Leaf Area Index Prediction

Authors: Jeroen Crompvoets, Rauf Akdemir, Guus van Heijningen

Kaggle:

Introduction

This technical document explains the process of predicting the leaf area index (LAI) for basil plants as was carried out by PEAX Data with Logiqs. The first part explains the inner working of the system and the second part covers how this can be productionised in a cloud environment.

Table of contents

Making the predictions

Dataset

The dataset consists of 50 image-pairs containing a top and a front image of the basil plant. Below is an example of such pair.

Top image

Front image

Model design choices

The design of the model was largely determined by its ability to work on relatively small (n<1000) datasets with decent and stable accuracy. We decided to make a regression model that uses the total perceivable green area (range) in the photos as its estimators for the LAI.

The way we currently see the problem is that if the dataset becomes larger (n≥1000) a convolutional neural network (CNN) can outperform the current system. One of the

inherent problems with “plainly” calculating the total green pixel-range in the photos is that does not capture the 3D reality of the plant, which imposes a hard cap on the model accuracy. It could be the case that a leaf is *down- or upwards facing the west or east-side* of the images above. The estimator contribution of this leaf would be minimal, while the actual LAI is unaccounted for, which results in lower accuracy.

Leaf surface for camera angle is close to maximal

Leaf surface for camera angle is close to minimal

The structure of a CNN could provide a solution to this as it has the ability to capture and learn about intricate details in images. It could for example be able to spot the venation pattern on leaves. It also has the ability to make more accurate predictions by effectively combining specific areas of the two images.

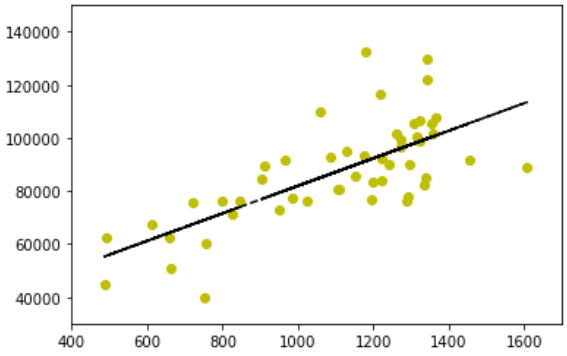
Feature selection

As we decided on making the model stable for the current dataset sizes as opposed to a model that could hypothetically work with large datasets, we finally came up with the following features that would allow us to achieve decent results without overfitting:

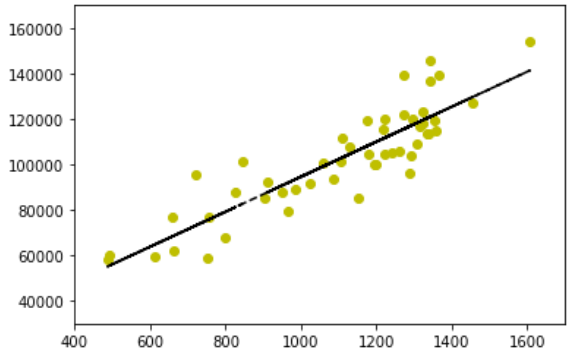
- Total green area range top image
- Total green area range front image

- Centroid of isolated area in top image (center of mass)
- Centroid of isolated area in top image (center of mass)

As can be seen, the green areas are clearly correlated with the total LAI.



