International Journal of Applied Science and Engineering

Automatic monitoring of the growth of plants using deep learning-based leaf segmentation

Megha Trivedi[1], Abhishek Gupta[2*]

[1] *School of Electronics & Communication Engineering, Shri Mata Vaishno Devi University, Katra, Jammu and Kashmir, India*
[2] *School of Computer Science & Engineering, Shri Mata Vaishno Devi University, Katra, Jammu and Kashmir, India*

ABSTRACT

Plants are a source of food, medicines, fiber, fuel, etc. and are therefore crucial for our survival. Due to this, intensive care of plants should be done and it requires monitoring of their growth, size, yield, etc. However, manually monitoring such factors is often time-consuming and necessitates one to have in-depth knowledge of agriculture and plants. Thus, automatic systems for plant image analysis would be beneficial for practical and productive agriculture. Therefore, an automatic method is proposed for monitoring the growth of plants by first performing the segmentation of leaves in plant images and then calculating the segmented area. A deep learning-based architecture "U-Net" was used for the segmentation task. A benchmark dataset of 810 images was used to train and test the proposed deep learning network. The proposed model was trained within 3 hours and achieved a dice accuracy of 94.91% on the training set, 94.93% on the validation set, and 95.05% on the testing set. The proposed architecture was found very lightweight with fewer computations but achieved promising results as compared to other methods in the literature.

Keywords: Leaf segmentation, Deep learning, U-Net, Plant monitoring, Computer vision.

OPEN ACCESS

Received: November 2, 2020

Accepted: December 2, 2020

Corresponding Author:
Abhishek Gupta
abhishekgupta10@yahoo.co.in

Publisher:
Chaoyang University of Technology
ISSN: 1727-2394 (Print)
ISSN: 1727-7841 (Online)

1. INTRODUCTION

Plants are extremely essential as they are a source of food, medicines, fibre, fuel, etc. (Gupta et al., 2020). So, it becomes important to monitor various factors of a plant such as its growth, its yield, its size, etc. To be able to manually monitor such factors, one requires in-depth knowledge of agriculture and plants. In addition to this, it's very time consuming if plants are grown on a large scale. Therefore, it is considered to evolve automatic systems that can assist agriculturists, scientists, gardeners, etc. to monitor various factors of plants (Gupta et al., 2020). Such a system would be beneficial to help in understanding what measures and actions should be taken for improving the productivity of plants and crops. As a result, in the area of computer vision, there has been an increase in the number of researches on image-based plant phenotyping. Researchers are working on various plant datasets to devise methods with minimal human interaction to ease the study of visual phenotypes of plants. These researches are crucial as in the future, they can play a huge role in increasing the crop yields and meeting the food requirements of billions of people.

In recent years, numerous researchers have focused a lot on the area of plant phenotyping and performed various researches to solve a variety of problems such as plant disease detection, leaf segmentation, counting leaves, etc. A brief description of literature based on such researches and image segmentation is given in Table 1. Kumar et al. (2012) have made "Leafsnap" app that can provide assistance to amateur botanists, scientists, foresters, etc. The "leafsnap" app can categorize the tree species from an image of its leaf. To recognize a tree species, four steps were performed, namely classification, segmentation, extraction and comparison. Minervini et al. (2013) proposed an algorithm to automatically segment and analyse the plant images from phenotyping experiments of species Arabidopsis rosettes. They acquired the data by setting a static camera in a general laboratory that captures a number of plants at the same time. Their algorithm is based on a new vector valued active contour model that can incorporate prior knowledge which reflects the likelihood of a pixel to belong to a plant. Barbedo (2016) proposed a semi-automatic method that can detect plant diseases from asymptomatic tissues in plant leaves. Their algorithm is based on the manipulation of histograms of H and a in HSV and L*a*b respectively. The algorithm is semi-automatic as it needs to have human interaction to decide which among the H or a channel provides better differentiation. Ozturk et al. (2017) proposed a method to segment leaves using gray wolf optimizer based neural network. Yin et al. (2018) proposed a method to perform segmentation, alignment and tracking on a fluorescence plant video. They conducted their experiments on Arabidopsis thaliana and evaluated the method on the Leaf Segmentation Challenge (LSC) dataset (Minervini et al., 2015; Scharr et al. 2015; Dee et al., 2016). Kumar et al. (2019) proposed a method to extract the regions of leaves and count the number of leaves present in a plant image. They have divided the proposed method into three steps. In the first step, the image is enhanced by a statistical-based method. In the second step, the leaf regions are extracted from the plant using a graph-based image. In the third step, the Circular Hough Transform is applied to count the number of leaves in the plant image.

