Analyzing crop images for improving agriculture

Divyang Doshi
(Student)

Dr. Anukriti Bansal
(Mentor)

CERTIFICATE

Acknowledgement

It is my privilege to express my sincerest regards to my project mentor, Dr. Anukriti Bansal , for her valuable inputs, able guidance, encouragement, whole-hearted cooperation and constructive criticism throughout the duration of my project. I deeply express my sincere thanks to him for encouraging and allowing me to present the project on the topic “ANALYZING CROP IMAGES FOR IMPROVING AGRICULTURE “ for the Summer Internship. I take this opportunity to thank all the staff members of the college who have directly or indirectly helped in my project. I pay my respect and love to my parents and all other family members and friends for their love and encouragement throughout my career. Last but not the least I express my thanks to my friends for their cooperation and support.

ABSTRACT

Crop diseases are a major threat to food security, but their rapid identification remains difficult in many parts of the world due to the lack of the necessary infrastructure. The combination of increasing global smartphone penetration and recent advances in computer vision made possible by deep learning has paved the way for smartphone-assisted disease diagnosis. Using a public dataset of 54,306 images of diseased and healthy plant leaves collected under controlled conditions, we train a deep convolutional neural network to identify 38 classes with name of crop and its disease (or absence thereof). The trained model achieves an accuracy of 90.23% on a held-out test set, demonstrating the feasibility of this approach. We also trained a network to identify 21 classes that included the name of disease in the given crop. This model achieved an accuracy of 92.27%. At the end we trained a network to distinguish between healthy and diseased crop and got a test accuracy of 98.89%. Overall, the approach of training deep learning models on increasingly large and publicly available image datasets presents a clear path towards smartphone-assisted crop disease diagnosis on a massive global scale.

Table of Contents

1. Introduction

Modern technologies have given human society the ability to produce enough food to meet the demand of more than 7 billion people. However, food security remains threatened by a number of factors including climate change, the decline in pollinators, plant diseases, and others. Plant diseases are not only a threat to food security at the global scale, but can also have disastrous consequences for smallholder farmers whose livelihoods depend on healthy crops.

1.1 Motivation

Various efforts have been developed to prevent crop loss due to diseases. Historical approaches of widespread application of pesticides have in the past decade increasingly been supplemented by integrated pest management (IPM) approaches. Independent of the approach, identifying a disease correctly when it first appears is a crucial step for efficient disease management. In more recent times, such efforts have additionally been supported by providing information for disease diagnosis online, leveraging the increasing Internet penetration worldwide. Even more recently, tools based on mobile phones have proliferated, taking advantage of the historically unparalleled rapid uptake of mobile phone technology in all parts of the world.
The combined factors of widespread smartphone penetration, HD cameras, and high performance processors in mobile devices lead to a situation where disease diagnosis based on automated image recognition, if technically feasible, can be made available at an unprecedented scale. Here, we demonstrate the technical feasibility using a deep learning approach utilizing 54,306 images of 14 crop species with 21 diseases (or healthy) made openly available through the project PlantVillage

1.2 Traditional Approaches

***S*IFT (Scale-Invariant Feature Transform)**

The SIFT algorithm uses a series of mathematical approximations to learn a representation of the image that is scale-invariant. In effect, it tries to standardize all images. This corresponds to the idea that if some feature (say a corner) can be detected in an image using some square-window of dimension σ across the pixels, then we would if the image was scaled to be larger, we would need a larger dimension kσ to capture the same corner. The general idea is that SIFT standardizes the scale of the image then detects important key features. The existence of these features are subsequently encoded into a vector used to represent the image.

SURF (Speeded-Up Robust Features)

SURF is simply a speeded-up version of SIFT. SURF works by finding a quick and dirty approxima- tion to the difference of Gaussians using a technique called box blur. A box blur is the average value of all the images values in a given rectangle and it can be computed efficiently

BRIEF (Binary Robust Independent Elementary Features)

RIEF takes a shortcut by avoiding the computation of the feature descriptors that both SIFT and SURF rely on. Instead, it uses a procedure to select n patches of pixels and computes a pixel intensity metric. These n patches and their corresponding intensities are used to represent the image. In practice this is faster than computing a feature descriptor and trying to find features.

