A Study of the Classification of Root Types Using Several Convolutional Neural Networks

James Spiker

May 13, 2023

Abstract

A recent proof of concept study by Erdem Erdemir of Tennessee State University examined the feasibility of using Local Binary Patterns to classify roots of plants grown in hydroponic systems as either hairy or non-hairy, as hairy roots are more likely to mature into adulthood. This study builds on the prior study by examining the performance of several convolutional neural network architectures on the same data set.

1 Introduction

Hydroponic agriculture, or growing plants without soil has been known and practiced for thousands of years. Technology advancements have renewed interest in this technique as it reduces the overall footprint needed for agriculture purposes. Institutions like NASA are especially interested in hydroponic techniques for this very reason. In order to maximize yields, ensuring the most viable plants are cultivated is essential. To that end, analyzing root growth can help cultivators focus their energy and efforts on plants that have a much higher probability for survival. Erdem Erdemir of Tennessee State University and Timothy Darrah of Vanderbilt University used Local Binary Patterns to classify roots as either "hairy," or "non-hairy," based on a data set of switch-grass root images.[ED18][Ros21] This data set consisted of 1000 images collected from 16 plants, with each image coming from different roots. In this study we will examine this same classification problem and data set by using convolutional neural networks of various depths and architectures to see if there is any improvement in the results.

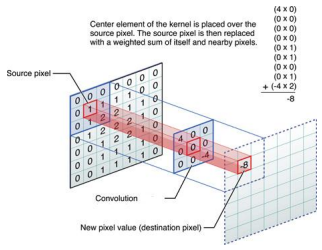

Figure 1: How a kernel convolves an input layer

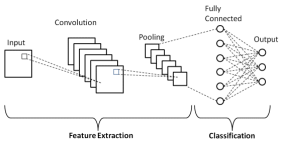

Figure 2: Basic Structure of a CNN

2 Convolutional Neural Networks: An Overview

Neural Networks transform an input vector through a series of hidden layers and activation functions. Each layer is made up of neurons, and each neuron receives an input from each neuron in the previous level, or each dimension in the vector. The last layer in the network represents the output layer and is the final classification of the standard neural network.

A convolutional neural network employs a number of trainable filters, or kernels to encode parameters that will in turn produce the correct classification. These filters allow the neural network to detect more signals in the input vectors and in turn, improve accuracy. The CNNs employed in this paper use the layers described in this section.

2.1 Convolution Layer

The central part of a Convolutional Neural Network, is of course, the Convolutional Layer. Convolutional Layers consist of the application of a number of filters, or kernels of size nxn, where n is a relatively small number and the kernel is applied to all channels of the input vector. In other words, think of a kernel as a small nxn matrix that slides over the input vector doing element-wise multiplication, summing the results and recording the result in a new 2 dimensional matrix for each channel. We combine each of these matrices into a 3 dimensional activation map. The volume of this activation map is determined by the depth, or channels in the input region, the stride, or how far the kernel slides across the input vector in each step, and zero-padding, or how the edges of the input matrix are modified to produce a mathematically consistent result.

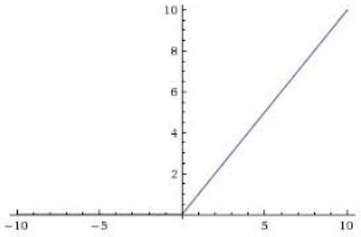

Figure 3: Line plot of the ReLU function

2.2 Activation Layers

In the Activation Layer, an activation function, often ReLU, is applied to the input volume to "activate" neurons. Often times, this is not included as a separate layer as it is assumed that immediately after a convolution layer, there will be an activation layer. This layer introduces non-linearity into the neural network. When the value of each neuron exceeds a certain value, the activation function will pass the neuron along as "activated," thereby increasing the importance of that particular neuron to the result.

