Study on Deep Convolutional Neural Networks for Leaf Counting *

Bharathi Chaudhury, Vasudha Joshi, S. S. Anand, and Pabitra Mitra

Indian Institute of Technology Kharagpur and SAC, ISRO

Abstract. The count of leaves in a plant gives valuable information related to the plant phenology and plant growth stage. It characterises the development and growth of plants. Manual leaf counting is a slow and labor-intensive task, therefore research to build an automated image-based leaf counting system is underway. Several deep learning based algorithms have been proposed for the task of leaf counting. Many of these approaches require per leaf segmentation and annotation for training. In our approach we have treated leaf counting as a regression problem requiring only the total leaf count per plant as annotation. We review different deep learning models that can be employed to leaf counting as a regression problem. Finally, a comparative study has been presented based on the experimental results on a benchmark dataset.

Keywords: plant phenotyping · deep learning · leaf count.

1 Introduction

Plant phenotyping is an emerging research area which deals with quantitatively measuring complex observable characteristics and traits of plants related to development, growth, yield, tolerance and physiology. Plant cultivation and genotyping has made enough advancements but, plant phenotyping is still in initial phase. Image based phenotyping is concerned with developing computer vision based systems for extracting data from plant images and accurately measuring plant phenotypes. The count of leaves per plant is one of the important plant phenotypes. Leaf count gives an idea of plant development, production yield and plant health. Manually counting the leaves is cumbersome. Therefore, studies are done to automate this task.

Detecting plant parts need efficient object detection and image segmentation techniques which makes automating leaf count a challenging problem. Moreover, leaves have varied shape, size and colour. They also grow, shift, rotate between frames. All these behaviour make their counting difficult. Image based leaf count is broadly solved in two ways: (i) leaf segmentation is used to count the number of leaves. (ii) leaf count by regression. Leaf segmentation method gave good results but required per leaf segmentation as annotation. Getting leaf segmentation is laborious and becomes more challenging when leaves are overlapping.

* Supported by SAC, ISRO.

Regression approach overcome this issue as it needs only per leaf count as annotation [1], [2], [3]. Approach adopted by [3] performed better than Computer Vision Problems in Plant Phenotyping (CVPPP 2020) challenge winner [1] and other current leaf segmentation techniques. In this paper, based on the techniques of [3], we perform comparative study of various CNN models for the purpose of leaf counting. The main contributions of this paper is

- We perform comparative study of various Deep CNNs for leaf counting.
- We tested our models on the benchmark dataset for leaf counting [4].
- Based on the results of our comparative study we propose two methods for leaf counting based on EfficientNetB3 [5] and MobileNet [6].

2 Related Work

In Deep Learning the task of counting is commonly achieved via detection, object segmentation, density estimation and regression. For leaf counting, segmentation is the most used approach. Counting via density estimation is not feasible for leaf counting because of occlusion, overlap and large scale variability of leaves. We briefly discuss the two directions explored for leaf counting problem.

Counting via segmentation: Inspired by the way humans count, [7] performed leaf counting via instance segmentation. [7] built an RNN based instance segmentation model which sequentially segments instances of the object from the image and retains spatial information within each image. They also derived a loss function for this problem. [8] proposed Recurrent Neural Network with attention mechanism for instance segmentation.

Regression: [2] utilised regression [9] to predict leaf counts. The article [1] used Support Vector Regressor on global image descriptors for obtaining the count. Input image I is converted to log polar domain I' w.r.t plant center. The log polar transformation orients all leaves parallel to each other. This ensures rotation and scale invariance and extracts good descriptors from the image in spite of large scale variability present in training set. A ratio of foreground pixels and background pixels is computed while scanning I' using a sliding window to identify the most informative region of I'. From these regions of interest $S \times S$ sized patches are extracted. These patches are clustered via K-means [10]. The clusters obtained undergo max-pooling to compute global descriptor for regression. Andrei [3] used modified ResNet50 [11] network for regression. To account for small dataset they pooled images from various sources.

3 Methodology

In this study, we apply transfer learning method on four networks i.e. VGG-16 [12], ResNet50 [11], MobileNet [6] and EfficientNetB3 [5] for counting leaves in plants.

3.1 Datasets

We train our model on CVPPP leaf counting dataset [4]. The dataset consists of 128, 31, 27, 624 images in four parts as A1, A2, A3, A4. RGB images of Arabidopsis and young tobacco plant constitute the dataset. The images in each dataset have resolutions ranging from 500x530 pixels in A1 to 2448x2048 pixels in A3.

3.2 Network Architectures

The study uses a deep neural network in the context of the regression task. In this approach we have used four pre-trained models: ResNet50, MobileNet, VGG16 and EfficientNetB3; all pre-trained in the ImageNet dataset.

