Using Convolutional Neural Network to Detect and Count Individuals on Eucalyptus Plantation

Luiz Antonio Macarini[1], **Antonio C. Sobieranski**[1,2]

[1]Brazilian National Institute for Digital Convergence (INCoD)
Federal University of Santa Catarina (UFSC)
Florianopólis – SC – Brazil

[2]Department of Computing
Federal University of Santa Catarina (UFSC)
Araranguá – SC – Brazil

Abstract. *Deep Learning constitutes a modern approach for image processing with considerable potential and promising results. As Deep Learning has been successfully applied to various application domains, it has also recently employed in Precision Agriculture. Taking this into account, this work proposes the use of machine learning techniques, more specifically Convolutional Neural Networks (CNN), to detect and count individuals in eucalyptus plantation images, acquired from Unmanned Aerial Vehicles (UAV). The obtained results were provided by a Faster R-CNN Resnet101, with validation procedure performed against manual human annotation. Experimental results demonstrated a overall precision of 95.77% and the affordability of the approach for forestry inventories.*

1. Introduction

With the development of new technologies, new opportunities are open to increase the work productivity and the intelligent resource management in several application domains. In the last few years, agriculture has been one of the areas that have obtained big advantages with the advance of the new technologies of monitoring and analysis, since the larger-scale observation is facilitated by the use of remote sensing and georeferencing [Bastiaanssen et al. 2000]. It can be done using satellite images, airplanes or Unmanned Aerial Vehicles (UAVs), i.e. drones, providing snapshots of the agricultural environment, being also a non-destructive method to collect information about the environment [Kamilaris and Prenafeta-Boldú 2018].

Images constitute a large part of the data collected through remote sensing. It usually provides a complete description of the agricultural environments, but also brings some challenges [Ozdogan et al. 2010]. The most popular techniques used for analyzing images includes the use of machine learning methods. There are many problems in agriculture that uses classical machine vision algorithms that could be benefited by the use of deep learning methods [Grinblat et al. 2016]. For some typical applications, there is a natural tendency to replace the classical techniques of machine vision for deep learning algorithms. These methods improved the state-of-art in object detection, object recognition and many other areas. It is bringing great advances in solving problems that the artificial intelligence community tried to solve for years [LeCun et al. 2015]. The use of deep learning in agriculture is recent. However, its growing popularity makes this as a promising technique since the advancements and applications of deep learning in other domains indicates a considerable potential [Kamilaris and Prenafeta-Boldú 2018].

Within this context, this work proposes the use of a CNN for detection and counting of individuals in an eucalyptus plantation. A *Faster R-CNN Resnet101* trained by 6000 steps using RGB images captured by UAVs was used to detect and count the individual plants. A total of 7833 eucalyptus plants in the region of interest can be found in a state that is detectable by a human being, also called here as Ground Truth (GT). The obtained results validate the use of CNNs for counting of those individuals, and also demonstrated the feasibility of the machine learning to provide analysis on forestry sector.

The remainder of this paper is organized as follows: Section 2 highlights the related works found over the literature. In Section 3 a detailed description of the proposed methodology is presented, describing the methods and techniques adopted to solve this particular problem. The obtained results, experimental environment and validation procedures are described in Section 4. Finally, conclusion, discussions and further works are discussed in Section 5.

2. Related Works

The higher resolution images captured by UAVs, and the powerful tools provided by deep learning open an opportunity to bring advances and to pave the way for precision agriculture. Over the literature we can find many researchers combining these tools to propose new solutions, and some of them are presented below.

In [Sa et al. 2016] the authors presented an approach to fruit detection using a Faster R-CNN adapted through transfer learning. Results show an improvement on accuracy, and in processing time (compared to prior works) to deploy for new fruits, as it requires bounding box annotation rather than pixel-level annotation. A simulated (trained on synthetic data and tested on real data) deep convolutional neural network for yield estimation was presented in [Rahnemoonfar and Sheppard 2017]. To capture features on multiple scales, it was used a modified version of the Inception-ResNet architecture. The algorithm counts efficiently even if fruits are under shadow, occluded by foliage, branches, or if there is some degree of overlap among fruits. The results show a better performance on synthetic images rather than on real images.