2.3 Pooling Layers

Pooling layers allow the neural network to reduce the size of the input layers in order to reduce the number of computations the network requires and to help reduce over-fitting to the training data set. There are two types of pooling, Max pooling, which is often used in the internal part of the network, and average pooling, which is mostly used just prior to the output layer of the network. The pooling layer is developed by, once again, sliding a nxn (where n is small) window over the input vector, like in the convolutional layer. However, rather than performing element-wise multiplication and summation, for max pooling, the result is simply the max value in the window. For average pooling, the output is the average of all values in the window.

2.4 Fully Connected Layers

Fully Connected Layers are layers where all neurons are connected to the activated neurons of the previous input layer. This is a typical neural network layer and reside directly prior to the output layer, where a softmax classifier will be applied to achieve the required, (for this data set,) binary result.

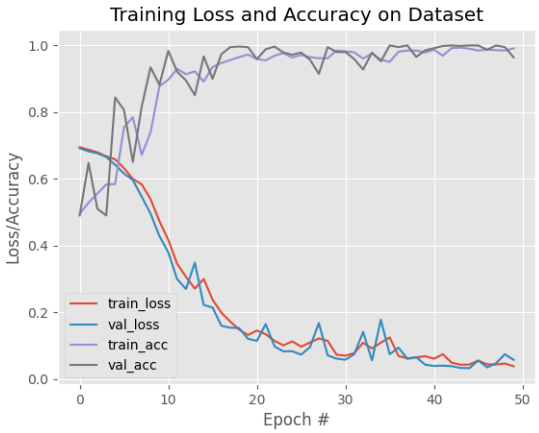

Figure 4: Plot of LeNet performance on the dataset.

2.5 Batch Normalization Layers

Batch Normalization Layers are used to normalize the activation outputs of a given layer before passing those as inputs into the next layer. The mean and standard deviations of the activation values for each batch are calculated during the training process. During testing, these values are replaced with a running average of these values. This normalization is very effective at reducing the number of epochs required to train the network, and help make the network easier to tune. [cite Ioffe and Szegedy,]

2.6 Dropout Layers

Dropout Layers are included to help reduce over fitting to a data set. This layer randomly deactivates, or "drops out," neurons to help distribute the responsibility for activation given a particular feature across several neurons.

3 The Networks and Their Results

3.1 LeNet

LeNet is a classic CNN architecture first proposed by Yann Lecun in 1998.[LBBH98] This architecture was first described to decipher hand written characters. Its structure is one of three sets of (Convolution, Activation, Max Pooling) layers fed into 2 fully connected layers. This architecture is used in software that is run on million of devices used to read handwriting. LeNet's execution time on this data set was: 458 seconds. LeNet achieved an F1 score of .96

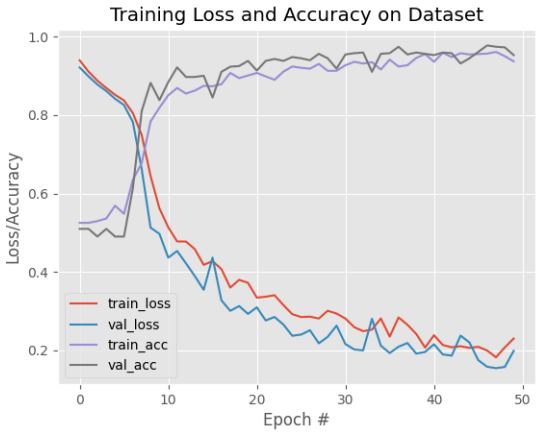

Figure 5: Plot of SimpleNet performance on the dataset.

3.2 SimpleNet

SimpleNet is a CNN architecture that was designed to be a good trade-off between computational efficiency and accuracy. It consists of three series of (Convolution, Activation, Max Pooling, Dropout) layers fed into two fully connected layers with the last fully connected layer using a softmax function for categorical variables.[HRFS23] When trained on this data set the SimpleNet architecture had good performance. The time it took for execution was 1722 seconds. The SimpleNet architecture achieved an F1 score of .95.