Modified ResNet50 has the ability to generalize and it is crucial in the "wild setting" of problem statement. The ResNet architecture is easier to optimize than other deep learning architectures. The reference ResNet50 is modified as shown in Figure 1 by removing the last layer which was intended for classification, flattening the network and adding two fully connected layers FC1 and FC2 of 1024 and 512 nodes respectively followed by ReLU activations as [3]. We apply an L2 activation regularization on the FC2 layer to penalize the layer activity during training and prevent overfitting. FC2 goes into a fully connected layer containing a single node which acts as the leaf count prediction where the last three layers are responsible for the regression task. To evaluate the architecture, we first trained the network on each of the CVPPP datasets individually. We used mean squared error (MSE) as the loss function and Adam optimizer [13] with a learning rate of 0.0001. We trained with an early stopping criterion, based on the validation loss to avoid overfitting.

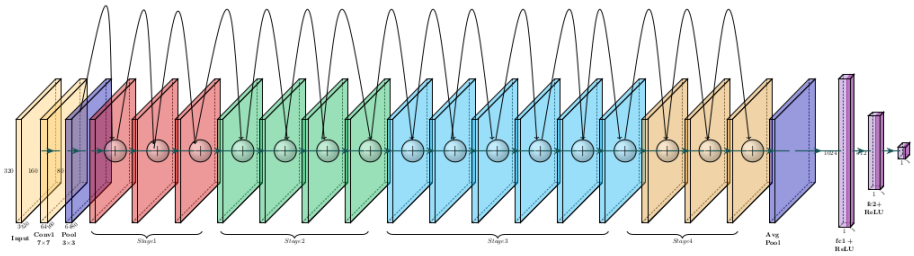

Fig. 1. Network Architecture using ResNet50

Modified VGG16 is a sequential network and it is easier to interpret the network. So we adopted the VGG-16 network as shown in Figure 2 [14]. The modification is similar to ResNet50 by removing the last layer which is meant

for classification (Figure 2). We flattened the output of the convolution layer and added 2 fully connected layers FC1 having 1024 nodes and FC2 having 512 nodes. Each fully connected layer is followed by ReLu activation. FC2 goes to an output layer containing a single node. The last three layers are responsible for the regression task. We maintained the same training and evaluation parameters as above.

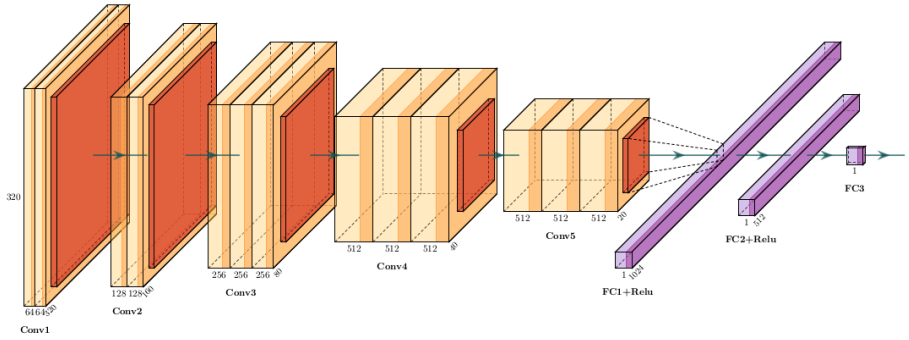

Fig. 2. Network architecture using VGG-16

Modified MobileNet is particularly useful for mobile based applications. Further to deploy the model in a mobile or embedded device it would be beneficial if the model is built upon fewer parameters and lesser size. The MobileNet [6] model is modified as shown in Table 1, by removing the last layer, flattening the network and adding three fully connected layers FC1, FC2 and FC3 of 1024,1024 and 512 nodes respectively followed by ReLU activation. Top layer containing a single node, is fed the output of FC3 and intended for leaf count prediction with softmax activation. We maintained the same training and evaluation parameters.

Modified EfficientNetB3 Since there are seven variants of EfficientNet CNN's we opted to use EfficientNetB3 as a good compromise between accuracy and parameter count. The EfficientNetB3 [5] model is modified as shown in Figure 3. We flattened the output of the convolution layer and added 2 fully connected layers FC1 having 1024 nodes and FC2 having 512 nodes. Each fully connected layer is followed by swish activation. FC2 goes to an output layer containing a single node. The last three layers are responsible for the regression task. We maintained the same training and evaluation parameters as of ResNet50.

