal{X}_T}\left[\mathcal{F}(D(G(t;\Theta_t);\Psi),1)\right], \tag{5}$$

where $\mathcal{F}$ is the adversarial loss, Ψ are the parameters of the discriminator network, Θ_s are the parameters of the generator pretrained on the supervised task of leaf counting (c.f. Section 3.1), and Θ_t are the parameters of the generator for the target domain to be optimised (Θ_t are initialised with Θ_s). The function $\mathcal{F}$ takes two parameters as input: the output of the generator, namely the representation $\Phi_{s,t}(x)$ of an input image, and either '0' or '1'. The discriminator can be seen as a domain classifier (similar as in [8]), where '0' means target domain and '1' means source domain. When the parameters Θ_t are optimised, the network leans how to produce features as similar as the ones obtained from the source domain, minimising Equation (1).

Different choices of $\mathcal{F}$ lead to different adversarial network formulations [22]. The vanilla adversarial loss is the cross-entropy [13]. Other possible choices are the Wasserstein loss [2] or least squares loss [19]. In our experiments, we used either the cross-entropy as in [13], or the least squares loss [19] as in Equation (3) function $\mathcal{F}$. In general,

Layer	Name	Input Size	Output Size	Activation
Dense	DD1	2048	1024	LeakyReLU
Dense	DD2	1024	512	LeakyReLU
Dense	DD3	512	1	Linear
				Sigmoid

Table 2. Discriminator network architecture design. The activation of the last layer depends on the loss function. In case of MSE [19], linear activation has to be used. In the case of cross-entropy, sigmoid activation is necessary instead [13, 38].

the cross-entropy $H(p,q)$ is defined as:

$$H(p,q) = -\sum_k p_k \log(q_k). \tag{6}$$

3.2.1 Avoiding the posterior collapse

In semi-supervised or unsupervised learning, algorithms may learn wrong predictions on the target domain during training. In the worst case scenario, an algorithm can output the same prediction for any inputs: this phenomenon is known as *posterior collapse* [42]. To cope with it, we optimise the generator such that the distribution of the predicted leaf counting is similar to a prior distribution. Therefore, inspired by [34], we minimise the Kullback-Leibler divergence D_{KL}, jointly with the adversarial learning.

Let us assume that the output of the layer D3 (c.f. Table 1) l is a matrix $b \times m$, where b is the batch size, and m is the output dimension (50 in our case, Table 1). In particular, we define l_s the output of the D3 layer obtained from the source domain. Likewise, l_t is the output of the layer obtained from the target domain dataset. We can define $p_s \equiv p(l_s|x_s) = \frac{1}{b}\sum_{r=1}^{b} l_s(r,\cdot)$, where $l_s(r,\cdot)$ indicates the r-th row in the matrix (e.g., we are averaging the matrix l_s w.r.t. the batch size). In this way, we obtain the probability that the node l_k is active. Similarly, we can define $p_t = \frac{1}{b}\sum_{r=1}^{b} l_t(r,\cdot)$. In this way, $p_s(k)$ indicates the probability that the k-th node in the D3 layer can be activated. Therefore, assuming p_s to be our prior distribution over the labels in the target domain, we can minimise the following Kullback-Leibler divergence:

$$\min_{\Theta_t} D_{\mathrm{KL}}(p_s||p_t) = \min_{\Theta_t} \sum_{k=1}^{m} p_s(k)\log\frac{p_s(k)}{p_t(k;\Theta_t)}. \tag{7}$$

This loss function is minimised alternately with the optimisation of Equations (4) and (5).

3.2.2 Summary of the domain adaptation architecture

Generator: The architecture of the generator is the ResNet-50 [14] used during the pretraining [7, 10] and its weights

