Unearthing Crop Insights: Plant Seedling Classification Using a Dense Convolutional Neural Network

Jacqueline Ennis
Stanford University
jqennis@stanford.edu

Madison Hall
Stanford University
mhall38@stanford.edu

Abstract

Leveraging computer vision for precision agriculture has the potential to transform commercial agriculture by offsetting both the cost and environmental impact of herbicides. Some attempts have been made to apply CNN's to classify cultivation crops from weeds, namely a robot capable of real-time segmentation at 91% accuracy, but none have leveraged the suite of powerful pre-trained networks trained on ImageNet and optimized for visual recognition tasks. We propose a convolutional neural network model to perform 12-class classification of close-up images of crop seedlings and weeds. After applying transfer learning to train on the pre-trained weights of DenseNet121, VGG16, and ResNet50, we find that the model still performs best, with an F1 score of 74.85. Finally, we discuss ways to improve and ideas for future work.

1 Introduction

Agricultural technology, or AgTech as it is known colloquially, faces unprecedented sustainability and productivity challenges over the coming decades. By 2050, the agriculture industry must produce more food than it has in totality over the last 8,000 years (Dutia, 2014). Doing so in the face of imminent environmental and climactic stresses will require a systematic overhaul of agricultural practices and technology as we know it. The digitization and subsequently increasing availability of agricultural data presents a unique opportunity for deep learning to play a role in these solutions.

Precision agriculture refers to the measurement and management of crop variability between and within fields. Leveraging remote-sensing and computer vision technology for micro-weed management - identifying and mapping weeds, crops, and soil on the scale of individual plants - promises to reduce the expense and environmental impact of herbicides, increase crop yields, and mitigate labor time and costs (Lamb & Brown, 2001). Precision agriculture techniques are not readily available at commercial scale due to their expensive nature.

Fortunately, state of the art computer vision models are posed to perform particularly well on such a visual recognition task. We propose an architecture for a convolutional neural network that can take as input an image of a crop seedling and predict the crop's species among 12 species classes. This work is important because identifying weeds from crops at the early growth stages is critical for identification and circumvents the difficulties of feature extraction from overlapping leaves.

2 Related work

Classification of cultivation plants has long been done manually, using knowledge of plant phonology and cross-referencing with databases of clippings, photos, and illustrations. As image recognition technology has improved over the last decade, more attempts to train computers to the task have cropped up. Applying deep learning represents the most recent and most successful of these attempts to date.

Traditional statistical approaches to plant classification achieve sufficient, but limited, results. Ok, Akar & Gungor use a random forest classifier to classify crop species from satellite imagery data, achieving an accuracy of 85.89% (2012). However, this method relies on identifying parcels of agricultural fields rather than individual plants. One promising work without deep learning is LeafSnap, a project and mobile application that claims to identify 184 species of trees given images of individual leaves on a solid color background (Kumar et al., 2012). The model relies on this non-textured background condition to employ a Support Vector Machine (SVM) and Expectation-Maximization to perform color segmentation and identify the contours of the leaf. Then, the model uses the K-Nearest Neighbors algorithm to identify the closest matching tree species from the database. LeafSnap reports that 96.8% of queries are matched to the correct species within the top 5 results shown to the user, but does not report accuracy for rank-1 matches, which would be necessary for precision agriculture use cases.

Without deep learning, these methods depend on hand-crafted features, like leaf shape and color segmentation. Deep learning approaches prove far more robust to diverse input sets. The DeepPlant project found that a CNN could predict the species of 44 plants with up to 99.5% given just close-up images of leaf veins when trained on a varied image set of both full and cropped leaves. The most state-of-the-art work with particular relevance to our project is a precision-agriculture robot in Germany, that uses a deep encoder-decoder CNN for semantic segmentation of crop fields in real-time (Milioto, Lottes & Stachniss, 2018). To separate weeds from sugar beet crops and background, it is fed only with RGB image data and 14-channel image-storing vegetation indices. In doing so, they make assumptions about the input data in order to improve resilience against varying field conditions. They achieve 91.88% on their own test data. Nkemelu, Omeiza & Lubalo attempt to improve on this very crop seedling classification problem by using a 6-layer CNN with max-pooling and dropout (2018). When using OpenCV's background segmentation preprocessing, they achieve an accuracy of 92.60% on their validation set.