3.3 Preprocessing and training

Each image was resized to $320 \times 320 \times 3$ pixels and normalized. The normalization changes the range of pixel intensity values as $[0-1]$. We used a random split from

Table 1. Network Architecture using MobileNet

Type/Stride	Filter Shape	Input Size
Conv/s2	3x3x3x32	320x320x3
Conv dw/s1	3x3x32 dw	160x160x32
Conv/s1	1x1x32x64	160x160x32
Conv dw/s2	3x3x64 dw	160x160x64
Conv/s1	1x1x64x128	80x80x64
Conv dw/s1	3x3x128 dw	80x80x128
Conv/s1	1x1x128x128	80x80x128
Convdw/s2	3x3x128 dw	80x80x128
Conv/s1	1x1x128x256	40x40x128
Conv dw/s1	3x3x256	40x40x256
Conv/s1	1x1x256x256	40x40x256
Conv dw/s2	3x3x256	40x40x256
Conv/s1	1x1x256x512	20x20x256
5xConv dw/s1		
5xConv/s1	3x3x512 dw	20x20x512
Conv dw /s2	3x3x512 dw	20x20x512
Conv/s1	1x1x1x512x1024	10x10x512
Conv dw/s2	3x3x1024 dw	10x10x1024
Conv/s1	1x1x1024x1024	10x10x1024
Avg Pool/s1	Pool 5x5	5x5x1024
FC	1024x1024	5x5x1024
FC	1024x1024	1x1x1024
FC	512x512	1x1x512
FC	512x1	1

the training set to 60% of the images used for training 20% for validation and 20% for (internal) testing. We trained four datasets (A1,A2,A3,A4) individually on each network.

Data Augmentation was performed while training all models. We used a generator which assigns training images a random affine transformation from a pool of random rotation from 0-170 degrees, zooming between 0-10% of the total image size and flipping vertically or horizontally. The training steps for each epoch was defined as the double the number of training images and the batch size per step was number of training samples . In total, the augmented dataset was 12 times the size of the original set per training epoch. Once the models are trained, obtaining test predictions just requires inputting the desired test images. The network output is not discrete, so we round the predictions to the nearest integer to get a leaf count.

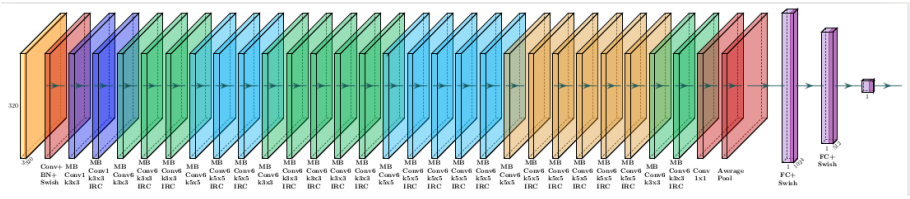

Fig. 3. Network architecture using EfficientNetB3

3.4 Evaluation Metric

To evaluate the performance of different network architectures we used the following two metrics

- Difference in Count (DiC). DiC computed as difference in predicted label and ground truth label, then averaging the computed differences $DiC = \#L_{Alg} - \#L_{GT}$
- $|DiC|$ is computed as the absolute difference in predicted label and ground truth label, then averaging the computed differences.
- MSE is Mean Square error. It is the square of difference between actual and predicted counts.
- Accuracy is total correct prediction obtained from the model divided by total size of training set.

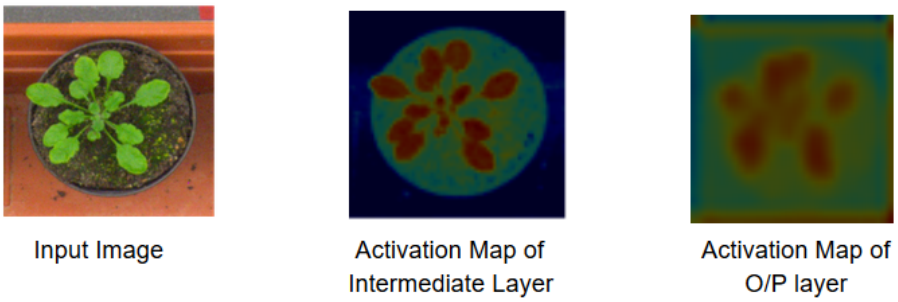

Fig. 4. Heatmap visualization of features used by ResNet50 in leaf count task

4 Experimental Results

The training of the four models (ResNet50, VGG-16, EfficientNetB3 and MobileNet) on the dataset is implemented in Python 3.6 and Tensorflow as back-end using Google Colab. The models took between 1 and 3 hours to train over an average of 50 epochs. We used the evaluation metrics as Difference in Count (Dic), absolute difference in count $|Dic|$, average loss and percent.

The modified CNN models as discussed in the methodology section are first trained on the training datasets. The training set accuracy and loss is given in Table 2. Average and standard deviation of predictions were computed for absolute and normal difference in count and included in Table 3. Fine tuned networks pretrained on ImageNet dataset gave better and consistent results than providing stronger annotations and using random initialization. So the learned ImageNet features were more valuable for this task than having the segmentation mask as an input. The four models are tested on four datasets individually and the results are shown in Table 3.