The objective of this study is to segment leaves from plant images having different backgrounds and illumination conditions with precise accuracy and less computational power requirements for training purposes and then monitor the growth of plants by computing the segmented area. The time taken to process an image by the methods proposed in some related articles that focus on segmenting leaves, increases with the size of the image. Thus, to overcome this limitation, we have trained the "U-Net" architecture (Ronneberger et al., 2015) for the segmentation task which is very lightweight, fast, and computationally less expensive. Also, every image is resized to 512 × 512 before feeding into the model and hence, the original size of the image does not affect the speed.

2. MATERIALS AND METHODS

2.1 Materials

The dataset used in this project was downloaded from the dataset page of Leaf Segmentation Challenge (Minervini et al., 2015; Scharr et al. 2015; Dee et al., 2016). It consists of four datasets, namely A1, A2, A3, and A4. Datasets A1, A2, and A4 consist of time-lapse images of the Arabidopsis plant, whereas, A3 dataset consists of images of the tobacco plant. A1 and A2 datasets were shot using a 7-megapixel Canon power-shot camera, whereas, A3 dataset was shot using Grasshopper cameras (Minervini et al., 2015). The A4 dataset's images were taken using the Photon Systems Instrument (PSI) platform's built-in camera (Dee et al., 2016). All the images were stored as lossless PNG files.

Each dataset has a different background and illumination condition. The A1 dataset consists of 128 images of 500 × 530 pixels with a complex background. In some images, moss is present in the soil which makes this dataset highly challenging as moss is of the same colour as the leaves. The A2 dataset consists of 31 images of 530 × 565 pixels. In some images, it contains extremely small-sized leaves which are difficult to detect. There are 27 images of 2448 × 2048 pixels in the A3 dataset. The low illumination conditions in a few images make this dataset complex. The A4 dataset consists of 624 images of 441 × 441 pixels. This dataset is highly varying as its images contain leaves of various sizes.

The data was split into training, validation, and testing sets such that 60% of images of each dataset were in the training set, 10% in the validation set and the rest 30% images in the testing set. So, there were a total of 486 images in the training set, 81 images in the validation set and 243 images in the test set.

2.2 Methods

This section deals with a detailed description of the applied methodology. The flow diagram of the applied methodology is shown in Fig. 1. Table 2 demonstrates a list of all layers used in the proposed architecture.

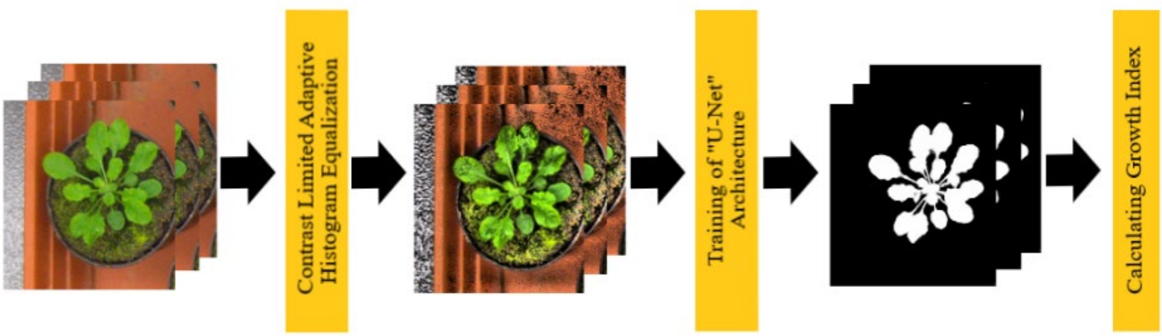


Fig. 1. Flow diagram of applied methodology

Table 1. Literature focusing on image segmentation and areas of plant phenotyping

