Deep Learning techniques for plant growth classification

Dmitrii Vypirailenko [1] **Elizaveta Kiseleva** [1]

Abstract

The problem of weed stage and species classification and detection is of high importance in agriculture and smart farming. In this work, we explore various CNN architectures and deep learning approaches for the task of weed stages classification. We used the labeled data of different weed species divided by the number of the leaves. Our model was able to beat the baseline of previous articles on this problem and dataset.

Our project GDrive link and Github.

1. Introduction

The ever-growing population and demand for food are forcing agriculture to develop ways to achieve greater productivity.

Plant growth modeling is important to increase farm productivity. A system for accurate modeling and classification of plant growth stages will help to understand better the patterns associated with care, and fertilization will allow you to achieve a larger crop.

Another important thing is weed control. Weeds are a nuisance on the farmland as they compete for plant nutrients. When conducting chemical or mechanical treatment of weeds, it is necessary to know the composition and growth stage of the weed species to apply herbicides or mechanical methods correctly with minimal harm to crops and minimal use of energy.

The above facts indicate the need to create an automatic system for recognizing the stages of growth and control of weeds. In the ideal case, such a system should separate weeds from crops, as well as determine the stages of their growth and distinguish between different species of them.

2. Related works

The problem of determining the stage of growth and classification of weeds using deep learning methods has not yet been widely studied. Most of the existing articles focus on the use of convolutional neural networks for the classification and recognition of weed crops, as well as their distinction from cereal crops.

Previously, CNNs have been widely used for various computer vision tasks connected with agriculture, such as plant phenotypes and species classification (2; 8). Also, CNNs were widely used for weed detection and segmentation (3). CNNs are effectively solving these problems due to their ability to extract important features from images. Most of the articles use standard convolutional architectures, such as VGG-16 or ResNet. Obviously, the more classes there are, the harder the task becomes.

One approach to classifying weed growth stages is to classify them by the number of leaves. Several works study the approach of using convolutional and deconvolutional networks for the plants classification based on the amount of the leaves (7; 9; 1). The main challenge of these tasks is the accuracy of classification, which could be pretty challenging for the datasets with a large amount of imbalanced classes. One of the promising approaches is transfer learning. (4)

The main goal of this work is to create an efficient end-to-end deep learning model on the task of weeds growth stage classification task.

3. Dataset

3.1. Dataset

The data are weed photographs taken in vivo. Their peculiarity is that images of weeds are deliberately included in the data set, where the leaves are poorly visible or covered by extraneous plants. In addition, photos were taken on ordinary phones and custom cameras, so they do not have very high image quality. These images were collected over three growing periods and cover a total of 18 species of weeds or families. These images were taken in various fields in all regions of Denmark; thereby covering various soil types, image resolution and lighting conditions. One of the critical factors that can affect the ability of the convolutional neural network to count the number of leaves is plants overlapping each other.

To make the convolutional network stable in the study of images with occluded leaves, it must be presented with images containing overlapping leaves at the training stage.

In total, 9649 image samples were obtained for various types of weeds, with each image manually classified by experts by type and stage of growth. There are two standards for leaf counting: one where only real leaves are counted, and the other - both cotyledon leaves and real leaves. Here, the leaves include cotyledon leaves. Nine classes are used here: 1-leaf, 2-leaves, 3-leaves, ..., 8-leaves and more than 8-leaves. Data is highly unbalanced, see Fig. 1. Images are publicly available at https://vision.eng.au.dk/leaf-counting-dataset/.

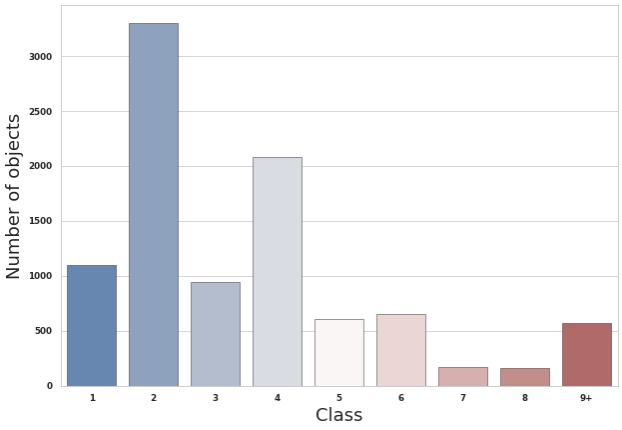

Figure 1. Number of objects by class

3.2. Examples

In Fig. 2 You can see examples of images of weeds. You may notice that they are all of different sizes and some of poor quality.

