UNIVERSITÀ
DEGLI STUDI
DI MILANO

Algorithms for Massive Datasets

Leaf Image Classification in Tensorflow

Luca Graziano, ID:987996

August 29, 2023

Abstract

The following project has been developed with the aim to building a fast and highly scaleable image classifier of 11 different kinds of plants. The model built is a convolutional neural network (CNN) with automatic hyperparameter tuning. It achieves a % on the test set composed of 55 images uniformly distributed across the 11 different classes of leaf.

I declare that this material, which I now submit for assessment, is entirely my own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my work. I understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion or copying. This assignment, or any part of it, has not been previously submitted by me or any other person for assessment on this or any other course of study

Introduction

The analysis follows the data processing and modeling of a deep learning based leaf classifier. The data processing involves the resizing of images to

Data and Preprocessing

The dataset consists of 7 GB of images of leaves of 11 different plant species. They contain both healthy and diseased images, but for this analysis only the diseased ones have been fed to the model for training. The train set consists in 2111 images while both the validation and test set consist in 55 images each.

The preprocessing and augmentation of the images follows four steps:

- Image resizing to 256 x 256 to reduce model's training time
- random flipping to either vertical or horizontal
- randomised rotation of the image
- randomised alteration of brightness

Data augmentation ensures the model is not memorizing images over the training epochs but rather allow it to learn the defining characteristics.

Setup and Evaluation

The evaluation metric chosen for analysing the results and comparing the models is accuracy computed as the fraction of the number of correctly predicted images over the total number of images. In this case accuracy suffices as an evaluation metric since the classes of the dataset are extremely balanced. The loss function considered for this problem is sparse categorical crossentropy.

The default optimiser for the CNN is the widely popular Adam optimiser that has proven to return the best results on a number of different problems.

The model has been trained over 50 epochs. Hyperparameter optimisation has been carried out automatically using the highly efficient and convenient keras tuner framework [1]. More specifically, dropout values and number of dense layer units and number of convolutional layer filters have been optimised through 10 trials. Finally, Tensorflow AUTOTUNE achieves better hardware performance by creating an efficient input pipeline that delivers data for the next step before the current step has finished.[2]

Convolutional Neural Network

A convolutional neural network is a particular neural network tipically used to perform Computer vision tasks. It receives images as inputs and outputs probabilities of how much the image belongs to each class. It is composed of consecutive convolutional blocks followed by a dense layer and an output layer. A Convolution layer performs convolutions, that is the process of applying filters to an input that results in activation in order to identify and learn specific features, that can either be a simple as lines or more complex ones, across the whole image. Adding multiple convolutional layers on top of another, allows the CNN to identify higher level features.

First a 2D convolution layer is run with kernel size of 3x3. The nodes are activated or not based on a ReLu activation function. Following this layer, there is a Max Pooling layer with the default size 2x2. This layer downsamples the convolutional layer by replacing every 4 pixels with the maximum value among them. Lastly, there is a Dropout layer that randomly turns off a proportion of the computed nodes during each training epoch preventing the model from overfitting. In total there are 3 of these convolution blocks.

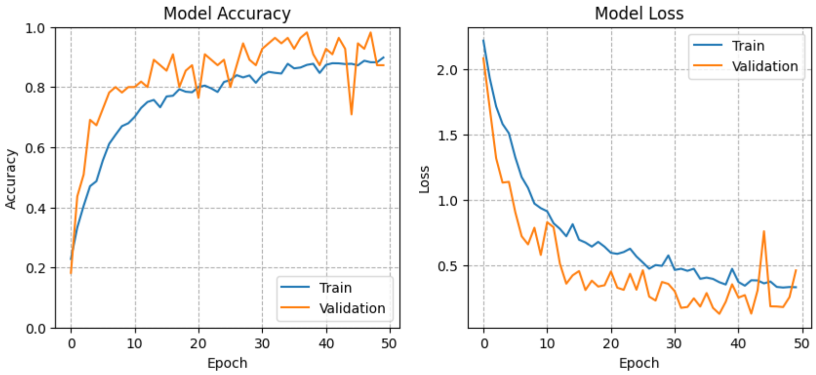
[1] https://keras.io/keras$_{t}uner/$

[2] https://www.tensorflow.org/guide/data$_{p}erformance$

The output from all the convolution layers is flattened and passed on to a dense layer followed by a dropout layer. The final dense layer has as many nodes as are the classes and it is activated using the softmax function resulting in the probability for each image to belong to one of the 11 classes.

Results

The model has been trained over 50 epochs and on data that has never seen before the accuracy tested to be a 94%score. The following are the model accuracy and loss for the train and validation set during the fitting process.


Model Accuracy
Accuracy
Epoch
Train
Validation
Model Loss
Loss
Epoch
Train
Validation