1.3 Why deep learning based approach?

The most important difference between deep learning and traditional machine learning is its performance as the scale of data increases. When the data is small, deep learning algorithms don't perform that well. This is because deep learning algorithms need a large amount of data to understand it perfectly. On the other hand, traditional machine learning algorithms with their handcrafted rules prevail in this scenario.
In Machine learning, most of the applied features need to be identified by an expert and then hand-coded as per the domain and data type whereas deep learning algorithms try to learn high-level features from data. This is a very distinctive part of Deep Learning and a major step ahead of traditional Machine Learning. Therefore, deep learning reduces the task of developing new feature extractor for every problem. Like, Convolutional NN will try to learn low-level features such as edges and lines in early layers then parts of faces of people and then high-level representation of a face.

1.4 Challenges

In order to develop accurate image classifiers for the purposes of plant disease diagnosis, we needed a large, verified dataset of images of diseased and healthy plants. The PlantVillage project has begun collecting tens of thousands of images of healthy and diseased crop plants , and has made them openly and freely available. Here, we report on the classification of 26 diseases in 14 crop species using 54,306 images with a convolutional neural network approach.
We trained a deep convolutional neural network to identify 38 classes with name of crop and its disease (or absence thereof). The trained model achieves an accuracy of 90.23% on a held-out test set, demonstrating the feasibility of this approach. We also trained a network to identify 21 classes that included the name of disease in the given crop. This model achieved an accuracy of 92.27%. At the end we trained a network to distinguish between healthy and diseased crop and got a test accuracy of 98.89%.

2. CNN for crop disease

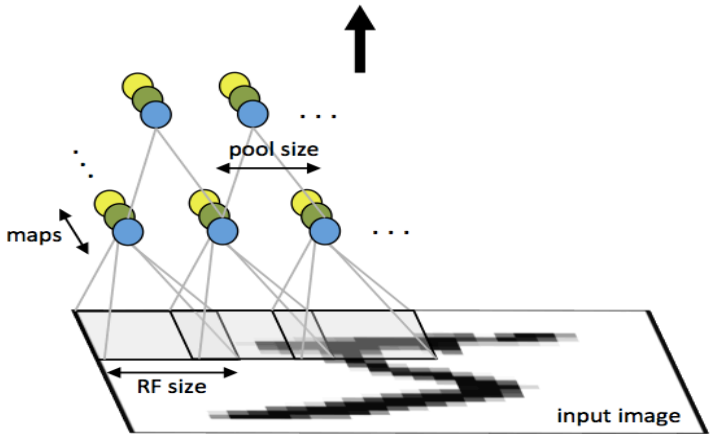
2.1 Convolutional Neural Network

2.1.1 Overview

A Convolutional Neural Network (CNN) is comprised of one or more convolutional layers (often with a subsampling step) and then followed by one or more fully connected layers as in a standard multilayer neural network. The architecture of a CNN is designed to take advantage of the 2D structure of an input image (or other 2D input such as a speech signal). This is achieved with local connections and tied weights followed by some form of pooling which results in translation invariant features. Another benefit of CNNs is that they are easier to train and have many fewer parameters than fully connected networks with the same number of hidden units.

2.1.2 Architecture

A CNN consists of a number of convolutional and subsampling layers optionally followed by fully connected layers. The input to a convolutional layer is a **m*m*r** image where **m** is the height and width of the image and **r** is the number of channels, e.g. an RGB image has **r=3**. The convolutional layer will have **k** filters (or kernels) of size **n*n*q** where **n** is smaller than the dimension of the image and **q** can be either same as the number of channels r or smaller and may vary for each kernel. The size of the filters gives rise to the locally connected structure which are each convolved with the image to produce **k** feature maps of size **m-n+1**. Each map is then subsampled typically with mean or max pooling over **p*p** contiguous regions where p ranges between 2 for small images (e.g. MNIST) and is usually not more than 5 for larger inputs. Either before or after the subsampling layer an additive bias and sigmoidal or relu nonlinearity is applied to each feature map. The figure below illustrates a full layer in a CNN consisting of convolutional and subsampling sublayers. Units of the same color have tied weights.



After the convolutional layers there may be any number of fully connected layers. The densely connected layers are identical to the layers in a standard multilayer neural network.