In [Bargoti and Underwood 2017a] was presented a fruit detection system using a Faster R-CNN in image data captured in orchards. Data augmentation techniques were found to improve performance with different number of training images. The study leads to the best yet detection performance (comparing with author's prior work). In [Bargoti and Underwood 2017b] was also proposed a framework for fruit detection and counting using orchard image data. A general purpose image segmentation approach was used, with Multilayer Perceptron and CNN. The pixel-wise fruit segmentation was done using the Watershed Segmentation and Circular Hough Transform. The results show an improvement in fruit segmentation performance, and the count estimates using CNN and Watershed Segmentation resulted in the best performance for the dataset used.

A fruit counting pipeline based on deep learning was described in [Chen et al. 2017]. A blob detector based on fully convolutional network extracts candidate regions in the images, and a counting algorithm based on a second convolutional network estimates the number of fruits in each region. Finally, a linear regression model maps the fruit count estimate to a final result. Experiments showed that the pipeline has a short training time and performs well. Another fruit counting pipeline was presented in [Liu et al. 2018]. First, a fully convolutional network was trained to segment video frame images into fruit and non-fruit pixels, and the evaluation of the algorithm was done by comparing the results with the ground truth annotated by humans. Results demonstrated that the pipeline was able to accurately and reliably count fruits across image sequences.

In [Xie et al. 2016] the authors proposed an algorithm for tobacco plants recognition and counting. The UAV captured images were processed using morphological reconstruction, and a Support Vector Machine was employed to classify the candidate regions as tobacco plants or not. Experimental results showed that the proposed method was adequate to the dataset. Also using UAV acquired images, in [Gnädinger and Schmidhalter 2017] images were analyzed to count maize plants. The analysis was made on the color of the leaves, using different image processing techniques. The error between the visually and digitally counted plants was small, demonstrating the capability of the proposed solution.

In [Reza et al. 2017] the goal was to automatically detect and count rice plants using images acquired with an UAV. It was applied morphological operations on binary images, and drawn boundaries to the connected components to count rice plants. The comparison between the numbers of rice plants detected and counted by the naked eye provided acceptable results. In [Ribera et al. 2017] the authors used an Inception v3 to count crop plants in field. Images were acquired using an UAV, and the number of plants was estimated using linear regression. The authors used a method to extract images of sections from an orthorectified image of the entire crop field, where these images were used for train and evaluate the CNN. Results showed a small error, validating the use of deep learning on the generated dataset.

3. Methodology

The main purpose of this work is to present a performance evaluation of a machine learning algorithm applied to the agriculture context. A well-defined methodology is important to present scientific robust results, as well as its validation procedure used to measure its overall performance. A general overview of the presented approach is summarized in the Figure 1, where the five steps are described and detailed in the next subsections.

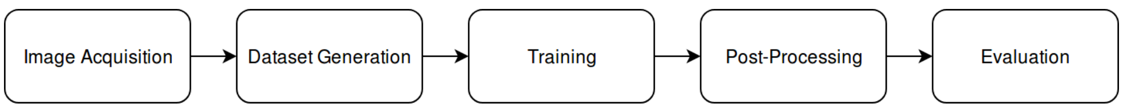


Figure 1. General overview of the proposed approach for detection and counting of eucalyptus plantation individuals.

3.1. Image Acquisition

The images used in this work were acquired using a fixed-wing mapping UAV, model Maptor, built by Hórus Aeronaves. This mapping method allows to acquire images with higher spatial resolution when compared to traditional remote sensing methods, like satellite imagery or even conventional aerophotogrammetry. The images were acquired with an embedded camera of 20MP from an altitude of 130m high, and were processed using the software *Pix4D*, aiming to obtain an orthomosaic from the flight region, as shown by Figure 2. The region of interest is placed on the center (marked as a reddish polygon).