Figure 2. Randomly selected examples of weed images from a dataset

Cases of partially hidden weeds are shown in Fig. 3.

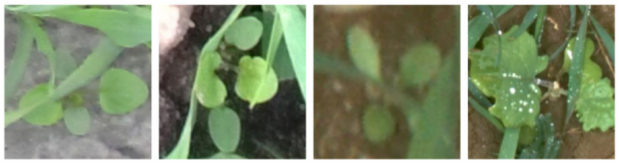

Figure 3. Low quality or poorly visible of plant leaves images

4. Methodology

4.1. Preprocessing and Augmentation

Preventing overfitting is an essential problem to overcome, especially for this task, because we only have 160 training images per class 7 and 8.

First, during training, each time when an image is fed to the model, a 128×128 resize generated from the image will be used instead. During validation and testing, we use the same. Then augment data by horizontal and vertical flipping, translation and rotation. We also used random contrast correction as an augmentation method. We set the scaling factor to be $random([0.9, 1.08])$ at every batch, and clip pixel values to the range $[0, 255]$ after correction process to guarantee valid augmented images. Another data augmentation method we used is random Affine transformation.

When an image is fed to the model, every augmentation method is applied to this image. In this way, the total number of training examples is increasing. Augmentation results can be seen in Fig. 4. We also use weights in cross entropy loss, to overcome imbalance in the dataset.

4.2. Optimization Algorithm

The primary optimization algorithm used was Adam(6), as implemented by Pytorch. Learning rate $= 0.007$

4.3. DenseNet architecture

To achieve the desired result, we tried many different architectures of neural networks: AlexNet, VGG16, ResNet18, ResNet50, Wide ResNet, SqueezeNet. Architecture showed the best results DenseNet169 (5) Fig. 5.

4.4. Transfer Learning

Transfer learning is a machine learning and deep learning technique where a model trained on one task is re-purposed on a second related task.

The base DenseNet169 model with the same parameters and image augmentation was trained on the data from the Kaggle competition. This is also classification a classification task, but there are 12 classes corresponding to the type of plant.

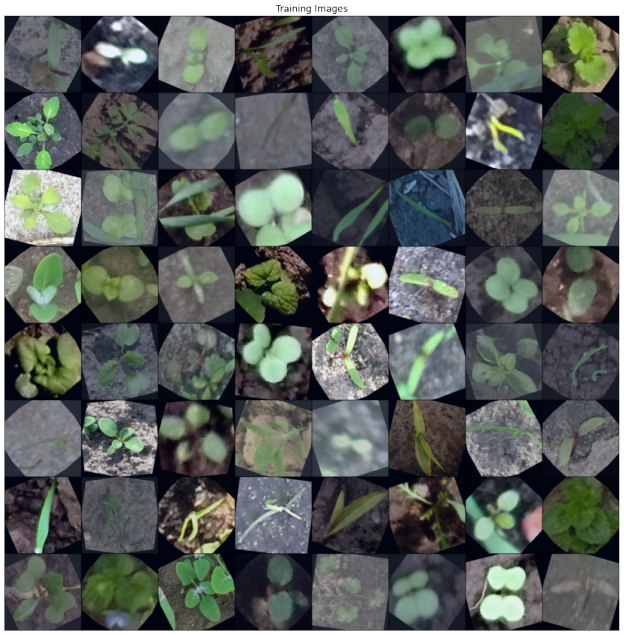

Figure 4. Augmentation results

The types of the plants are similar to ones we have in our main datasets, except 3 classes.

Further, this model was trained on our data to determine the number of leaves and perform growth stage classification task.

5. Experiments and Results

5.1. DenseNet169

As mentioned above, we used the architecture of DenseNet169 with parameters:

$criterion = CrossEntropyLoss$

$optimizer = Adam(lr = 0.007)$

$scheduler = StepLR(stepsize = 40, gamma = 0.3)$

$epochs = 100$

Also, we changed the last layer to fit our classification task to 9D.

The results can be seen on Fig. 6.

The table 1 presents the best results of verified models. As a baseline, we took the results of the work described in the article (9).