Sr. No.	Reference	Objective	Dataset	Method	Result	Remarks
1.	Chen et al. (2002)	To propose an image segmentation algorithm.	Photographic images	Based on spatially adaptive colour and texture features	Images were segmented for an unlimited range of topics (people, nature etc.)	Colour segmentation is based on the adaptive clustering algorithm and texture analysis is based on an estimate of the energy of the coefficients of a wavelet decomposition
2.	Kumar et al. (2012)	To describe the Leafsnap app	Contains coverage of 184 tree species of North-eastern United States	Classifying, segmenting, extracting and comparing	Performs well on the real-world images from Leafsnap dataset	Leafsnap is a mobile app that can automatically identify plant species using visual recognition
3.	Minervini et al. (2013)	To propose an algorithm for the automated segmentation and analysis of plant images	Data acquired by them in a general laboratory	Combination of level set and learning- based segmentation	Accuracy (dice similarity coefficient) = 96.7%	The proposed method is able to properly segment images even with complicated and changing background
4.	Barbedo (2016)	To segment plant leaf disease symptoms	Images of 19 species containing 82 different diseases or examples of pest damage	Manipulation of histograms of the H and a from HSV and L*a*b color spaces respectively	r = 0.95	The algorithm is robust as it allows variation in symptom color, leaf color, etc.
5.	Ozturk et al. (2017)	To segment leaf images	26 leaf images	Gray wolf optimizer based neural network	Accuracy= 99.31%	Components from four different color spaces were used to train the neural network
6.	Yin et al. (2018)	To process fluorescence plant video	41 Arabidopsis Thaliana videos and LSC dataset (Minervini et al., 2015; Scharr et al., 2014)	Segmentation, alignment and tracking	Leaf Segmentation SBD accuracy on LSC dataset= 78.0% (±7.8)	When the overlap ratio between the leaves is greater than 23%, the algorithm recognizes two leaves as a single leaf
7.	Kumar et al. (2019)	To extract the regions of leaves and count the number of leaves present in a plant image	LSC dataset (Minervini et al., 2015; Scharr et al., 2014)	Statistical based, graph-based and Circular Hough Transform (CHT)	Accuracy (dice coefficient) = 95.4%, Counting accuracy= -0.7 (DiC)	The proposed method can process each image of A1 dataset or A2 dataset in approximately 2s, while it takes about 10s to process an image of A3 dataset

A. The Pre-Processing Stage

The images of A1 and A2 datasets were 4 channel RGBA images, while those of A3 and A4 datasets were 3 channel RGB images. Also, the size of the images was different for each dataset. Since the deep learning models expect all the input images to have the same dimensions, the four-channel RGBA images were converted to 3 channel RGB images and all the images were resized to 512 × 512 using the "skimage" python package. The original RGB images contained shadows and illumination effects which may have affected the performance of the deep learning model. Thus, contrast limited adaptive histogram equalization (CLAHE) was performed on all the images in order to enhance the local details even in the regions that were darker or lighter than most of the image. For an RGB image, the image is first converted into HSV colour space. Then CLAHE is performed on the V channel. The image is then returned after being converted back to RGB colour space (scikit-image). We did not apply an ordinary adaptive histogram equalization (AHE) on our dataset because it tended to overamplify noise in relatively homogenous regions (Yang et al., 2017). This happens due to the fact that in such regions, the histogram is highly concentrated. CLAHE uses clip-limit in order to reduce the noise amplification problem (Yang et al., 2017). Block size (or kernel size) and clip limit are the two important parameters of CLAHE as the quality of the enhanced image is mainly controlled by them (Yang et al., 2017). We performed CLAHE on our dataset using the "skimage" python package. We experimented with different settings for the clip limit, kernel-size, and nbins (number of gray bins for histogram) and after hyperparameter tuning, we set the kernel-size to 1/8 of image height by 1/8 of image width, nbins to 256, and the clip limit to 0.5 (scikit-image). The slope of the transformation function provides the contrast amplification in the given pixel value's area (Magudeeswaran et al., 2017). This is proportional to the slope of the neighbourhood CDF (Gabbiani et al., 2010) and consequently to that histogram's value which is present at that pixel value (Magudeeswaran et al., 2017). CDF can be defined as in Equation (1):

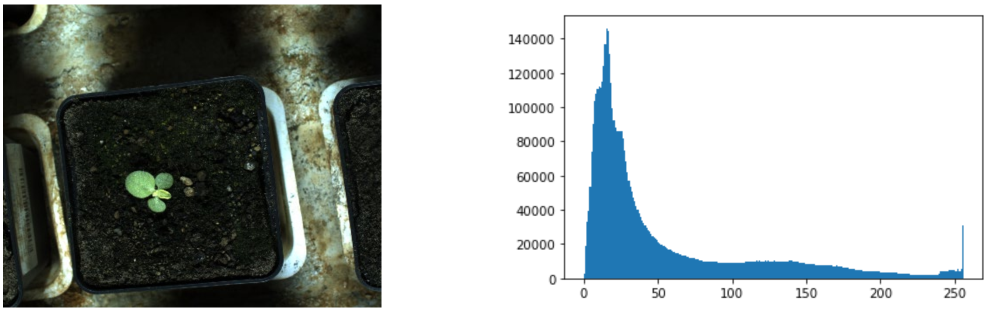
$$F(x) = P(X \leq x) \qquad (1)$$

where $P(X \leq x)$ represents the probability of a random variable X being less than or equal to a given value x (Gabbiani et al., 2010). CLAHE clips the histogram at 0.5 (clip limit) before computing the CDF (scikit-image). Due to this, the CDF's slope and therefore the transformation function's slope is limited. As a result, noise amplification is reduced (scikit-image). A comparison of an image along with its histogram before and after applying CLAHE is shown in Fig. 2.