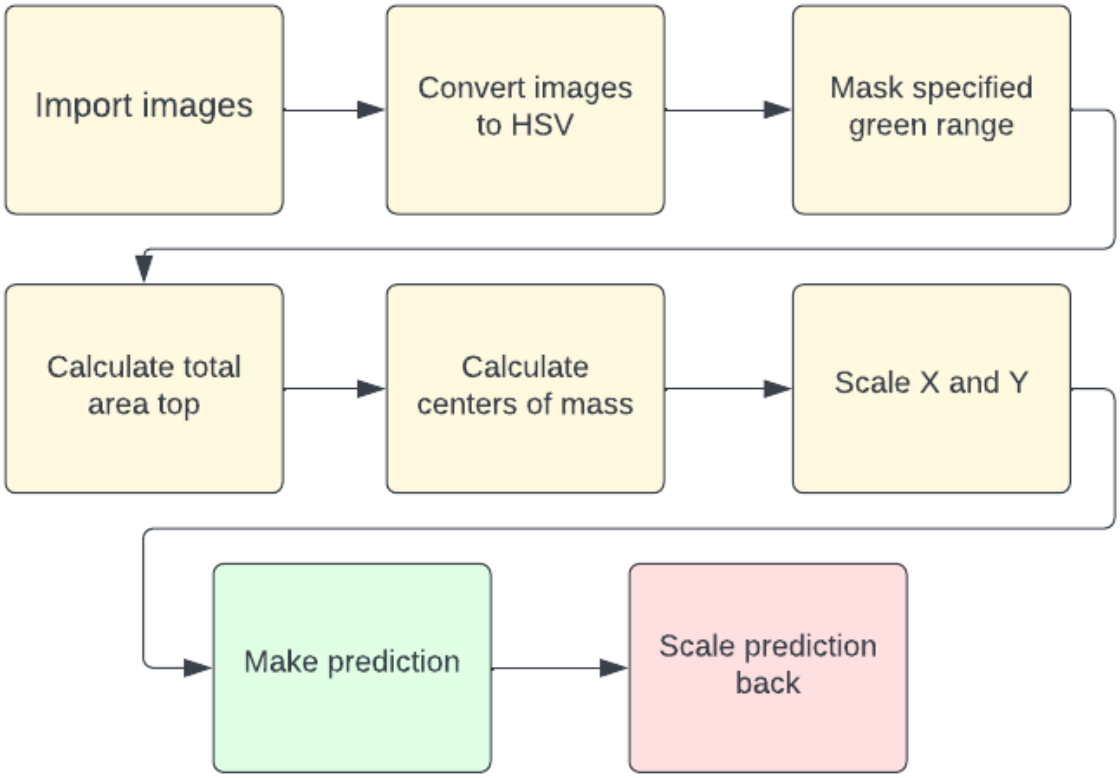
Ordinary Least Squares for Area front-LAI



Ordinary Least Squares for Area top-LAI

Pre-processing

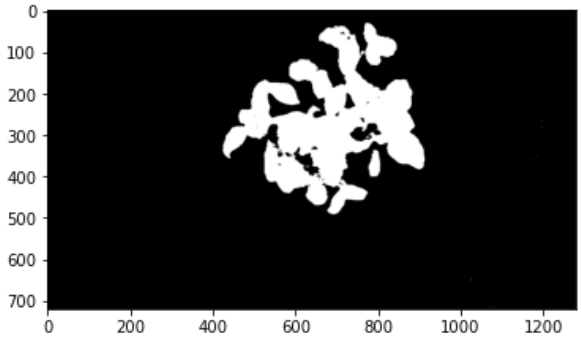
The processing pipeline is a multi-step approach to process the images to a quantity that can be an input to our model. The steps are as follows:



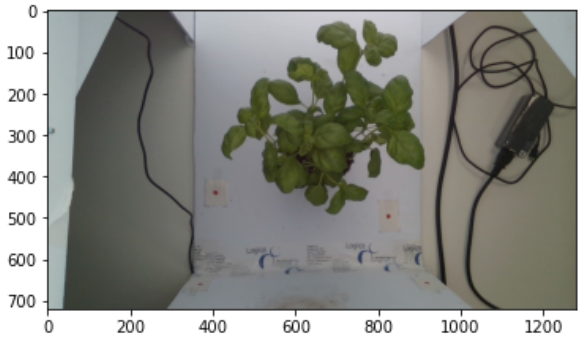
Final processing pipeline. In yellow are the pre-processing steps, in green the prediction steps and in red the post-processing steps.

A few elaborations on the steps that can be seen above:

1. **Convert images to Hue-Saturation-Value (HSV)**. This image format has allowed us to make a clear selection of the green parts of the images that are of interest to us.
2. **Mask the specified green range**. This is the selection of the green parts. In the images below the 2D-selection of the plant can be seen.



Mask of green area.



Original photo.

3. **Scaling of X and Y**. In order to optimise and generalise model performance, we scale the features and label with the standard scaling technique. The inverse is also used to scale the prediction back.

$$z = \frac{x - \mu}{s}$$

Applied standardisation.

Training

We have followed our conventional but proven method of iterating through various models which we believe make sense in this type of problem.

Models

The models we have tried and gone through are the following:

- Linear regression
- N-layer neural network
- AutoML
- XGBoost
- Random Forest

This allowed us to make a pre-selection before actually validating the model performances on the dataset using a K-fold validation. The way we measure the model performance is by taking the the error ratio with the following formula.

$$Error\ ratio = \frac{MAE}{\mu},\ where$$
$$MAE = mean\ squared\ error$$

The result of the final model performances can be seen in the table:

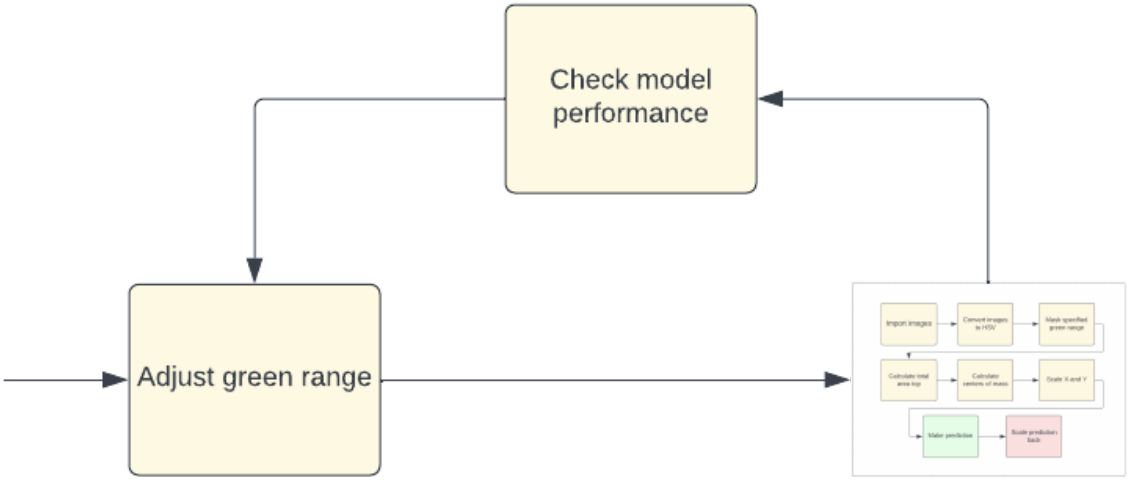
Model	**Performance (error)**
2- layer neural network	0.08788
AutoML	0.11252
Random Forest	0.10381

From these results we can see that the **2-layer neural network** outperforms the other models overall.

Optimising green area selection

The green area selection was mostly done by watching multiple samples being masked by the range that we specified. Despite the belief that this pre-processing step would perform within reasonable bounds, we decided to make the extra step of optimisation by iterating through multiple range-values and seeing what their outcome would be if we re-trained the model on the new data.

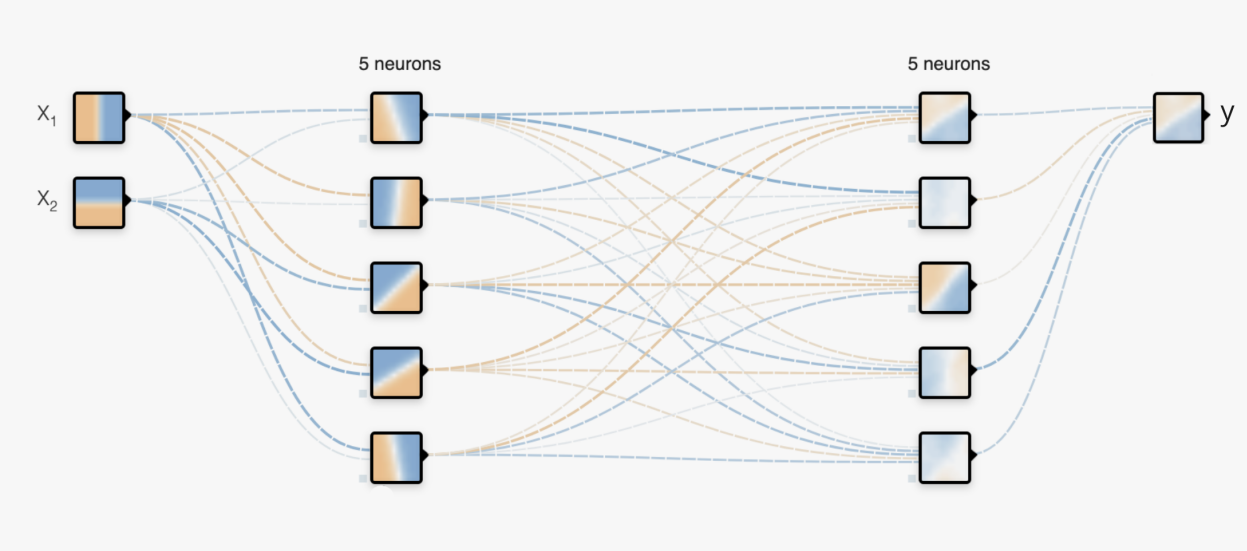
Since the neural network already outperformed the others, we used the neural network for this.



The green value range is adjusted after which the whole training pipeline is triggered and finally the model performance is checked.