3 Dataset and Features

Our dataset contains 5,539 labelled, colored PNG images of 960 individual plants at various growth stages, constituting 12 different species. Each image has an original resolution of 10 pixels per mm. This dataset was recorded at Aarhus University Flakkebjerg Research station and provided by a collaboration between University of Southern Denmark and Aarhus University (Giselsson et al., 2017). In 2018, a Kaggle prediction competition popularized this dataset for classification models.

The species and class sizes in this dataset are as follows:

Species	Status	Number of examples
Maize	Crop	257
Common wheat	Crop	253
Sugar beet	Crop	463
Scentless Mayweed	Weed	607
Common Chickweed	Weed	713
Shepherd's Purse	Weed	274
Cleavers	Weed	335
Charlock	Weed	452
Fat Hen	Weed	538
Small-flowered Cranesbill	Weed	576
Black-grass	Weed	309
Loose Silky-bent	Weed	762

Figure 1. List of classes and their sizes

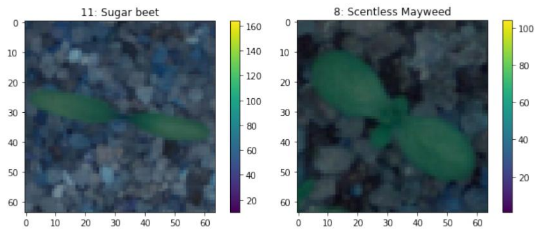


Figure 2. Examples images of crop (Sugar beet, left) and weed (Scentless mayweed, right.

Following standard best practices, we use random shuffling to split the data into 4433 examples (80%) for the training set and 1108 examples (20%) for the test set. We then hold out 443 examples (10%) as the validation set during training.

3.1 Data preprocessing

First, we resize the images to resolutions of 64 × 64 in order to standardize the input to our model before converting the images into Numpy arrays (Van Der Wal, Colbert, & Varoquaux, 2011). Then, we normalize the input arrays per RGB channel in order to speed up training and reduce amplitude variance. We attempt two ways to normalize the images: feature scaling to range [0,1], and standard-scoring using the mean and standard deviation. We found that the former method performs better when using our softmax activation function and followed it for the remaining training and testing. We use OpenCV and Scikit-Learn packages during these preprocessing steps (Bradski & Kaehler, 2000; Pedregosa et al., 2011). As one can see in Figure 1, the classes of this dataset are very unbalanced, with a nearly 500-image gap between the smallest minority class and the largest majority class. To remedy this, we perform data augmentation on the in order to a) enhance the size and variety of the training data and b) over-sample the minority classes to balance out the clas sizes. These transformations include random rotations, reflections, and adding noise. This step increases our training set to 13120 with roughly 1,000 images per class, as recommended for image classification tasks. Finally, we shuffle the data to prevent batches from having the all same labels during training.

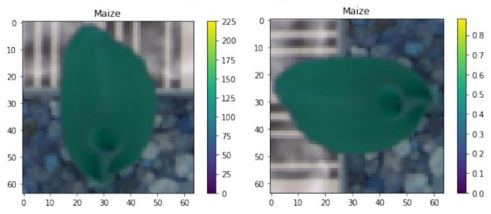


Figure 4. Image of maize seedling before and after rotation, which helps the model generalize to the soil walls

4 Methods

4.1 Baseline model

To measure baseline performance on the data, we train a simple network composed of two 2D convolutional layers (of sizes 32 × 32 and 64 × 64, respectively) accompanied by a 2D max pooling to reduce dimensionality, and 10% drop out, to prevent overfitting. The last two layers are fully-connected. All convolutional layers use the ReLU activation function, except for the final dense layer, which uses softmax activation to output interpretable prediction probabilities. The model is compiled using the ADAM Optimization algorithm, a popular choice for deep learning models for its computational efficiency and relative ease of implementation (Kingma & Ba, 2014). The class labels are converted to a one-hot encoding representation for the categorical cross-entropy loss function (Figure 4), which is commonly used for multiclass classification problems.

$$J = -\frac{1}{m}\sum_{i=0}^{m}\left(y^{(i)}\log\left(a^{[2](i)}\right) + (1-y^{(i)})\log\left(1-a^{[2](i)}\right)\right)$$

Figure 5. Cross-entropy loss function (CS230)

4.2 Pre-trained weights

Since the original crop seedling dataset is relatively small (<1000 per species class), we decide to leverage Keras architectures with weights pre-trained on ImageNet, a database of about 50 million word-labeled, full-resolution images which has since become a mainstay resource for computer vision research (Deng et al., 2009). Using pre-trained weights has been shown to improve performance for datasets as small as 100 images per class to levels expected of much larger datasets (Cho et al., 2015). We adapt three such architectures – DenseNet121, VGG16, and ResNet50 – and compare their performance.