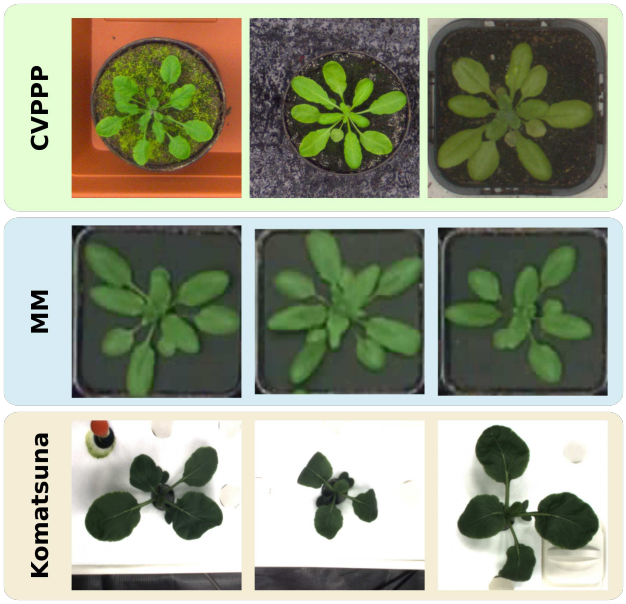

Figure 5. Samples of the images taken from the adopted datasets. *Top row*: CVPPP A1 (left), A2 (centre), and A4 (right). *Middle row*: Multi-modal imagery. *Bottom row*: Komatsuna images [40].

are initialised with the pretrained model learnt in the pretraining phase (c.f. Section 3.1).

Discriminator: The architecture of the discriminator is similar to the leaf counting network and we summarise it in Table 2. We set the parameter $\alpha = 0.2$ for the LeakyReLU activation function [43]. As shown in Figure 2, the output of the generator is provided to the discriminator as input.

Regressor: The regressor network is the LCN network without the SumLayer (c.f. Table 1). As for the discriminator, the output of the generator is provided to the regressor network as input.

3.3. Learning and Inference Process

The learning and inference processes are displayed in Figure 2 and we summarise them below.

Pretraining: The ResNet-50 with the LCN are trained on the source domain on the leaf counting task (supervised learning). The parameters of the ResNet-50 are initialised with the ImageNet weights [17] to reduce overfitting. The parameters of the ResNet-50 Θ_s are used for the domain adaptation step, whereas the parameters of the LCN are stored for the inference time.

Domain adaptation: The features of the source domain are extracted (last layer of the ResNet-50), using the parameters Θ_s, as well as the leaf count to compute the prior distribution. These data are needed to perform the domain adaptation on the target domain. Then, the parameters of the generator network Θ_t are initialised with Θ_s and Equations (4), (5) and (7) are alternately optimised. During this phase, the generator learns to output features as similar as the ones extracted from the images in the source domain. The aim of the adversarial learning is to minimise Equation (1). To improve the performance of the generator, we schedule two iterations of Equation (5) for every iteration of Equations (4) and (7).

Inference: The ResNet-50 is initialised with Θ_t, whereas the LCN uses the pretrained weights. Now, leaf counting can be performed using images of the target domain.

Dataset	Input Range	Multi-output	MSE
CVPPP*	$[0, 255]$	No	2.02 ± 0.16
CVPPP*	$[-1, 1]$	No	1.98 ± 0.28
CVPPP*	$[-1, 1]$	Yes	1.80 ± 0.23

Table 3. Comparison of different data normalisation approaches and the use of the proposed multi-output regression on the CVPPP* dataset. Overall, the use of data normalisation to $[-1, 1]$ and the multi-output regression obtain the best performance.

	Unrounded |DiC|	Fractional Diff
Single-output	1.02 ± 0.91	0.23 ± 0.14
Multi-output	0.77 ± 0.85	0.20 ± 0.14

Table 4. Single-layer vs. Multi-output evaluation w.r.t. two metrics: (i) difference in count on unrounded predictions; and (ii) difference between the unrounded and rounded prediction. The difference between single- and multi-output w.r.t. the two metrics is statistically significant (p-value < 0.0001).

4. Experimental Results

In our experiments, we used the following datasets, whose examples are displayed in Figure 5:

1. **CVPPP*:** we used the training Arabidopsis images of the CVPPP 2017 dataset A1,A2, and A4 [3, 21]. (We used the star * symbol to emphasise that we are not using the entire CVPPP dataset, as A3 was excluded as train and test set because it contains Tobacco plants – different species than the Arabidopsis.);
2. **MM:** we used the RGB Arabidopsis images of the Multi-Modal Imagery for Plant Phenotyping [5] (intra-species domain adaptation).
3. **Komatsuna**: we also used the Komatsuna plant dataset [40] (inter-species domain adaptation).