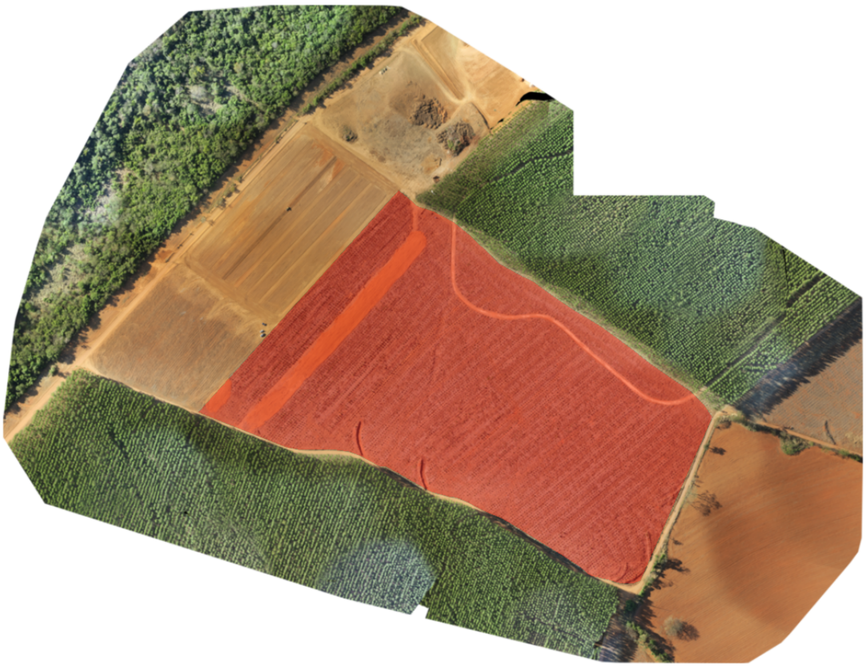


Figure 2. The original orthomosaic and region of interest highlighted in red, used in the experiments.

3.2. Dataset Generation

In machine learning applications, two sets of data are required: (i) a set for training and (ii) another one for testing with zero bias for performance evaluation. The training set need to be carefully selected in order to avoid convergence problem during the training step. Also, a good dataset must contain enough samples to be able to cover the most part of the cases that is necessary to classify [Kamilaris and Prenafeta-Boldú 2018].

The orthomosaic usually cover a large area, having tens of kilometers of pictorical information. Additionally, it is made necessary to process this information in blocks, since the CNNs demands a lot of computational resources, even in the presence of smaller images. In this work the region of interest was divided in fixed blocks of 512×512 pixels. To annotate the images, it was used the *LabelImg*[1], an open-source software tool for graphical image annotation. It was selected only the plant's canopy, avoiding to select other artifacts such as plant shadow or its neighborhood parts. A total of 218 images were generated, where 70% were randomly used for training (153 images, 5563 individuals) and the remainder 30% (65 images, 2270 individuals) for testing.

3.3. Training Step

The used CNN was developed in Python using the *TensorFlow*[2] framework. This tool was chosen by its popularity, good documentation and vast community support. In this work,

[1]https://github.com/tzutalin/labelImg
[2]https://tensorflow.org

it was used the *Faster R-CNN Resnet101*. The choice was based on the study showed by [Huang et al. 2017], where the authors presented the speed/accuracy/memory trade-offs in modern CNNs, using the *COCO Dataset*[3] for the evaluation. The selected meta-architecture stays approximately on the center of the graph when measuring accuracy and speed, providing a good balance.

In most cases, the CNN is not trained from scratch, with random initialization. It happens because is relatively rare to have a dataset large enough to do this. So, it is common to train a network with a large dataset, i.e. ImageNet, and use its weights to retrain the CNN for a similar problem [Pan et al. 2010]. This process is known as Transfer Learning. In our case, the transfer learning was done using the pre-trained models found in TensorFlow Model Zoo[4].