2.2 Network used

2.2.1 Description

We evaluate the applicability of deep convolutional neural networks for the classification problem described above. We focus on architecture named VGG16 which were designed in the context of the "Large Scale Visual Recognition Challenge" (ILSVRC) for the ImageNet dataset.

The runner-up at the ILSVRC 2014 competition is dubbed VGGNet by the community and was developed by Simonyan and Zisserman . VGGNet consists of 16 convolutional layers and is very appealing because of its very uniform architecture. Similar to AlexNet, only 3x3 convolutions with stride of 1 and same padding, and MaxPooling layers with filter size of 2 and stride of 1, but lots of filters. Trained on 4 GPUs for 2–3 weeks. It is currently the most preferred choice in the community for extracting features from images. The weight configuration of the VGGNet is publicly available and has been used in many other applications and challenges as a baseline feature extractor. However, VGGNet consists of 138 million parameters, which can be a bit challenging to handle.

The Vgg16 network consists of 5 convolutional blocks. The 1st and 2nd block consists of 2 convolutional layer and a max pool layer, 3rd, 4th and 5th block consists of 3 convolutional layer and a max pool layer. It also consists 2 fully connected layers at the end and a softmax output layer.

2.2.2 Architecture

```
Layer (type)                 Output Shape              Param #
=================================================================
```

input_1 (InputLayer)	(None, 224, 224, 3)	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
flatten (Flatten)	(None, 25088)	0
fc1 (Dense)	(None, 4096)	102764544
fc2 (Dense)	(None, 4096)	16781312
predictions (Dense)	(None, 1000)	4097000

2.3 Methodology and Results

2.3.1 Methodology

Data Set description

- Analyzed 54,306 images of plant leaves, which have a spread of 38 class labels assigned to them. Each class label is a crop-disease pair, and we make an attempt to predict the crop-disease pair given just the image of the plant leaf.
- Also attempted to predict just the disease name, which reduces the class labels to 21. Later we also attempted to distinguish between diseased and healthy plant which reduces the task to binary classification.
- Resize the images to 224*224 pixels, and perform both model optimization and predictions on these downscaled images.
- To get a sense of how our approaches will perform on new unseen data, and also to keep a track of if any of our approaches are overfitting, we run all our experiments across a whole range of train-test set splits, namely 80–20 (80% of the whole dataset used for training, and 20% for testing).

CNN training

- Took a pretrained VGG16 model trained on image net dataset and removed its output layer and added 2 fully connected layers with 512 hidden units each.
- For all the three cases (38 classes, 21 classes, 2 classes) we added the output softmax unit accordingly.
- Froze all the layers except the fully connected layers that we added and the output layer.
- Trained the models on our dataset.
- Used Keras deep learning framework with TensorFlow backend for training and testing models.

Hyper Parameters used –

- Epochs = 2
- Batch size = 64
- Optimizer = Adams

- Loss function = categorical-crossentropy.
- Learning rate = 0.001, beta_1 = 0.9, beta_2 = 0.999
- Solver Type = Batch gradient descent

SVM on CNN features

CNN models that are pre-trained on ImageNet can be used as generic feature extractor for other tasks. This is done by removing top output layer, adding one or two fully connected layer and using activations from last fully connected layer as features. It turns out that this off-the-shelf feature extractor give features that yield better results than handcrafted features.
But it is not always the case that last fully connected layer gives us the best features for classification, as sometimes our network might get confused when we go deeper in network, so we took activations form other layers as features and inputted in SVM in order to check which layer gives us the best set of features for our classification purpose.

- Extracted features from last layer of block 3, 4, 5 and last fully connected layer of all the three models.
- Took a random set of 8751 images from our dataset and used 10-fold cross validation for testing accuracy and avoiding overfitting.
- Inputted the features in SVM in order to see which layer gives us the best accuracy.
- Used Scikit-learn for implementing SVM.
- Kernel = linear, penalty = l2, loss = squared-hinge
- Multiclass classification is done according to one-vs-rest scheme

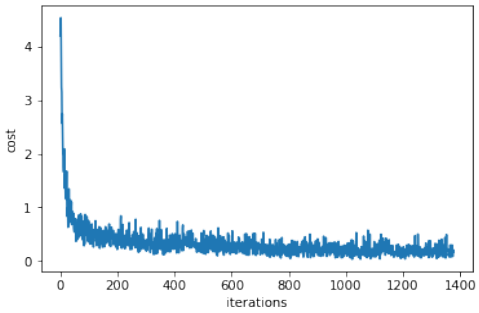
Important Test Cases –

At the end we handpicked 215 important images from across all the classes that could possibly be the difficult cases for our network and then test our models on those images to check the performance.