To assess the performance of our proposed model, we use the CVPPP evaluation metrics [10, 29]. Assuming $\epsilon_i = y_i - \hat{y}_i$ to be the difference between the ground truth and algorithmic prediction, the evaluation metrics are:

- *Difference in Count* [DiC]: $\frac{1}{N}\sum_{i=1}^{N} \epsilon_i$;
- *Absolute Difference in Count* [|DiC|]: $\frac{1}{N}\sum_{i=1}^{N} |\epsilon_i|$;

Method	*Intra-species* DiC↓	|DiC|↓	MSE↓	%↑	R^2 ↑	*Inter-species* DiC↓	|DiC|↓	MSE↓	%↑	R^2 ↑
FT	0.02 ± 0.68	0.40 ± 0.56	0.47	64	0.84	0.49 ± 1.21	0.94 ± 0.91	1.71	38	0.45
No FT	-13.91 ± 4.21	13.91 ± 4.21	211.22	0	0.09	-7.05 ± 2.42	7.05 ± 2.42	55.53	0	0.01
Ours (XE)	-0.81 ± 2.03	1.68 ± 1.39	4.78	20	0.02	$\underline{-0.78 \pm 1.12}$	$\underline{1.04 \pm 0.87}$	$\underline{1.84}$	$\underline{26}$	$\underline{0.46}$
Ours (LS)	$\underline{-0.39 \pm 1.49}$	$\underline{1.18 \pm 0.98}$	$\underline{2.36}$	$\underline{26}$	$\underline{0.34}$	-3.72 ± 1.93	3.72 ± 1.93	17.50	2	0.38

Table 5. Domain adaptation experiment for the CVPPP* to MM (*Intra-species*) and CVPPP* to Komatsuna (*Inter-species*) experiments. We compared our proposed unsupervised method with fine-tuning [FT] (supervised) and without fine-tuning (i.e., pretraining on CVPPP* and testing on MM without any adaptation. We used two adversarial losses: [XE] cross entropy, as in Equation (6); [LS] mean squared error, as in Equation (3). We underline the best results between the two variants of our proposed method.
↓: best values are closer to 0 – ↑: best values are closer to 1 (or 100 in the case of Percentage Agreement).

- *Mean Squared Error* [MSE]: $\frac{1}{N}\sum_{i=1}^{N}\epsilon_i^2$;
- *Percentage Agreement* [%]: $\frac{1}{N}\sum_{i=1}^{N}\mathbf{1}\left[\epsilon_i = 0\right]$;
- *Coefficient of Determination* [R^2]: measures the goodness of the predicted leaf count w.r.t. a fitted model (e.g., linear);

where $\mathbf{1}\left[\cdot\right]$ is the indicator function, which returns '1' if the error $\epsilon_i = 0$, otherwise returns '0'.

4.1. Setup

As in [7], we perform histogram normalisation as a preprocessing step and rescale all the images to a size of 320×320 pixels. In addition, we also normalise the input images from a range to $[-1, 1]$.[1] Before we train our network on a target domain, we validate our preprocessing approach and multi-output layer on the CVPPP* dataset [3, 21]. In our experiments, we set the size of the D3 layer to 50 (c.f. Table 1), as we know that the leaf count does not exceed 50 in the employed datasets. However, this can be extended to accommodate higher leaf counting, if needed.

We split the training dataset into 4 training-validation sets, as in [7]. The trained network was then tested on the CVPPP training set (A1, A2, and A4 only). As the test set is the same for all the splits, we computed the average and standard deviation of the MSE. In Table 3, we report the experimental results on CVPPP* data and it can be noticed that data normalisation to $[-1, 1]$ and the use of the multi-output regression obtain the best results. This validation experiment reinforces the benefits of our multi-output regression layer.

We also evaluated the capability of our network to perform (quasi-)integer predictions. It is important to point out that, as reported in [11], we round the network output to the nearer integer before computing the evaluation metrics. Here, we consider the unrounded leaf count prediction $\tilde{y}$, s.t. $\hat{y} = \mathrm{round}(\tilde{y})$. To assess the effectiveness of our model to output numbers close to their rounded version, namely $\hat{y} \approx \tilde{y}$, we introduce the following metrics:

- *Unrounded* $|DiC|$: $\frac{1}{N}\sum_{i=1}^{N}|y - \tilde{y}|$,
- *Fractional difference*: $\frac{1}{N}\sum_{i=1}^{N}|\hat{y} - \tilde{y}|$.

We evaluated these metrics on the CVPPP* testing dataset and we report the results in Table 4. Overall, our multi-output obtains more accurate predictions and close to the integer numbers. We run a paired t-test and the differences of the two methods are statistically significant (p-value $<$ 0.0001) for both of the metrics.