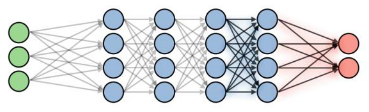


Figure 5. Visualization of freezing all but last layers of a pre-trained network (Amidi & Amidi, 2018)

We use transfer learning to apply each of these architectures to our data by training the weights on softmax as well as, experiment-permitting, the last few layers. Otherwise, we freeze all remaining top layers. We vary the number of frozen layers as the independent variable in one of our experiments. Using the pre-trained network as the base, we then add two dense layers to customize our output shape and a 10% dropout to combat overfitting.

DenseNet121 DenseNet121 is a 121-layer deep CNN, the smallest offering of a suite of models known as Densely Connected Convolutional Networks (Huang et al., 2017). DenseNets' layers are connected feed-forward, with feature-maps of each preceding layer used as input for each subsequent layer. We chose DenseNet for its many advantages: addressing the vanishing-gradient problem, encouraging feature reuse, reducing the number of parameters needed, all at relatively lower computation costs for high performance.

VGG16 VGG16 is a 16-layer deep convolutional network designed for image detection (Simonyan, K., & Zisserman, 2014) that uses small, 3×3 convolutional layers and 2 max-pooling layers. VGG's depth allows it to extract more complex and representative features from images.

ResNet50 ResNet is a 50-layer Deep Residual Learning designed for image recognition (He et al, 2016). It is well-known for its robustness against the accuracy degradation problem when deeper models converge.

5 Results and discussion

5.1 Evaluation metrics

As our main evaluation metric, we focus on F1 score:

$$F1 = \frac{2 \times precision \times recall}{precision + recall}$$

This is a more nuanced and expressive metric than accuracy for two reasons. First, our dataset is relatively unbalanced (as little as 253 examples for the smallest class and 762 examples for the largest class – a 500 example gap). The model could achieve a high accuracy by over-predicting the majority class, Loose Silky-bent, even if it were under-predicting minority classes (in fact, this was a common problem in the early stages of this project). Second, the weed-detection problem is particularly sensitive to false negatives and false positives; failing to classify weeds as weeds or, even worse, incorrectly classifying cultivation crops as weeds, would prove costly in the use case of automated herbicide spraying. Therefore, we require an evaluation metric like F1 that captures both precision and recall.

5.2 Experiments

Freezing pre-trained network layers When using transfer learning to take advantage of pre-trained weights, one can freeze the weights of certain layers to prevent them from changing during training. In Amidi & Amidi, its recommended to freeze all layers when training on a small dataset and unfreezing all layers on a sufficiently large dataset. To find the optimal adjustment for our crop seedling data, we gradually unfreeze the bottom layers of the network - we chose DenseNet121 for this experiment, since it was the fastest to train - as the independent variable and record the effect on both the test accuracy and test F1 score of the model.

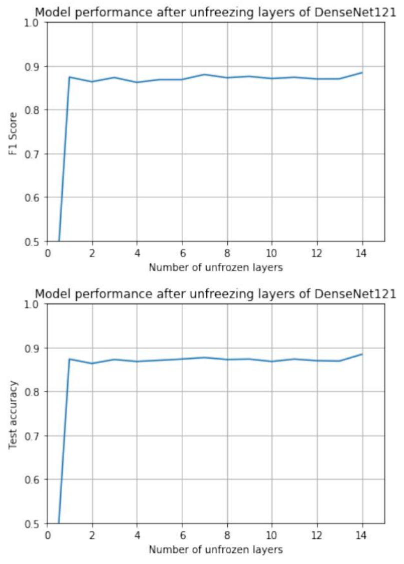


Figure 6. Results of unfreezing up to 14 layers of DenseNet121

After training the model for 10 epochs at a time using up to 14 unfrozen layers, we found that the benefits of unfreezing layers tapers off after the first layer. The highest 88.40% F1 score is achieved at 7 and 14 unfrozen layers, but these only represents a 1% improvement over the F1 score of 87.40% achieved after 1 unfrozen layer.