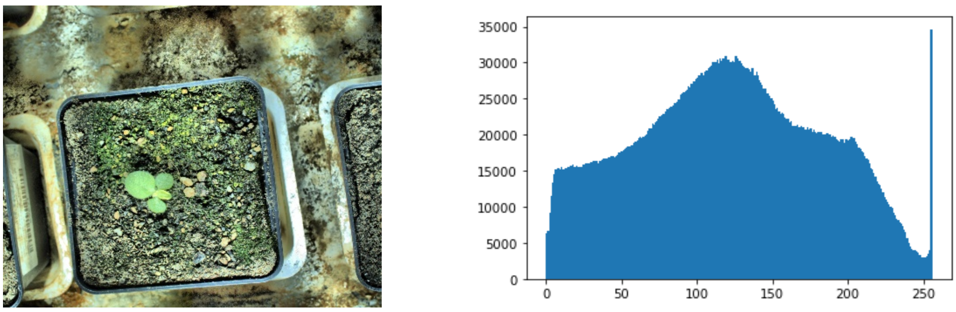
B. The Training of Deep Learning Model

The U-Net architecture was to train the model as it can be trained end to end on very few images and still perform well. Moreover, the network is quite fast as it requires less than a second to segment a 512 × 512 image on a GPU. The U-Net architecture was first proposed by Ronneberger et al. (2015).

The final model used is defined in Table 2. The first element of the output shape, i.e., the batch size was set to 1. The network has a U-shaped architecture and consists of a contracting path and an expansive path. Each convolutional



(a)



(b)

Fig. 2. Image along with its histogram (a) Before applying CLAHE and (b) After applying CLAHE

layer in the contracting path is followed by a ReLU activation function and a MaxPooling2D layer (Ronneberger et al., 2015; Kızrak, 2020). After the contraction, we get reduced spatial information and increased feature information. On the contrary, during the expansion, the feature and spatial information are combined through a sequence of Conv2D, UpSampling2D and skip connections (Add layers) with high resolution features from the contracting path (Ronneberger et al., 2015; Kızrak, 2020). In the last convolutional layer, we used the sigmoid activation function that can be defined as (TensorFlow) in Equation (2):

$$Sigmoid(x) = \frac{1}{1+e^{-x}} \quad (2)$$

This activation function was used in the last layer as we had only two classes in the output- one belonging to the leaf region (positive) and another belonging to the non-leaf region (negative), which is a binary classification problem. Here, the sigmoid activation function predicts the probability for a pixel to belong to a leaf region. If the predicted probability is greater than or equal to the threshold, i.e., 0.5, the pixel is assigned the positive class, else, the pixel is assigned the negative class. A label of 0 corresponds to the non-leaf region, whereas a label of 1 corresponds to the leaf region. The pixels which are assigned the positive class (label-1) look white coloured and the ones which are assigned the negative class (label-0) look black coloured in the output (segmented) image.

The number of epochs was set to 50 and ModelCheckpoint callback was used to save the model's weights for which the dice coefficient for the validation set was maximum. The model was trained with Adam as the optimizer with learning rate set to 0.001. Adam optimizer was first introduced by Kingma et al. (2014). We used this optimizer as it is computationally efficient and has little memory requirements (Kingma et al., 2014). Binary cross-entropy loss which was used as the loss function can be defined as in Equation (3):

$$J(w) = -\frac{1}{N}\sum_{n=1}^{N}[y_n log\hat{y}_n + (1-y_n)\log(1-\hat{y}_n)] \quad (3)$$

where N is the number of examples, y_n is the actual label (0 or 1), and $\hat{y}_n$ is the probability predicted by the model for a particular pixel to have a label of 1 (Nielsen, 2020). The model was implemented using the TensorFlow framework. We trained it using a GPU on Google Colab for less than 3 hours and got a training pixel accuracy of 98.81%, a validation pixel accuracy of 98.82% and a testing pixel accuracy of 98.69%.