4.2. Domain Adaptation Experiments

In this section, we present two experiments to demonstrate the performance of our approach. We use the CVPPP* dataset as a starting point to pretrain the leaf counting network, then we adapt for the MM dataset (intra-species domain adaptation) and for the Komatsuna dataset (inter-species domain adaptation). We show experiments using two variants of our proposed method, which they differ from the adopted adversarial loss: (i) cross entropy (shortened as XE) as in [13, 38]; and the least squares loss (shortened as LS) as in LSGAN [19].

4.2.1 CVPPP* to MM (Intra-species DA)

Here, we use the CVPPP* dataset as source domain and the MM dataset as target domain. For the pretraining, we used the hyper-parameters setup as in [7, 10], whereas for the unsupervised domain adaptation part, used the setup as in [38]. We compare with finetuning [FT], which establishes a best performance bound as a fully supervised transfer learning approach. Specifically, the network is trained on CVPPP* and then fine tuned on MM. We also test on the MM dataset using the pretrained network on CVPPP*, without performing any domain adaptation step [No FT].

Experimental results are shown in Table 5 (*intra-species* columns). Overall, our method drastically outperforms the no-finetuning when the LS loss is used, with an MSE of 2.36, which edges closer to the best case of full supervision. In the other variant, where the XE loss was used, the

[1] GAN Hacks: https://github.com/soumith/ganhacks

trained model providing unreliable results on the target domain. Overall, when LS was used in our experiments, our method learned to perform leaf counting on an unseen target dataset of Arabidopsis plants (intra-species experiment). In the next section, we will perform a similar test, where the target dataset contains plants of different species (inter-species domain adaptation experiments).

4.2.2 CVPPP* to Komatsuna (Inter-species DA)

In this section, we perform domain adaptation from the CVPPP* to the Komatsuna dataset [40], which samples are displayed in Figure 5. In this experiment, we assess the ability of our method to perform inter-species domain adaptation for leaf counting. We run this experiment keeping the same setup as before, except for the loss function. Specifically, we used the cross-entropy loss (Equation (6)), as it proved to perform better than the MSE loss in this case. Moreover, we used the same data setup as in [10].

We report the experimental results in Table 5 (*inter-species* columns). It can be observed that the overall MSE is below 2 for our method and is very close to the finetuning performance (XE loss). When the LS loss was used, we noticed that the training was unstable providing unreliable results. We also compared with the case when no-finetuning or domain adaptation is performed and the predictions are very inaccurate. Therefore, our method is also able to perform leaf counting in the inter-species domain adaptation case, using the cross entropy loss, opposed as the inter-species experiment (Section 4.2), where the training with the LS loss provided the best results.

5. Conclusion

Deep neural networks have shown considerable success in addressing plant phenotyping analysis tasks. However, they require significant amount of annotated data to be trained effectively and they lack generalisation capabilities on previously unseen datasets. In this paper, we tackle these problems training a network to perform leaf counting on unseen unlabelled datasets.

Inspired by [38], we used the unsupervised adversarial domain adaptation framework to infer leaf count on another target dataset, which differs from the source training set. Our method is divided into three major steps (c.f. Figure 2). First, we pretrain the leaf counting network on the source domain. Then, we use adversarial training to learn image representation on the target domain as similar as the features extracted from the source domain. Lastly, the adapted network is used to count leaves on the target domain.

We presented two different experiments to assess the performance of our domain adaptation approach: (i) CVPPP* to MM [intra-species]; and (ii) CVPPP* to Komatsuna [inter-species]. Overall, as summarised in Table 5, our method can learn to count leaves on unseen data using the proposed unsupervised adversarial domain adaptation.

This paper introduces several novelties. Since labels in the target domain are unavailable and this might cause posterior collapse [42], we impose a prior distribution over the predicted leaf counting during the adaptation step. We minimise the Kullback-Leibler divergence between a prior and the predicted labels in the target domain during the domain adaptation step (c.f. Figure 2(2)).

To learn a distribution over the leaf counts, we introduce the multi-output regression, opposed to the state-of-the-art single-output regression [1, 7, 10, 16]. The multi-output regression layer uses a sigmoid function as activation, therefore it outputs a sequence of 0s and 1s. The final leaf counting is obtained with SumLayer, which agglomerates the predictions of the multi-output layer. Besides the learning of a leaf counting distribution, the multi-output layers brings another benefit. We can regularise the learning process to push towards (quasi-)integer leaf counting prediction, by constraining the first derivative of the sigmoid function in the multi-output layer.