3.3 AlexNet

In 2012 Alex Krizhevsky won the Imagenet competition, training on millions of images with 1000 classes. This network consists of a two series of (Convolution, Activation, Normalization, Max Pooling) layers, followed by two series of (Convolution, Activation, Normalization) layers, followed by one more (Convolution, Activation, Normalization, Max Pooling), then fed into three fully connected layers.[KSH12] When trained on this data set, it achieved an F1 score of .99 in a time of 1893 seconds.

3.4 VGGNet

VVGNets are a family of CNNs that only use 3x3 filters and follows the pattern of (Convolution, Activation, Normailization, Convolution, Activation, Normailization) layers followed by a Max Pooling layer, then a Dropout layer. Any number of this pattern may be used before feeding the outputs into two fully connected layers separated by a dropout layer.[SZ15] The multiple convolution layers before any pool layers should allow the network to extract and account for more features than shallower networks and therefore should perform better on the validation set. For this study, 2 stacks of the VGG pattern were used. The time it took for execution was 1645 seconds. The VGGNet architecture

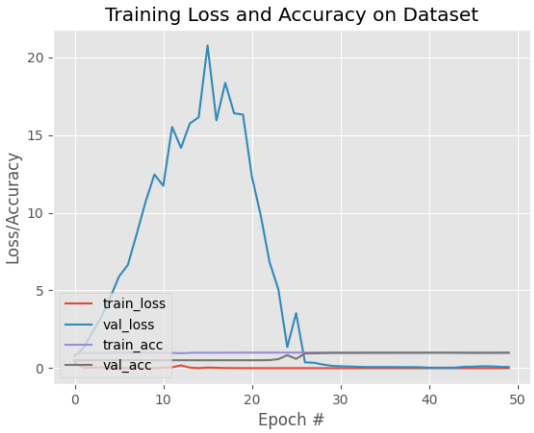

Figure 6: Plot of AlexNet performance on the dataset.

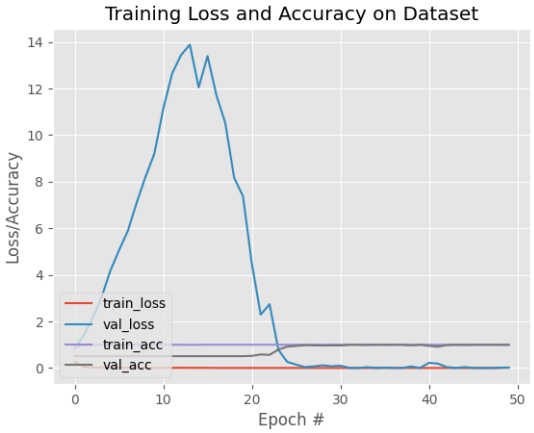

Figure 7: Plot of VGGNet performance on the dataset.

achieved an F1 score of 1.0

4 Conclusions

All convolutional neural network architectures performed well, while some were much quicker than the others. The depths of the AlexNet and VGGNet algorithms allows for more generalization and therefore higher validation scores. The simplicity of LeNet allows for a much quicker training time. Ultimately it is left to the user to decide to prioritize speed of training over accuracy, or vice versa.

References

[ED18] Erdem Erdemir and Tim Darrah. Real-time root monitoring of hydroponic crop plants: Proof of concept for a new image analysis system. pages 313–323, 12 2018.

[HRFS23] Seyyed Hossein Hasanpour, Mohammad Rouhani, Mohsen Fayyaz, and Mohammad Sabokrou. Lets keep it simple, using simple architectures to outperform deeper and more complex architectures. 2023.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[LBBH98] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[Ros21] Adrian Rosebrock. Deep learning, hydroponics, and medical marijuana, Apr 2021.

[SZ15] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.