C. Calculating Growth Index

The growth index is directly proportional to the area occupied by the leaves in a plant image, for which, the minimum and maximum values can be 0 and 1 respectively. The growth index of a plant is calculated from its segmented image using Equation (4). Since the output of our model is an image of 512 × 512 pixels, the total number of pixels in a segmented image is 2,62,144 Equation (5). A plant image having a higher growth index signifies more growth as compared to the one having a lower growth index.

$$G.I. = \frac{Number\ \ of\ \ Positive\ \ Pixels\ \ in\ \ the\ \ Segmented\ \ Image}{Total\ \ Number\ \ of\ \ Pixels\ \ in\ \ the\ \ Segmented\ \ Image} \quad (4)$$

Table 2. Different layers of the proposed architecture for the automatic segmentation of leaves

S. No.	Layer (type)	Stride	Params	Output shape
1.	Input layer	-	0	(Batch_size, 512, 512, 3)
2.	Conv2D	1	1792	(Batch_size, 512, 512, 64)
3.	Conv2D	1	36928	(Batch_size, 512, 512, 64)
4.	MaxPooling2D	1	0	(Batch_size, 256, 256, 64)
5.	Dropout	1	0	(Batch_size, 256, 256, 64)
6.	Conv2D	1	73856	(Batch_size, 256, 256, 128)
7.	Conv2D	1	147584	(Batch_size, 256, 256, 128)
8.	MaxPooling2D	1	0	(Batch_size, 128, 128, 128)
9.	Conv2D	1	295168	(Batch_size, 128, 128, 256)
10.	UpSampling2D	1	0	(Batch_size, 256, 256, 256)
11.	Conv2D	1	295040	(Batch_size, 256, 256, 128)
12.	Conv2D	1	147584	(Batch_size, 256, 256, 128)
13.	Add	1	0	(Batch_size, 256, 256, 128)
14.	UpSampling2D	1	0	(Batch_size, 512, 512, 128)
15.	Conv2D	1	73792	(Batch_size, 512, 512, 64)
16.	Conv2D	1	36928	(Batch_size, 512, 512, 64)
17.	Add	1	0	(Batch_size, 512, 512, 64)
18.	Conv2D	1	577	(Batch_size, 512, 512, 1)

Total params: 1,109,249
Trainable params: 1,109,249
Non-trainable params: 0

$$G.I. = \frac{Number\ of\ Positive\ Pixels\ in\ the\ Segmented\ Image}{262144} \quad (5)$$

3. RESULTS

To evaluate a segmentation model's performance, we cannot totally depend on the pixel accuracy as it is the percent of the correctly classified pixels in an image, and in our dataset, we have a lot of images in which there is a problem of class imbalance. Therefore, we also calculated the intersection over union and dice coefficient metrics. The formulae to compute these metrics are shown in Equation (6)-(8). For evaluating the task of monitoring the growth (calculating growth index), we calculated the mean absolute error (MAE), the formula for which is shown in Equation (9).

$$Pixel\ Accuracy\ (\%) = (TP + TN)/(TP + TN + FP + FN) \quad (6)$$

$$IoU\ (\%) = (TP/(TP + FP + FN)) \times 100 \quad (7)$$

$$Dice\ Coefficient\ (\%) = \left(\frac{2TP}{2TP+FP+FN}\right) \times 100 \quad (8)$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |Y_i - \hat{Y}_i| \quad (9)$$

where TP, TN, FP, and FN are the number of true positives, true negatives, false positives, and false negatives respectively. Y_i, $\hat{Y}_i$, and n are growth index calculated from ground truth, growth index calculated from predicted image, and the total number of examples respectively.

The pixel accuracy, intersection over union (also known as Jaccard Index), dice coefficient and MAE were separately computed for training, validation and test datasets and are shown in Table 3. Examples of images predicted by our model on A1, A2, A3 and A4 datasets of the test set along with their respective original RGB images and ground truths are shown in Fig. 3. Also, the calculated growth index is mentioned below the ground truths and the predicted images. The time taken by our model is less than a second to segment an image on a GPU.