During the domain adaptation, we do not assume direct access to the source dataset, but to the features extracted from it. We also do no assume direct access to the leaf count in the source domain, but we assume to know only the obtained distribution (after training). Therefore, if our network is pretrained on a private dataset, data do not need to be provided along with the network for the domain adaption step. In case our model is trained on restricted datasets, our approach promotes data confidentiality.

Future works should focus on stabilising the adversarial training. In our case, intra- and inter-species experiments had best performance using two different loss functions (MSE and cross-entropy respectively) to address the difficulty of training adversarial networks [25, 28].

Acknowledgements

This work was supported by the BBSRC grant BB/P023487/1 (http://chickpearoots.org) and also partially supported by The Alan Turing Institute under the EPSRC grant EP/N510129/1.

References

[1] S. Aich and I. Stavness. Leaf Counting with Deep Convolutional and Deconvolutional Networks. In *IEEE International Conference on Computer Vision Workshops Workshops*, pages 2080–2089. IEEE Computer Society, 2017. 2, 8

[2] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein GAN. *arXiv preprint arXiv:1701.07875*, 2017. 5

[3] J. Bell and H. M. Dee. Aberystwyth Leaf Evaluation Dataset. nov 2016. 6, 7

[4] C. Cortes, M. Mohri, and A. M. Medina. Adaptation based on generalized discrepancy. *Journal of Machine Learning Research*, 20(1):1–30, 2019. 3

[5] J. A. Cruz, X. Yin, X. Liu, S. M. Imran, D. D. Morris, D. M. Kramer, and J. Chen. Multi-modality imagery database for plant phenotyping. *Machine Vision and Applications*, 27(5):735–749, jul 2016. 6

[6] G. Csurka. Domain adaptation for visual applications: A comprehensive survey. *arXiv preprint arXiv:1702.05374*, 2017. 2

[7] A. Dobrescu, M. V. Giuffrida, and S. A. Tsaftaris. Leveraging multiple datasets for deep leaf counting. In *IEEE International Conference on Computer Vision Workshops Workshops*. IEEE Computer Society, sep 2017. 2, 3, 4, 5, 7, 8

[8] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016. 3, 5

[9] M. V. Giuffrida, F. Chen, H. Scharr, and S. A. Tsaftaris. Citizen crowds and experts: observer variability in image-based plant phenotyping. *Plant Methods*, 14(1):12, dec 2018. 1

[10] M. V. Giuffrida, P. Doerner, and S. A. Tsaftaris. Pheno-Deep Counter: a unified and versatile deep learning architecture for leaf counting. *The Plant Journal*, 96(4):880–890, nov 2018. 2, 3, 4, 5, 6, 7, 8

[11] M. V. Giuffrida, M. Minervini, and S. Tsaftaris. Learning to Count Leaves in Rosette Plants. In *CVPPP workshop*, page 13. British Machine Vision Association, 2015. 2, 7

[12] M. V. Giuffrida, H. Scharr, and S. A. Tsaftaris. Arigan: synthetic arabidopsis plants using generative adversarial network. In *Proceedings of the IEEE International Conference on Computer Vision Workshop – CVPPP*, pages 2064–2071, 2017. 1

[13] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014. 3, 4, 5, 7

[14] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June 2016. 3, 5

[15] J. Hoffman, S. Guadarrama, E. S. Tzeng, R. Hu, J. Donahue, R. Girshick, T. Darrell, and K. Saenko. Lsda: Large scale detection through adaptation. In *Advances in Neural Information Processing Systems*, pages 3536–3544, 2014. 2

[16] Y. Itzhaky, G. Farjon, F. Khoroshevsky, A. Shpigler, and A. B. Hillel. Leaf counting: Multiple scale regression and detection using deep cnns. In *CVPPP Workshop, BMVC*, 2018. 2, 8

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 6

[18] M. Long, Y. Cao, J. Wang, and M. Jordan. Learning transferable features with deep adaptation networks. In F. Bach and D. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 97–105, Lille, France, 07–09 Jul 2015. PMLR. 3