2.3.2 Results and Discussion

38-class classification (crop name + disease name)

- Model was trained on 43,444 images and tested on 10,861 images.
- Model was trained for only 2 epochs as it converged (as seen in the graph below).



- Accuracy on test set came out to be 90.6340843%.
- When features from different layers were inputted in SVM, last layer of block 3 in our architecture gave the best accuracy.
- The accuracy of our model on typical hand-picked important cases came out to be 81.3953488372%.
- Some of the correct predictions made by our model are shown below.

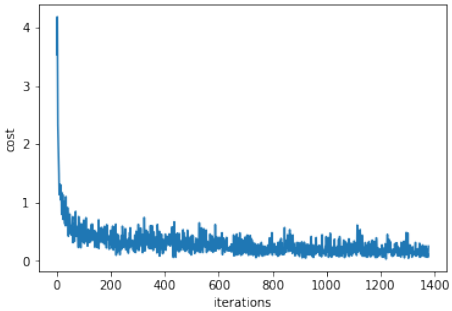
Figure 1 predicted - Cherry_(including_sour) ___healthy original = Cherry_(including_sour) ___healthy

Figure 1 predicted = Peppr,_bell___healthy original = Pepper,_bell___healthy

Figure 3 predicted = Corn_(maize)___Northern_Leaf_Blight original = Corn_(maize)___Northern_Leaf_Blight

21 – class classification (disease name or healthy)

- Model was trained on 43,444 images and tested on 10,861 images.
- Model was trained for only 2 epochs as it converged (as seen in the graph below).



- Accuracy on test set came out to be 92.278343023%.
- When features from different layers were inputted in SVM, last fully connected layer in our architecture gave the best accuracy.

- The accuracy of our model on typical hand-picked important cases came out to be 86.046511627%.
- Some of the correct predictions made by our model are shown below.

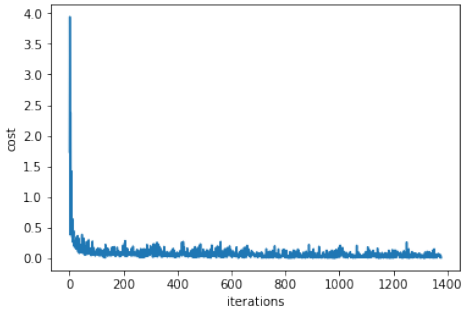
Figure 1 predicted Esca_(Black_Measles) original = Esca_(Black_Measles)

Figure 2 predicted = Esca_(Black_Measles) original = Esca_(Black_Measles)

Figure 3 predicted =Leaf_blight original = Leaf_blight

2 – Classification (healthy - diseased)

- Model was trained on 43,444 images and tested on 10,861 images.
- Model was trained for only 2 epochs as it converged (as seen in the graph below).



- Accuracy on test set came out to be 98.9734738372%.
- When features from different layers were inputted in SVM, last fully connected layer in our architecture gave the best accuracy.
- The accuracy of our model on typical hand-picked important cases came out to be 96.2790697674%.
- Some of the correct predictions made by our model are shown below.

Figure 4 predicted = healthy original = healthy

Figure 6 predicted = diseased original = diseased

Figure 5 predicted = diseased original = diseased

Model / Accuracy	Images taken	2-classes	21-classes	38-classes
CNN classifier	10861	98.9734738372%	92.2783430233%	90.6340843%
SVM on features extracted from last fc layer	8751	98.9259088758%	95.2814629363%	95.1117271991%
SVM on features extracted from last layer of 5th block	8751	97.6004803850%	94.0020893847%	95.1308870005%
SVM on features extracted from last layer of 4th block	8751	98.2632599366%	95.1774412182%	95.8260185339%
SVM on features extracted from last layer of 3rd block	8751	97.8743747098%	93.6917301269%	94.5239910594%