The most part of the parameters used in the experiments were default network values suggested for this architecture and configuration. The network was trained by 6000 steps, with the batch size set to 1. The only change made was the learning rate, whose value was decreased to 1/10 of its previous values (at every 1500 steps), allowing the network to get closer to its global optimum state.

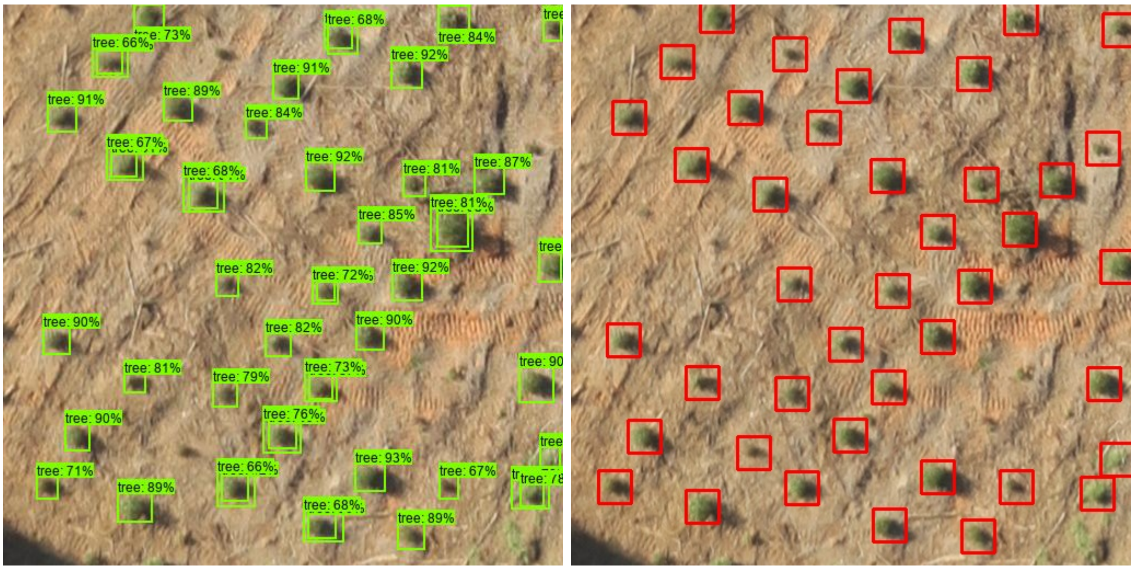
The *TensorFlow Object Detection API*[5] was used with this[6] specific commit. This Application Programming Interface (API) provides a fast way to train and test different meta-architectures of networks and feature extractors. Nonetheless, periodically the software stores a checkpoint file with information related to the training process until the present time. Since the output of the API is the value of loss only, the *TensorBoard 1.11.0* was used to check the progress of the training step. The training was finalized based on the value of the variable *TotalLoss*. In other words, when this value can be considered stabilized, the training process is stopped.

3.4. Post-Processing

After the training process is completed, a validation procedure to inspect the precision of the proposed model is necessary. An automated procedure was developed in Python to load the inference graph and process the test images used in the dataset. The obtained results presented (for some cases) the problem of double detection (the same individual detected by the network twice), as illustrated by Figure 3a.

In order to solve this problem, a simple post-processing step was taken to the output of the network (a file with the information about the detected points). This process is based on a simple Euclidean Distance among every other positive responses on the image. A minimal distance thresholding value was used (10 in this approach) to be considered a double detection, and the individual who has the higher probability is then kept, removing the others with smaller probabilities. The result of the post-processing step is shown in Figure 3b.

[3] https://cocodataset.org
[4] https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md
[5] https://github.com/tensorflow/models/tree/master/research/object_detection
[6] https://github.com/tensorflow/models/tree/42f98218d7b0ee54077d4e07658442bc7ae0e661



(a) An example of double detection problem

(b) Result of the application of the post-processing algorithm.

Figure 3. Post-Processing: classification of objects of interest is shown at the left side, and with post-processing at the right side, respectively.