[19] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2794–2802, 2017. 5, 7

[20] M. Minervini, M. M. Abdelsamea, and S. A. Tsaftaris. Image-based plant phenotyping with incremental learning and active contours. *Ecological Informatics*, 23:35–48, 2014. 1

[21] M. Minervini, A. Fischbach, H. Scharr, and S. A. Tsaftaris. Finely-grained annotated datasets for image-based plant phenotyping. *Pattern Recognition Letters*, 81:80–89, oct 2016. 6, 7

[22] S. Nowozin, B. Cseke, and R. Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. In *NIPS*, 2016. 5

[23] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010. 2

[24] F. Radenovi, G. Tolias, and O. Chum. Fine-tuning cnn image retrieval with no human annotation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2018. 3

[25] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *International Conference on Learning Representations (ICLR)*, 2015. 8

[26] M. Ren and R. S. Zemel. End-To-End Instance Segmentation With Recurrent Attention. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 2

[27] B. Romera-Paredes and P. H. S. Torr. Recurrent Instance Segmentation. In B. Leibe, J. Matas, N. Sebe, and M. Welling, editors, *Computer Vision – ECCV Proceedings, Part VI*, pages 312–329. Springer International Publishing, Cham, 2016. 1, 2

[28] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. In *Advances in neural information processing systems*, pages 2234–2242, 2016. 8

[29] H. Scharr, M. Minervini, A. P. French, C. Klukas, D. M. Kramer, X. Liu, I. Luengo, J.-M. Pape, G. Polder, D. Vukadinovic, X. Yin, and S. A. Tsaftaris. Leaf segmentation in plant phenotyping: a collation study. *Machine Vision and Applications*, 27(4):585–606, may 2016. 1, 6

[30] S. Shekhar, V. M. Patel, H. V. Nguyen, and R. Chellappa. Generalized domain-adaptive dictionaries. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 361–368, 2013. 3

[31] A. K. Singh, B. Ganapathysubramanian, S. Sarkar, and A. Singh. Deep Learning for Plant Stress Phenotyping: Trends and Future Perspectives. *Trends in Plant Science*, 23(10):883–898, oct 2018. 1

[32] B. Sun, J. Feng, and K. Saenko. Return of frustratingly easy domain adaptation. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016. 3

[33] C. Sun, A. Shrivastava, S. Singh, and A. Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 3

[34] D. Tanaka, D. Ikami, T. Yamasaki, and K. Aizawa. Joint optimization framework for learning with noisy labels. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 5

[35] A. Tapas. Transfer learning for image classification and plant phenotyping. *International Journal of Advanced Research in Computer Engineering and Technology (IJARCET)*, 5(11):2664–2669, 2016. 3

[36] S. A. Tsaftaris, M. Minervini, and H. Scharr. Machine Learning for Plant Phenotyping Needs Image Processing. *Trends in Plant Science*, 21(12):989–991, dec 2016. 1

[37] S. A. Tsaftaris and H. Scharr. Sharing the Right Data Right: A Symbiosis with Machine Learning. *Trends in Plant Science*, nov 2018. 2

[38] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7167–7176, 2017. 2, 3, 4, 5, 7, 8

[39] J. Ubbens, M. Cieslak, P. Prusinkiewicz, and I. Stavness. The use of plant models in deep learning: An application to leaf counting in rosette plants. *Plant Methods*, 14(1), 2018. 3

[40] H. Uchiyama, S. Sakurai, M. Mishima, D. Arita, T. Okayasu, A. Shimada, and R. Taniguchi. An easy-to-setup 3d phenotyping platform for komatsuna dataset. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 2038–2045, Oct 2017. 6, 8

[41] D. Ward, P. Moghadam, and N. Hudson. Deep leaf segmentation using synthetic data. *Computer Vision Problems in Plant Phenotyping, held in conjuction with BMVC*, 2018. 1, 2

[42] M. Willetts, S. J. Roberts, and C. C. Holmes. Semi-Unsupervised Learning with Deep Generative Models: Clustering and Classifying using Ultra-Sparse Labels. jan 2019. 5, 8

[43] B. Xu, N. Wang, T. Chen, and M. Li. Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*, 2015. 6

[44] Y. Zhu, M. Aoun, M. Krijn, J. Vanschoren, and H. T. Campus. Data augmentation using conditional generative adversarial networks for leaf counting in arabidopsis plants. *Computer Vision Problems in Plant Phenotyping, held in conjuction with BMVC*, 2018. 1, 2