4. DISCUSSION

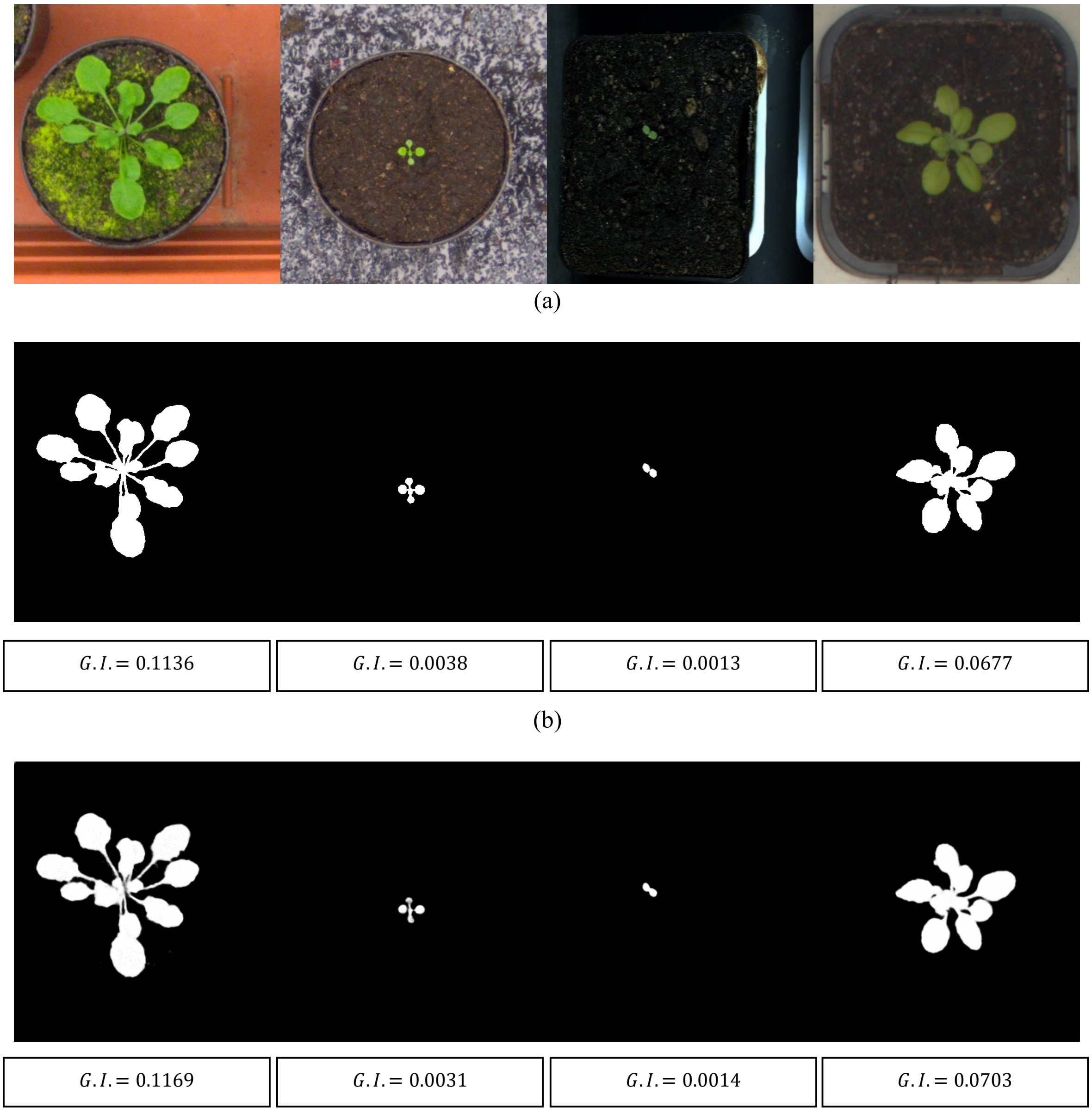
Numerous studies have been published that focus on automatic leaf segmentation. Ozturk et al. (2017) used a neural network with "gray wolf" as the optimizer to segment leaves. Their objective was to segment leaf images with different illumination conditions. They achieved a pixel accuracy of 99.31% which is slightly higher than that achieved by our model (training = 98.81%, testing = 98.69%). However, the dataset they used was different from ours and consisted of only 26 images. So, we can't compare the performance of our model with theirs.

The state of the art SLIC_Seg (Simple Linear Iterative Clustering superpixels segmentation) method (Minervini et al., 2016) uses SLIC superpixel to segment the leaf region. There is no need for training in this method. The SLIC_Seg method achieved a dice score of 94.6% on the A1 dataset, 87.5% on the A2 dataset and 79.4% on the A3 dataset. Our method performs better than the SLIC_seg method on every dataset as we achieved a dice score of 95.63% on the A1 dataset, 91.21% on the A2 dataset and 79.90% on the A3 dataset. Furthermore, their method is quite fast as it takes less than a second for each image in A1 and A2 datasets. However, the time taken on an image of the A3 dataset is 1-5 seconds, which may be due to the fact that an image of A3 dataset is of larger size as compared to an image of A1 and A2 dataset. Whereas, the time taken by our model on every image is always the same (< 1 second).