3.5. Evaluation

An evaluation procedure was necessary to validate the network and its performance. To do so, the *Inference Graph* was used, i.e., a file containing the network weights and optimized model structure ready to be used. Usually, the measure provided by the API is the *Mean Average Process (mAP)*. It computes the average precision among all the processed images in the test dataset. It is particularly useful to compare given two distinct networks, but doesn't provide reliable data to measure the performance on individuals counting. So, the aforementioned metric is out of the scope of this paper and will not be used and discussed here.

To check whether the network detect the individuals properly, the ground truth file was loaded with the script previously described, combined with the output file provided by the network. A square of 30×30 was defined around the center point provided by ground truth files, and it was checked whether the point detected by the network belongs to this area. If positive, it is accounted as a true positive (TP). On the contrary, it is considered a false positive (FP). All the remaining individuals that were not detected by the model, is accounted as false negative (FN). The term "true negative" is not used here because when is the case, the network simply does not detect anything.

To evaluate the network as part of the goal of this approach, the metrics will not take into account the size of the resulting bounding box, since it is important to verify how many individuals can be detected by the model. Only the center points of the bounding box were used. Also, the detection probability was used only as an elimination criterion, i.e., the detected individuals that had its score below 50% will be excluded. To provide a fair result, it is necessary to take into account the false positives and false negatives. The false positives refer to an individual that is detected when there is no plant, while the false negatives measure when a tree exists, but is not detected by the network.

Two metrics were used to evaluate the network performance. The first is the Detection Rate. It measures the number of individuals that were correctly detected and it is defined by Equation 1. The sum $TP + FN$ represents the total of individuals.

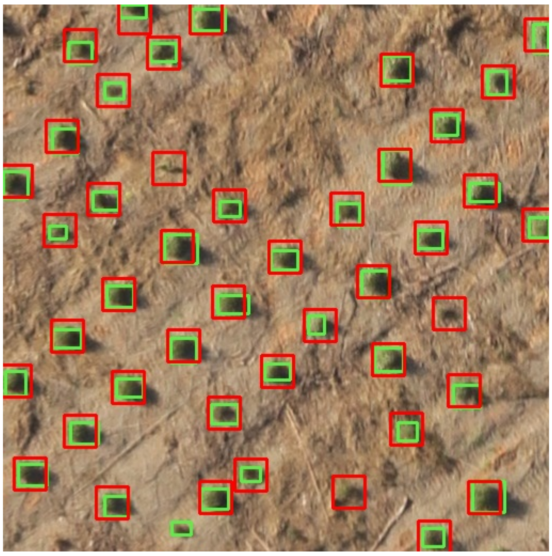
$$Detection\ Rate = \frac{TP}{TP + FN} \tag{1}$$

The Error Rate measures the percentage of individuals that were incorrectly detected by the CNN. In another words, measures the detections that aren't real individuals. It is defined by Equation 2.

$$Error\ Rate = \frac{FP + FN}{TP + FN} \tag{2}$$

Figure 4a shows the input image, and Figure 4b shows an example of evaluation. The green bounding boxes shows the actual trees, and the red ones shows the detection of the network.

(a) Input image



(b) Post-processed image (overlapped with the ground truth data)

Figure 4. An example of input and output images for this algorithm.

In this example, there were 42 trees, where 41 of them were correctly detected by the network (true positives), resulting in a detection rate of 97.62% (Equation 1). Also, there were a total of 44 detections, having 3 false positives and 1 false negative, resulting in an error rate of 9.1% (Equation 2).

4. Results

The whole system was implemented using the Python Programming Language. The processing unit is equipped with an Intel Core i3-4030U CPU @ 1.90GHz $\times$ 2, with 8GB of RAM, a SSD Sandisk PLUS with 240 GB and an Intel Corporation Haswell - ULT Graphics Controller. The operational system was Ubuntu Linux 16.04.5, 64-bit with

4.15.0-36-generic Linux Kernel. The TensorFlow 1.11.0 was installed in a virtual environment. Using the previously described methodology, the results were obtained after the network trained by 6000 steps, using the test dataset only. Figure 5 summarizes the obtained results.