Results demonstrate that EfficientNetB3 has given good result on all four datasets. One natural question that arises is whether the network is learning to actually look at leaves to count or is influenced by the material in the background (i.e. it relies on background cues). We tested the network's ability to learn by imposing a heatmap on the images used in training. The activations of the layers are used to obtain the heatmap. Figure 4 displays the heatmap generated for Resnet50 [11] model as an example. We see that the networks do focus on the leafs as important features.

Table 2. Performance of deep learning architectures on the training set for each of the training dataset

Dataset/ Model	A1		A2		A3		A4	
	Accuracy %	MSE	Accuracy %	MSE	Accuracy %	MSE	Accuracy %	MSE
VGG-16	32.09	1.51	26.0	5.10	12.5	4.62	26.5	2.55
ResNet50	12.3	2.23	31.5	4.57	18.75	3.25	31.82	1.99
MobileNet	28.3	1.76	26.31	3.15	50.0	0.5	29.0	1.41
EfficientNetB3	33.3	1.51	16.04	2.97	37.5	0.63	57.6	0.49

Table 3. Performance of deep learning architectures on the test set for each of the training dataset

Dataset/ Model	A1			A2			A3			A4		
	DiC	$\|DiC\|$	MSE	DiC	$\|DiC\|$	MSE	DiC	$\|DiC\|$	MSE	DiC	$\|DiC\|$	MSE
VGG-16	-0.53 (1.34)	1.07 (0.957)	1.99	1.71 (2.86)	2.0 (2.67)	11.29	0.833 (1.86)	1.5 (1.38)	4.06	0.992 (1.63)	1.328 (1.37)	3.55
ResNet50	1.0 (1.14)	1.23 (0.89)	2.17	2.71 (2.76)	3.28 (2.05)	14.9	1.67 (1.25)	1.67 (1.25)	4.2	0.8 (1.62)	1.28 (1.28)	3.08
MobileNet	-0.46 (1.31)	1.07 (0.89)	1.77	1.28 (1.38)	1.57 (1.05)	3.99	0.67 (0.94)	1.0 (0.58)	1.129	-0.704 (1.13)	1.0 (0.87)	1.75
EfficientNetB3	0.46 (1.31)	1.0 (0.961)	1.79	1.42 (1.08)	1.42 (1.08)	3.0	-0.16 (1.34)	0.83 (1.07)	1.26	0.48 (1.085)	0.832 (0.85)	1.34

5 Conclusion

In this study, we present several deep learning approach for leaf counting in plants using modified VGG-16, ResNet-50,MobileNet,and EfficientNetB3 deep architecture respectively. We found from our study that EfficientNetB3 provides better accuracy than other networks. In resource constraint devices MobileNet can be used in the wild setting.

References

1. M. V. Giuffrida, M. Minervini, and S. A. Tsaftaris, "Learning to count leaves in rosette plants," 2016.
2. J.-M. Pape and C. Klukas, "Utilizing machine learning approaches to improve the prediction of leaf counts and individual leaf segmentation of rosette plant images," *Proceedings of the Computer Vision Problems in Plant Phenotyping (CVPPP)*, pp. 1–12, 2015.
3. A. Dobrescu, M. Valerio Giuffrida, and S. A. Tsaftaris, "Leveraging multiple datasets for deep leaf counting," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 2072–2079, 2017.
4. J. Bell and H. Dee, "Aberystwyth leaf evaluation dataset," *URL: https://doi.org/10.5281/zenodo*, vol. 168158, no. 17-36, p. 2, 2016.
5. M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International Conference on Machine Learning*, pp. 6105–6114, PMLR, 2019.
6. A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.
7. B. Romera-Paredes and P. H. S. Torr, "Recurrent instance segmentation," in *European conference on computer vision*, pp. 312–329, Springer, 2016.
8. M. Ren and R. S. Zemel, "End-to-end instance segmentation with recurrent attention," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6656–6664, 2017.
9. M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: an update," *ACM SIGKDD explorations newsletter*, vol. 11, no. 1, pp. 10–18, 2009.
10. C. Elkan, "Using the triangle inequality to accelerate k-means," in *Proceedings of the 20th international conference on Machine Learning (ICML-03)*, pp. 147–153, 2003.
11. K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.
12. K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
13. D. P. Kingma and J. Ba, "Adam: A methodfor stochastic optimization," in *International Conference onLearning Representations (ICLR)*, 2015.
14. A. Dobrescu, M. Valerio Giuffrida, and S. A. Tsaftaris, "Understanding deep neural networks for regression in leaf counting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 0–0, 2019.