The objective of the study by Kumar et al. (2019) was to extract the regions of leaves and count the number of leaves present in a plant image. They have first enhanced the plant images using a statistical-based approach and then applied a graph-based technique to extract the leaf region. The dataset they used was the same as ours except for the fact that the A4 dataset was not included in it. They achieved a dice score of 95.4% on the leaf region extraction task which

Table 3. Performance of our model on various datasets for segmentation task

	Datasets	Pixel accuracy (%)	IoU (Jaccard Index) (%)	Dice coefficient (%)	MAE
Train	A1	98.37	91.87	95.76	0.0028
	A2	99.42	84.90	91.56	0.0007
	A3	99.31	82.23	88.79	0.0010
	A4	98.85	90.90	95.13	0.0013
	Total	**98.81**	**90.56**	**94.91**	**0.0015**
Validation	A1	98.39	92.12	95.90	0.0032
	A2	99.64	80.62	88.89	0.0009
	A3	98.27	83.24	90.69	0.0045
	A4	98.89	91.02	95.23	0.0014
	Total	**98.82**	**90.53**	**94.93**	**0.0018**
Test	A1	98.16	91.63	95.63	0.0031
	A2	99.26	84.34	91.21	0.0013
	A3	98.64	72.36	79.90	0.0045
	A4	98.77	91.87	95.69	0.0016
	Total	**98.69**	**90.98**	**95.05**	**0.0019**



(c)

Fig. 3. (a) RGB images, (b) Ground truths with growth index and, (c) Predicted images with growth index of A1, A2, A3 and A4 datasets

is slightly higher than that achieved by our model (training = 94.91%, testing = 95.05%). However, to fully process an image, their method takes 2 seconds on A1 and A2 datasets and 10 seconds on an A3 dataset image. Whereas, our model takes less than a second to segment any image, irrespective of which dataset it belongs to or what its original size is as every image is resized to 512 × 512 before giving as an input to our model.

We have proposed a model which is able to segment leaves from plant images with high pixel accuracy. Furthermore, our model performs well on other metrics (dice score and Jaccard index) too. We have also calculated the growth index from the segmented image, which can be used to monitor the growth. Since we have used a dataset that consists of images of four plants, each having different lightning and background, our model is quite robust. Additionally, since we have used the "U-Net" architecture, our model is pretty fast too. Also, unlike other methods, the speed of the model does not vary with the original size of an image. This is due to the resizing of the images before feeding to the model.

The deep learning model was trained using images in which there was only a single plant present. Also, the images were of Arabidopsis and tobacco plants only and they both have green leaves. The proposed model is not validated on the images where there are multiple plants in an image or where the leaves of a plant are not green. To use the model in such cases, the model will require training on a similar dataset. Thus, a limitation of the proposed method is that the U-Net architecture will require retraining (keeping all the hyperparameters same) if the images from which the growth has to be monitored is different (in terms of the number of plants present in an image and colour of leaves) from the ones the model is trained on. However, retraining the proposed model is not a tedious task as the U-Net architecture is quite fast and performs well even if trained on a small dataset. Future research work may focus on overcoming this limitation up to an extent by training the model using a much diverse dataset of plant images with different coloured leaves and multiple plants present in an image.

5. CONCLUSION

We have proposed a deep learning U-Net architecture for automatic leaf segmentation using plant images and then also calculated the growth index to monitor the growth of plants. This "U-Net" architecture was found very lightweight with less computational power requirements but achieved a promising pixel accuracy. The other metrics (Dice coefficient and Jaccard Index) judging the proposed method were found satisfactory considering the method of automatic segmentation of leaves. The mean absolute error which was used to measure the error in the growth index made it evident that the proposed method can be used to monitor the growth of plants with precise accuracy. The proposed method can be used where there is a need to perform automatic monitoring of the growth of plants. In the future, the proposed method can be further extended to monitor the growth of multiple plants at once which can be very useful in large-scale farms.

ACKNOWLEDGEMENT

The Author Dr. Abhishek Gupta would like to acknowledge the work completed under the project grant RP-103 received from the UGC (University Grants Commission).

REFERENCES

Barbedo, J.G.A 2016. A novel algorithm for semi-automatic segmentation of plant leaf disease symptoms using digital image processing, Tropical plant pathology, 41, 210–224.