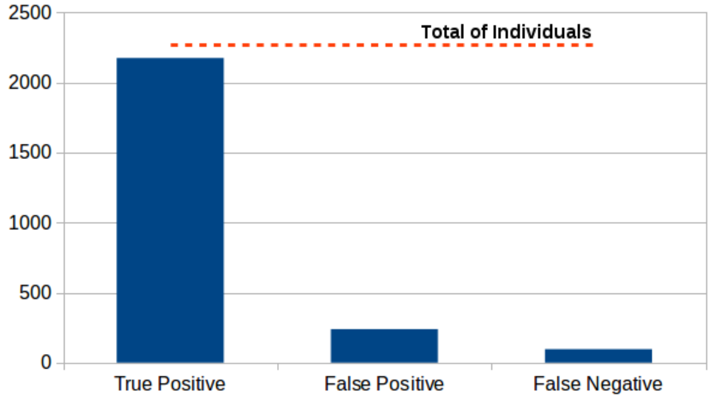


Figure 5. Overall precision obtained by the proposed approach.

Looking at the results, one can seen that the network detected 2174 of 2270 individuals, most part of them, achieving a detection rate of 95.77%. The false positive rate was relatively high, mostly if compared with the false negative rate, where the trained model detected 239 false individuals. Also, there were 96 false negatives, resulting in an error rate of 14.76%.

The use of RGB images, without the application of any additional filter, brought a good performance to the network. Also, one of the main advantages when using CNNs is the fact that it works as a general-purpose solver-problem since the features extraction is done automatically. In other words, by the use of backpropagation, the network adjust its weights generating complex filters to detect complex patterns [Krizhevsky et al. 2012]. Developing a hand-engineered feature extractor probably would be difficult, regarding the nature and complexity of the problem. According to [Kamilaris and Prenafeta-Boldú 2018], the automatic feature extraction performed by deep learning models is more effective when compared to the traditional approaches found over the literature, which fits well to the purpose of this problem.

On the other hand, it is hard to justify about the execution time required to train the model, since a personal computer was used in our experiments. To complete the training phase with 6000 steps, the framework required 1d 22h 30m, with a final *loss* of 0.3121. Deep learning techniques naturally takes a large amount of time to train the model when compared with traditional machine learning methods. However, testing, also by the nature of the CNNs, is a fast and parallelizable procedure, and can be performed mostly in real time [Kamilaris and Prenafeta-Boldú 2018]. Even taking almost two days (a long time when compared with classical machine learning approaches) to train this model, it is necessary to take into account two points: firstly, the experiments were performed in a conventional personal computer, without a Graphical Processing Unit (GPU), and the use of this device would decrease the training time; secondly, the time spent on manually design filters to extract features turns the time needed to labeling each image and training a CNN almost negligible [Sørensen et al. 2017].

5. Final Remarks

Deep Learning is a modern approach for image processing that arguably reached its maturity with potential and promising results. These techniques definitely have improved the state-of-the-art in object recognition and detection. Since deep learning has been successfully applied to various application domains, it has also recently employed in the domain of agriculture. In this context, this work proposed the use of Convolutional Neural Networks (Faster R-CNN Resnet101) to detect and count individuals on an eucalyptus plantation, on images acquired from UAVs.

The obtained results validates the possibility of use CNNs to detect and count individuals. The results shows the success of this approach, since it achieved a detection rate of 95.77%, close of the industry rate (about 97% of precision), which are not used commercially. The processing time is quite high (for training), since the experiment was performed on a personal computer, without any GPU support. However, the time spent during the training of this model is acceptable when compared with the current scenarios on deep learning. The time to export and load the model, and to process the images was small, mostly when we point that this application doesn't need to be performed in real time (in this case). Even with the good results achieved, it is hard to compare with other works, since each paper uses different pre-processing techniques, metrics, models, and parameters.

The dataset plays a crucial role on machine learning. So, it is believed that increasing the quality of the labeling process should help to obtain better results since on most part of applications exists a need for experts to annotate the input images. The use of appropriate pre-processing techniques could improve the results too. Extending the number of samples on the dataset, covering most cases, should improve the outcome from the network, because deep learning methods needs larger datasets.