Final model

The final model that we are using is the 2-layer neural network that takes in two parameters: the total top area and front area. The input layer(s) use a ReLu activation function and the second 5-neuron layer uses a hyperbolic tangent function. The structure of the neural network can be visualised as follows:



Neural network that performs prediction on x1 and x2, which are the top area and front area for a given sample.

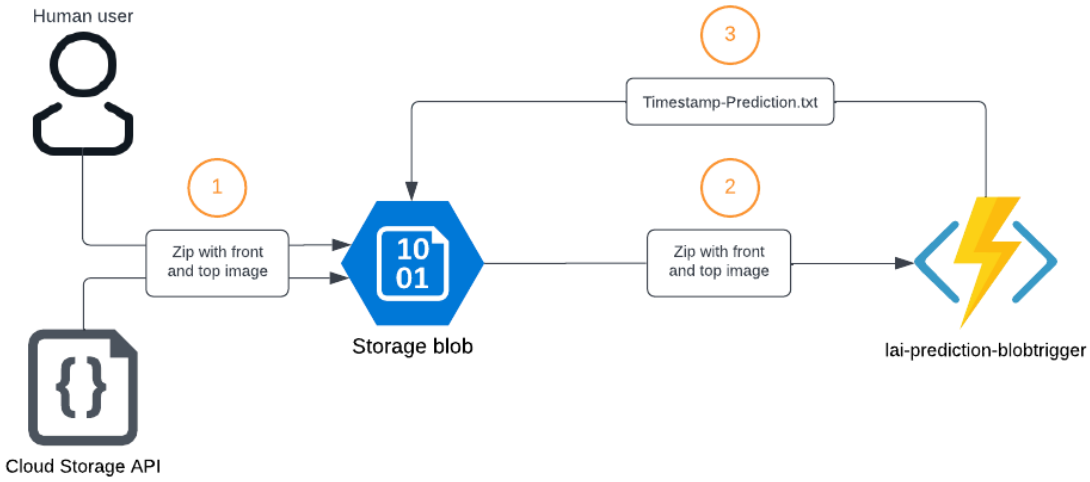
Bringing the model into production

This section discusses how the model runs in an Azure cloud environment. This architectural design can be applied to all popular cloud environments that provide serverless functions.

Higher architecture

A drawing of the higher architecture can be seen below. The process that this architecture follows is the following.

1. A human user or programme uploads a zip containing the front and top image to a blob storage.
2. A blob-triggered function app receives the zip, unpacks this and runs the ML pipeline on the contents.
3. The function app uploads the contents back to the blob (this can very easily be changed to an API endpoint or a database-write).



Directory structure

The directory structure to deploy the function is as follows. One important point is that the function fetches the scalers and models from its local file system. These are outlined in yellow.

Next steps

The next steps PEAX thinks are good to undertake in this domain are listed here.

More data

In order to make the models perform better (regardless of type) and more stable, we recommend gathering more data and re-iterating of the techniques that were currently applied. This should result in a better and more robust system.

Other plant types

Despite the positive outcomes for the basil plant for the low dataset size, we recommend that the same system is applied to different plant types. We have currently only tested this on basil - a leafy green. However, the only way to find out whether this same technique works as well on other plant types (those that yield fruits, for example) is by actually carrying out this same exercise.

Potential new models

Because of the small dataset size we were restricted in terms of what model we could apply. The models that we believe still should be tested if the dataset increases, are the following:

- Convolutional Neural Network
- Variable Autoencoder
- AutoKeras

Next steps in autonomous growing

The purpose of Logiqs is to create access to honest food. In line with this purpose, Logiqs aims to democratize food production by facilitating the possibilities of autonomous growing. With the help of autonomous growing food can be produced on a much larger scale and become much more accessible to the people that don’t have the right knowledge to grow food.

Autonomous growing will be facilitated through growth recipes. These growth recipes are optimized on the data of many historic growth cycles that took part in the vertical farming solution of Logiqs. A crucial factor for developing these growth recipes using data, is to measure the anticipated growth of the plants during a cycle.

Leaf area index (LAI) is a commonly used parameter to measure growth. Especially for leafy greens it tells a lot about the performance of a growth cycle. A challenge these

days is to measure the LAI in a non-intrusive way. Currently an entire plant needs to get stripped in order to measure the LAI. This is not scalable, especially if we also want to track the LAI during the cycle, which will be a requirement to increase data gathering.

This initial POC project is therefore a great starting point to further develop the ability of the Logiqs system to measure the LAI using cameras. The goal should be to not only measure the LAI if a batch is done, but also during the harvest cycle.

Therefore, the next step is to use the learnings from project and build a model that is able to measure the LAI of plants within the pod during the entire cycle.