In Fig. 7 the confusion matrix of this model is presented. It is easy to see that weeds with 2 and 4 leaves are recognized best. The first third and sixth grades are recognized much worse. The fifth, seventh and eighth are very poorly recognized.But if you look more closely, the main errors are concentrated in neighboring classes. If the model is wrong, then often just one leaf. We think this is due to the high imbalance of classes, and the expansion of the sample will help solve this problem.

Layers	Output Size	DenseNet-169	DenseNet-201
Convolution	112 × 112	7 × 7 conv, stride 2	
Pooling	56 × 56	3 × 3 max pool, stride 2	
Dense Block (1)	56 × 56	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 6$
Transition Layer (1)	56 × 56	1 × 1 conv	
	28 × 28	2 × 2 average pool, stride 2	
Dense Block (2)	28 × 28	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 12$
Transition Layer (2)	28 × 28	1 × 1 conv	
	14 × 14	2 × 2 average pool, stride 2	
Dense Block (3)	14 × 14	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 48$
Transition Layer (3)	14 × 14	1 × 1 conv	
	7 × 7	2 × 2 average pool, stride 2	
Dense Block (4)	7 × 7	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 32$
Classification Layer	1 × 1	7 × 7 global average pool	
		1000D fully-connected, softmax	

Figure 5. DenseNet Architecture

Figure 7. Confusion matrix for DenseNet169 model

5.2. Transfer Learning

The base model was trained using Kaggle data and showed good results. The accuracy was 92%. Further, this model was trained on our data to determine the number of leaves.

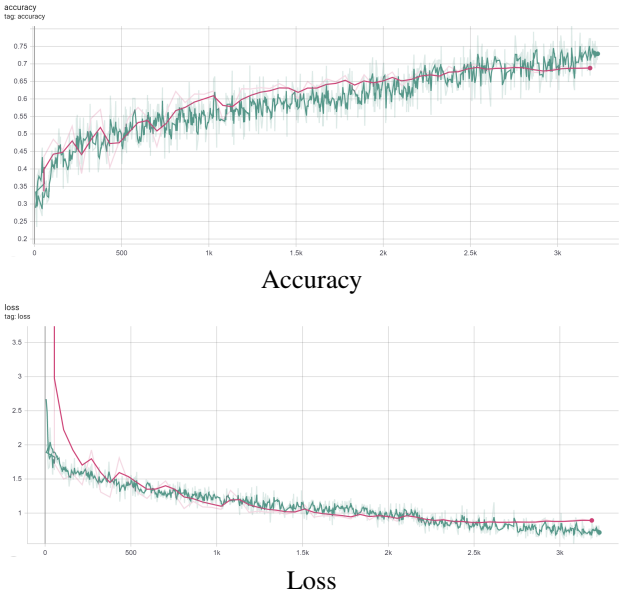

Figure 6. Result of training DenseNet169 model

The process of further training the model can be seen in Fig. 8. The learning outcomes are presented in the table1. From Fig. 8 and Fig. 6 it can be seen, that transfer learning helps with initial loss and accuracy. Also, in our experience, transfer learning model was more stable in training then the standard DenseNet.

In Fig. 9 can be seen the confusion matrix for the DenseNet169 model with transfer learning. It is interesting to note that classes that are well recognized by the simple DenseNet169 have become even more recognizable. Some classes ceased to be found at all. Most likely this is due to the fact that the model did not have time to adjust to the new data of some classes due to their small number. We are again faced with the problem of high class imbalance.

Model	Best Accuracy
ResNet18	65.71%
VGG11	65.95%
Inception_v3	68.27%
WideResNet50	69.34%
DenseNet121	70.53%
DenseNet169 (transfer learning)	**71.16%**
DenseNet169	**71.81%**
Inception_v3, *Baseline*	70%

Table 1. Table of accuracy of different models

6. Conclusion

In our work, we successfully created a deep learning model for the classification of weeds growth stages based on the number of leaves. Moreover, our model was better than the model from the main article we referenced with fewer parameters and layers than the original one. We hope that this research can serve as a reliable basis for our future work. The transfer learning approach also showed good results, and we think that it has excellent potential for this particular task, especially if we manage to find a bigger dataset for the base model.