Bell, J., Dee, H. 2016. Aberystwyth leaf evaluation dataset: A plant growth visible light image dataset of Arabidopsis thaliana. Zenodo.

Chen, J., Pappas, T.N., Mojsilovic, A., Rogowitz, B. 2002. Adaptive image segmentation based on color and texture. In Proceedings. International Conference on Image Processing, 3, 777–780. IEEE.

Cumulative distribution function. ScienceDirect. Available: https://www.sciencedirect.com/topics/mathematics/cumulative-distribution-function. [Accessed 8th August, 2020].

Gabbiani, F., Cox, S. 2010. Mathematics for neuroscientists. Amsterdam: Elsevier/Academic Press.

Gupta, A., Prakash, D. 2020. A fast and efficient color model for automatic monitoring of plants based on leaf images. Journal of Critical Reviews, 7, 2398–2404.

Kingma, D.P., Ba, J. 2014. Adam: A method for stochastic optimization. arXiv:1412.6980.

Kızrak, A. 2020. Deep learning for image segmentation: U-Net architecture. Heartbeat. Available: https://heartbeat.fritz.ai/deep-learning-for-image-segmentation-u-net-architecture-ff17f6e4c1cf. [Accessed 8th August, 2020].

Kumar, J.P., Domnic, S. 2019. Image based leaf segmentation and counting in rosette plants. Information Processing in Agriculture, 6, 233–246.

Kumar, N., Biswas, A., Jacobs, D.W., Kress, W.J., Lopez, I., Soares, J.V.B. 2012. Leafsnap: A computer vision system for automatic plant species identification. Presented at the in The 12th European Conference on Computer Vision (ECCV).

MA, J., Fan, X., Yang, S.X., Zhang, X., Zhu, X. 2017. Contrast limited adaptive histogram equalization based fusion for underwater image enhancement. Preprints, 2017030086 (doi: 10.20944/preprints201703. 0086.v1).

Magudeeswaran, V., Singh, J.F. 2017. Contrast limited fuzzy adaptive histogram equalization for enhancement of brain images. International Journal of Imaging Systems and Technology. 27, 98–103.

Minervini, M., Abdelsamea, M.M., Tsaftaris, S.A. 2013. Image-based plant phenotyping with incremental learning and active contours. Ecological Informatics, 23, 35–48.

Minervini, M., Fischbach, A., Scharr, H., Tsaftaris, S.A. 2015. Finely-grained annotated datasets for image-based plant phenotyping. Pattern Recognition Letters, 1–10.

Minervini, M., Scharr, H., Fischbach, A., Tsaftaris, S.A. 2014. Annotated image datasets of rosette plants. Technical Report No. FZJ-2014-03837, Forschungszentrum Jülich, GmbH, Jülich, Germany.

Nielsen, M. 2020. Neural networks and deep learning. [online] Neuralnetworksanddeeplearning.com. Available at: http://neuralnetworksanddeeplearning.com/chap3.html [Accessed 3 March 2020].

Ozturk, S., Akdemir, B. 2017. Automatic leaf segmentation using grey wolf optimizer based neural network. Electronics, 1–6.

Plant-phenotyping.org. 2020. Plant phenotyping datasets. [online] Available at: https://www.plant-phenotyping.org/datasets-home [Accessed 31 July 2020].

Ronneberger, O., Fischer, P., Brox, T. 2015. U-Net: Convolutional networks for biomedical image segmentation. In International Conference on Medical image computing and computer-assisted intervention, 234–241. Springer, Cham.

Scharr, H., Minervini, M., French, A.P., Klukas, C., Kramer, D.M., Liu, X.M. et al. 2016. Leaf segmentation in plant phenotyping: a collation study. Machine Visions and Applications 27, 585–606.

Scikit-image.org. 2020. Module: exposure — skimage v0.19.0.dev0 docs. [online] Available at: https://scikit-image.org/docs/dev/api/skimage.exposure [Accessed 8 August 2020].

Sharma, S., Gupta, A. 2020. A review for the automatic methods of plant's leaf image segmentation. International Journal of Intelligence and Sustainable Computing 1, 101–114.

TensorFlow. 2020. tf.keras.activations.sigmoid TensorFlow Core v2.4.1. [online] Available at: https://www.tensorflow.org/api_docs/python/tf/keras/activations/sigmoid [Accessed 9 August 2020].

Yin, X., Liu, X., Chen, J., Kramer, D.M. 2018. Joint multi-leaf segmentation, Alignment, and Tracking for Fluorescence Plant Videos. IEEE Transactions on Pattern Analysis and Machine Intelligence, 40, 1411–1423.