Increasing dropout rate In an effort to reduce overfitting, we tried increasing the dropout rate at 0.1 intervals and recorded the effect on the F1 score. Despite our hypothesis, this resulted in no significant change in the F1 score.

Data augmentation and shuffling We were able to measure our model's performance on both the original dataset (normalization, no data augmentation) and the augmented dataset (includes transformations of images). Since the class sizes were so imbalanced in the original dataset, there was a clear preference for the majority class (*Loose silky-bent*, with the original model almost exclusively predicting the same one class. After performing data augmentation and retraining the models, their performances improved, as discussed in the next section.

5.3 Results

After training each model on the normalized training data for 20 epochs, we can compare their performances.

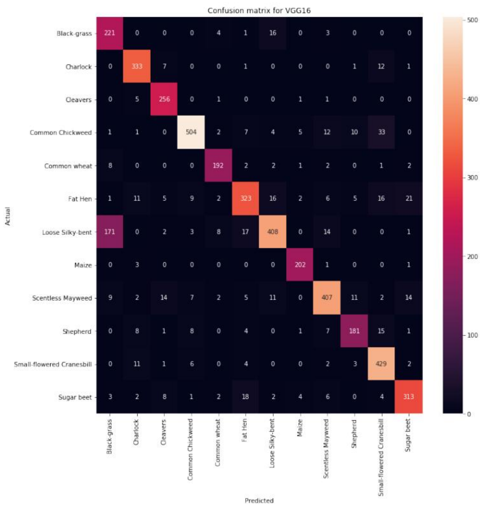
Hyperparameters. For each instance, we hold out 10% of examples as the validation set and use a learning rate of 1e-3 and a batch size of 32, the default parameters, balancing convergence speed and stability. We use a dropout rate of 0.4 to prevent overfitting to the training data, and freeze all but the bottom layer for the pre-trained networks.

Model	F1 score	Test accuracy (%)	Train accuracy (%)	Train time (s)
Baseline	74.85	73.62	70.05	167.81
DenseNet121	56.94	56.72	62.46	233.13
VGG16	59.21	59.44	63.24	222.11
ResNet50	56.48	56.91	90.82	252.41
Baseline*	73.04	73.62	99.83	587.78
DenseNet121*	62.32	64.02	77.50	360.95
VGG16*	65.67	65.14	66.49	328.00
ResNet50*	70.67	71.23	75.84	358.84

*Figure 6. Comparing the performances of each model. * denotes that the model was trained on the augmented dataset.*

5.4 Discussion

We can see in the results above that the baseline model performed the best both before and after data augmentation. All models except for ResNet50 saw an improvement in performance after training on the augmented data. The relatively low F1 scores for the pre-trained networks are inconsistent with our hypothesis that the pre-trained networks would perform far better than a model trained from scratch. To investigate why, we examine a confusion matrix and misclassified examples to understand how the models are making their predictions.



Actual \ Predicted	Black-grass	Charlock	Cleavers	Common Chickweed	Common wheat	Fat Hen	Loose Silky-bent	Maize	Scentless Mayweed	Shepherd	Small-flowered Cranesbill	Sugar beet
Black-grass	221	0	0	0	4	1	16	0	3	0	0	0
Charlock	0	333	7	0	0	1	0	0	0	1	12	1
Cleavers	0	5	256	0	1	0	0	1	1	0	0	0
Common Chickweed	1	1	0	504	2	7	4	5	12	10	33	0
Common wheat	8	0	0	0	192	2	2	1	2	0	1	2
Fat Hen	1	11	5	9	2	323	16	2	6	5	16	21
Loose Silky-bent	171	0	2	3	8	17	408	0	14	0	0	1
Maize	0	3	0	0	0	0	0	202	1	0	0	1
Scentless Mayweed	9	2	14	7	2	5	11	0	407	11	2	14
Shepherd	0	8	1	8	0	4	0	1	7	181	15	1
Small-flowered Cranesbill	0	11	1	6	0	4	0	0	2	3	429	2
Sugar beet	3	2	8	1	2	18	2	4	6	0	4	313

Figure 8. Confusion matrix for VGG16 model

Here we can see that there is a high number of *Loose silky-bent* images that the model is predicting as *Black grass*.