Future works must include testing different models of meta-architectures and feature extractors. Also, it is desired to use CNNs on different datasets, changing the type of trees and evolutive stages of growth. Another experiment to be done would be to test this CNN with this same dataset, but applying some pre-processing techniques. Some comparisons on detection rate and processing time can be done among some classical techniques and deep learning methods.

Acknowledgments

We would like to thank Hórus Aeronaves for provide the images used on this work. Also, we thank Giovanni Cimolin da Silva for generating the dataset. Lastly, we thank the Santa Catarina Teaching and Engineering Foundation (FEESC) for funding this work.

References

Bargoti, S. and Underwood, J. (2017a). Deep fruit detection in orchards. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3626–3633.

Bargoti, S. and Underwood, J. P. (2017b). Image segmentation for fruit detection and yield estimation in apple orchards. *Journal of Field Robotics*, 34(6):1039–1060.

Bastiaanssen, W. G., Molden, D. J., and Makin, I. W. (2000). Remote sensing for irrigated agriculture: examples from research and possible applications. *Agricultural water management*, 46(2):137–155.

Chen, S. W., Shivakumar, S. S., Dcunha, S., Das, J., Okon, E., Qu, C., Taylor, C. J., and Kumar, V. (2017). Counting apples and oranges with deep learning: A data-driven approach. *IEEE Robotics and Automation Letters*, 2(2):781–788.

Gnädinger, F. and Schmidhalter, U. (2017). Digital counts of maize plants by unmanned aerial vehicles (uavs). *Remote Sensing*, 9(6):544.

Grinblat, G. L., Uzal, L. C., Larese, M. G., and Granitto, P. M. (2016). Deep learning for plant identification using vein morphological patterns. *Computers and Electronics in Agriculture*, 127:418–424.

Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S., et al. (2017). Speed/accuracy trade-offs for modern convolutional object detectors. In *IEEE CVPR*, volume 4.

Kamilaris, A. and Prenafeta-Boldú, F. X. (2018). Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture*, 147:70–90.

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

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436.

Liu, X., Chen, S. W., Aditya, S., Sivakumar, N., Dcunha, S., Qu, C., Taylor, C. J., Das, J., and Kumar, V. (2018). Robust fruit counting: Combining deep learning, tracking, and structure from motion. *arXiv preprint arXiv:1804.00307*.

Ozdogan, M., Yang, Y., Allez, G., and Cervantes, C. (2010). Remote sensing of irrigated agriculture: Opportunities and challenges. *Remote sensing*, 2(9):2274–2304.

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

Rahnemoonfar, M. and Sheppard, C. (2017). Deep count: fruit counting based on deep simulated learning. *Sensors*, 17(4):905.

Reza, M. N., Na, I. S., and Lee, K.-H. (2017). Automatic counting of rice plant numbers after transplanting using low altitude uav images. *International Journal of Contents*, 13(3):1–8.

Ribera, J., Chen, Y., Boomsma, C., and Delp, E. J. (2017). Counting plants using deep learning. In *Signal and Information Processing (GlobalSIP), 2017 IEEE Global Conference on*, pages 1344–1348.

Sa, I., Ge, Z., Dayoub, F., Upcroft, B., Perez, T., and McCool, C. (2016). Deepfruits: A fruit detection system using deep neural networks. *Sensors*, 16(8):1222.

Sørensen, R. A., Rasmussen, J., Nielsen, J., and Jørgensen, R. N. (2017). Thistle detection using convolutional neural networks. In *2017 EFITA WCCA CONGRESS*, page 161.

Xie, H., Fan, Z., Li, W., Rong, Y., Xiao, Y., and Zhao, L. (2016). Tobacco plant recognizing and counting based on svm. In *Industrial Informatics-Computing Technology, Intelligent Technology, Industrial Information Integration (ICIICII), 2016 International Conference on*, pages 109–113.