- The models are not tested on images taken under different conditions from the images used for training.
- The model currently constrained to the classification of single leaves, facing up, on a homogeneous background. While these are straightforward conditions, a real world application should be able to classify images of a disease as it presents itself directly on the plant.
- When tested on typical images picked from the data set, accuracy of all the models of reduced significantly and it was noticed that model generally gets confused when the image is not under proper lighting condition or the leave is folded.
- A more diverse set of training data is needed to improve the accuracy. The results indicate that more (and more variable) data alone will be sufficient to substantially increase the accuracy.
- New image collection efforts should try to obtain images from many different perspectives, and ideally from settings that are as realistic as possible.
- A different architecture can be used like Inception_v3 which can improve the accuracy drastically.
- Data Augmentation like color shifting, flipping, rotation etc. can be used to increase the training data.

From the above results we can conclude that we need more imaged in different environment in order to make our model useful in real life situation. We also found out that reducing the number of classes can improve our accuracy tremendously. We also noticed that it is not always the case that last layer of our architecture gives us the best features for our classification task, sometimes middle layers may give us optimal features. Also supervised learning using deep convolutional neural network architecture is a practical possibility even for image classification problems with a very large number of classes, beating the traditional approaches using hand-engineered features by a substantial margin in standard benchmarks

3. Conclusion and Future Work

3.1 Conclusion

The performance of convolutional neural networks in object recognition and image classification has made tremendous progress in the past few years. Previously, the traditional approach for image classification tasks has been based on hand-engineered features, such as SIFT, HoG , SURF, etc., and then to use some form of learning algorithm in these feature spaces. The performance of these approaches thus depended heavily on the underlying predefined features.

End-to-end supervised training using a deep convolutional neural network architecture is a practical possibility even for image classification problems with a very large number of classes, beating the traditional approaches using hand-engineered features by a substantial margin in standard benchmarks. The absence of the labor-intensive phase of feature engineering and the generalizability of the solution makes them a very promising candidate for a practical and scaleable approach for computational inference of plant diseases.

We trained a deep convolutional neural network to identify 38 classes with name of crop and its disease (or absence thereof). The trained model achieves an accuracy of 90.23% on a held-out test set, demonstrating the feasibility of this approach. We also trained a network to identify 21 classes that included the name of disease in the given crop. This model achieved an accuracy of 92.27%. At the end we trained a network to distinguish between healthy and diseased crop and got a test accuracy of 98.89%.

Thus we can conclude that reducing number of classes to just recognizing diseased and healthy can increase the accuracy of the model tremendously. Also if we just detect the name of disease, it can also increase the accuracy of the system.
We can also conclude that it is not always the case the last layer of the model is giving us the best features, sometimes middle layers may also give us features which are best for our classification, which we can see from our SVM results.

What's more, in the future, image data from a smartphone may be supplemented with location and time information for additional improvements in accuracy. Last but not least, it would be prudent to keep in mind the stunning pace at which mobile technology has developed in the past few years, and will continue to do so. With ever improving number and quality of sensors on mobiles devices, we consider it likely that highly accurate diagnoses via the smartphone are only a question of time.

3.2 Future Work

- To select a different, faster and deeper architecture (like inception_v3) and check accuracy on that.
- To cluster features and check which of the following feature is more important and thus give it more weight during training.
- Collect data set from different environment and conditions which can help our model to perform better in real life situation.

4. Bibliography

- Data - https://github.com/salathegroup/plantvillage_deeplearning_paper_dataset
- Mohanty SP, Hughes DP and Salathé M (2016) Using Deep Learning for Image-Based Plant Disease Detection. *Front. Plant Sci.* 7:1419. doi: 10.3389/fpls.2016.01419
- Ben Athiwaratkun and Keegan Kang, "Feature Representation In Convolutional Neural Networks arXiv:1507.02313v1 [cs.CV] 8 July 2015
- Karen Simonyan and Andrew Zisserman, "Very Deep Convolutional Networks For Large-Scale Image Recognition," In Proc. ICLR 2015
- https://github.com/fchollet/keras
- https://github.com/fchollet/deep-learning-models