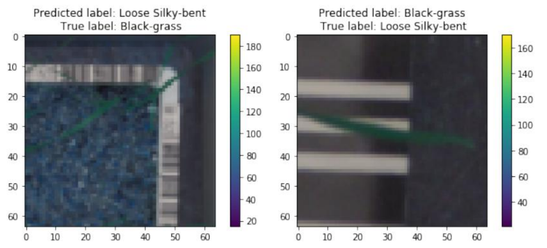

Predicted label: Loose Silky-bent
True label: Black-grass
Predicted label: Black-grass
True label: Loose Silky-bent

Figure 9. Misclassified examples of similar looking seedlings

Clearly these two seedlings are very similar in terms of size, shape, and color, making them difficult to distinguish. Despite our data augmentation efforts, we posit that the model would need more training data (especially since *Black grass* is one of the most underrepresented classes) to be able to recognize them from each other.

6 Conclusion and future Work

Overall, the simple CNN model performed the best in all categories. This was surprising considering our hypothesis that the pre-trained networks would outperform a model trained from scratch when using weights from training on ImageNet. Possible reasons for which this was not the case could be that our dataset is too dissimilar from ImageNet; we normalized our data differently from the pre-trained networks; we modified the structure of the pre-trained networks too much; we did not customize the structure of pre-trained networks enough. In the future, we would like to try adding texture analysis as a feature to improve the results.

7 Contributions

J.E. implemented DenseNet and ResNet models. J.E. explored data augmentation, freezing layers, and increasing dropout rate. J.E. also preprocessed the data.
M.H. implemented baseline and VGG model. M.H. created confusion matrices and compiled results. M.H. conducted literature review.

8 Github Link

`https://github.com/jackieennis/cs230-seedling-classifier/blob/master/jackie%20(2).ipynb`

References

Amidi, A., & Amidi, S. (2018, November). *Deep Learning Tips and Tricks Cheatsheet.* Retrieved from https://stanford.edu/ shervine/teaching/cs-230/cheatsheet-deep-learning-tips-and-tricks

Bradski, G., & Kaehler, A. (2000). OpenCV. Dr. Dobb's journal of software tools, 3.

Cho, J., Lee, K., Shin, E., Choy, G., & Do, S. (2015). How much data is needed to train a medical image deep learning system to achieve necessary high accuracy?. arXiv preprint arXiv:1511.06348.

Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009, June). Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition (pp. 248-255). Ieee.

Dutia, S. G. (2014). Agtech: Challenges and opportunities for sustainable growth. Innovations: Technology, Governance, Globalization, 9(1-2), 161-193.

Giselsson, T. M., Jørgensen, R. N., Jensen, P. K., Dyrmann, M., & Midtiby, H. S. (2017). A public image database for benchmark of plant seedling classification algorithms. arXiv preprint arXiv:1711.05458.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4700-4708).

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

Kumar, N., Belhumeur, P. N., Biswas, A., Jacobs, D. W., Kress, W. J., Lopez, I. C., & Soares, J. V. (2012, October). Leafsnap: A computer vision system for automatic plant species identification. In European Conference on Computer Vision (pp. 502-516). Springer, Berlin, Heidelberg.

Lamb, D. W., & Brown, R. B. (2001). Pa—precision agriculture: Remote-sensing and mapping of weeds in crops. Journal of Agricultural Engineering Research, 78(2), 117-125.

Lemaître, G., Nogueira, F., & Aridas, C. K. (2017). Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. The Journal of Machine Learning Research, 18(1), 559-563.

Milioto, A., Lottes, P., & Stachniss, C. (2018, May). Real-time semantic segmentation of crop and weed for precision agriculture robots leveraging background knowledge in cnns. In 2018 IEEE International Conference on Robotics and Automation (ICRA) (pp. 2229-2235). IEEE.

Nkemelu, D. K., Omeiza, D., & Lubalo, N. (2018). Deep Convolutional Neural Network for Plant Seedlings Classification. arXiv preprint arXiv:1811.08404.

Ok, A. O., Akar, O., & Gungor, O. (2012). Evaluation of random forest method for agricultural crop classification. European Journal of Remote Sensing, 45(1), 421-432.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. Journal of machine learning research, 12(Oct), 2825-2830.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

Van Der Walt, S., Colbert, S. C., & Varoquaux, G. (2011). The NumPy array: a structure for efficient numerical computation. Computing in Science & Engineering, 13(2), 22.