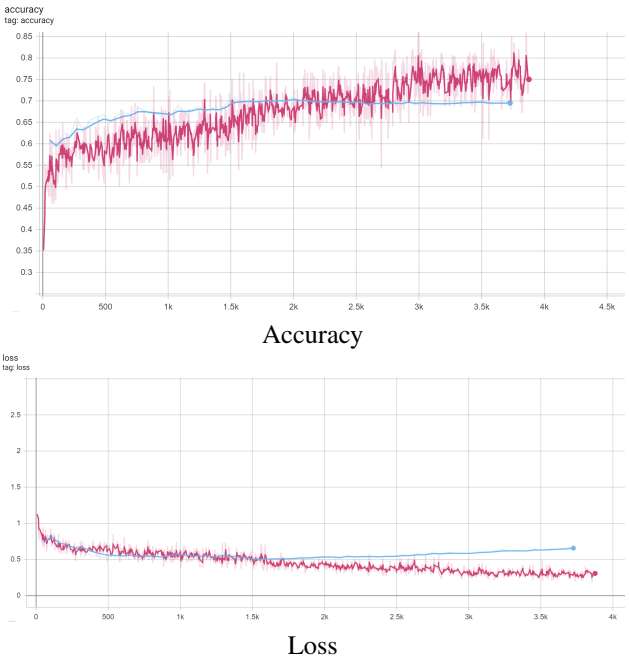

Figure 8. Result of training DenseNet169 model with transfer learning

The ultimate goal is to create a complete system for efficient segmentation and detection of plants and weeds, as well as the classification of their species and growth stages in real-time mode.

6.1. Future work

As the next steps in the development of this project, we see the creation of algorithms for detecting weeds on beds of cultivated plants Fig. 10 and classifying them by type. Based on this work, you can create an application that will help both ordinary households and large agricultural complexes to combat weeds effectively.

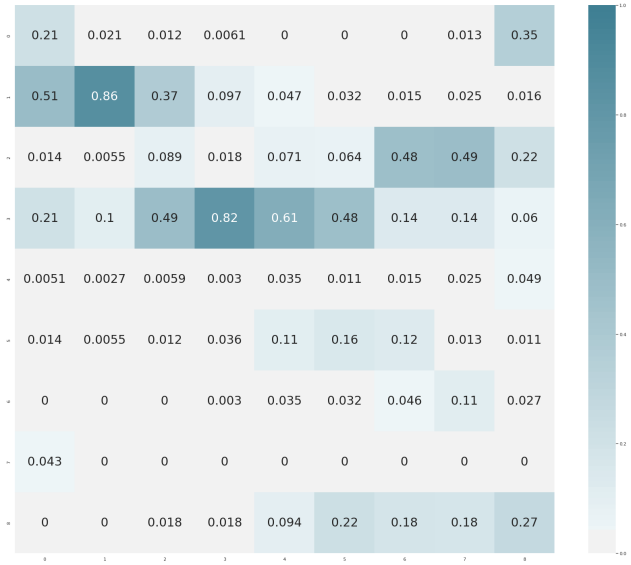

Figure 9. Confusion matrix for DenseNet169 model with transfer learning

Figure 10. Pepper Bed Weed Detection Example

7. Contribution

Dmitrii Vypirailenko

Implemented transfer learning code, studied the papers, provided the metrics and plots. Writed points 1,2, and 6 from the paper (report preparation).

Elizaveta Kiseleva

Initial idea, implementation of the baseline models training cycle, data preparation and augmentation, report, presentation preparation and Github repository preparation.

8. Third party code

Most of the code for the experiments was written independently. We relied on some code from HW 2. Some code for data preparation of transfer learning experiments was taken from https://www.kaggle.com/c/plant-seedlings-classification.

References

[1] Shubhra Aich and Ian Stavness. Leaf counting with deep convolutional and deconvolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 2080–2089, 2017.

[2] Belal AM Ashqar, Bassem S Abu-Nasser, and Samy S Abu-Naser. Plant seedlings classification using deep learning. 2019.

[3] Mads Dyrmann, Rasmus Nyholm Jørgensen, and Henrik Skov Midtiby. Roboweedsupport-detection of weed locations in leaf occluded cereal crops using a fully convolutional neural network. *Advances in Animal Biosciences*, 8(2):842–847, 2017.

[4] Mostafa Mehdipour Ghazi, Berrin Yanikoglu, and Erchan Aptoula. Plant identification using deep neural networks via optimization of transfer learning parameters. *Neurocomputing*, 235:228–235, 2017.

[5] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

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

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

[8] Sarah Taghavi Namin, Mohammad Esmaeilzadeh, Mohammad Najafi, Tim B Brown, and Justin O Borevitz. Deep phenotyping: deep learning for temporal phenotype/genotype classification. *Plant methods*, 14(1):66, 2018.

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