Tree species with CNN-based detection

Nikil Ramesh
Master in AI Engineering
jönköping University
jönköping,Sweden
rani19lu@student.ju.se

Mohanavignesh Gopal Rajkumar
Master in AI Engineering
jönköping University
jönköping,Sweden
gomo19hc@student.ju.se

***Abstract*—Image/video processing for fruit detection in the tree using hard-coded feature extraction algorithms has shown high accuracy on fruit detection during recent years. Besides, the recent years have witnessed an impressive progress of deep learning methods for object detection. Motivated by this scenario, we proposed and evaluated the usage of Convolutional Neural Network (CNN)-based tree species detection. By building sequential model with number of layers the data is trained to predict the appropriate tree. Initially the image data is converted into array for the model to understand. Later it is normalized because the data which we use contains gray scale value. The experimental analysis delivered good accuracy with an associated processing times below 39ms/sample.**

Keywords—CNN, Normalization ,Activation, Sequential model, Detect tree

1.INTRODUCTION

Biodiversity is declining steadily throughout the world. The current rate of extinction is largely the result of direct and indirect human activities . Building accurate knowledge of the identity and the geographic distribution of trees is essential for tree identification is essential for effective study and management of bio diversity.

In a manual identification process, botanist use different tree characteristics as identification keys, which are examined sequentially and adaptively to identify tree species. This series of answered questions leads eventually to the desired species. Traditional tree species identification is almost impossible for the general public and challenging even for professionals that deal with botanical problems daily, such as, conservationists, farmers, foresters, and landscape architects. The situation is further exacerbated by the increasing shortage of taxonomists. Recently, taxonomists started searching for more efficient methods to meet species identification requirements, such as developing digital image processing and pattern recognition techniques. CNNs learn hierarchical combinations of image features that focus on the object-level, rather than pixel-level, representations of objects. Finally, neural networks are re-trainable to incorporate the idiosyncrasies of individual datasets. This allows for models to be refined with data from new local areas without discarding information from previous training sets.[5]

The influence and impact of digital images on the modern society is tremendous and is considered a critical component in a variety of application areas including pattern recognition, computer vision, industrial automation, and healthcare industries. Usually, the local visual features are extracted by a so called convolutional neural network (CNN), and a successive classifier consists of a fully connected network. CNNs preserve the spatial relationship between pixels by learning internal feature representations using small squares of input data.[1][2]

2.DATSET

Images were collected around jönköping,Sweden during the month of febraury . Multiple images of same tree have been taken with different angles and during different time of the month with different weather and light conditions. Two different common trees in this region (Larix decidua and Conifers) been selected for the data collection

Fig2.1: Dataset image

3.DATA PREPARATION

Data preparation and pre-analysis is the most important step in building ML algorithms. After images were collected together, we started the exploratory data analysis. Depending on the model we use, the images were resized into 256 pixels. Everything was done with a script written in Python, that would take a batch of images and output the desired scaling .

Images are divided into Training set and Testing set. For Testing we used only four images from the data. Those images went through all procedures of other images, until training the model. Those images are never shown to the training, as it is important for testing the accuracy of the models into images not seen before. This way we can see if

the model is over-fitting on those training images or is still able to generalize and detect tree on other image.

Fetching the right amount and type of data is a difficult task. In addition, the data should have good diversity as the object of interest needs to be present in varying sizes, lighting conditions and poses for our network to generalize well during the training phase. To overcome this problem of limited quantity and limited diversity of data, we generated our own data with the existing data which we have. This methodology of generating own data is known as data augmentation.[2]

4.METHODS AND PROCESS

I. CONVOLUTION NEURAL NETWORK

Convolutional neural networks are a specialized type of ANNs used for image analysis. Since computers sees image as a matrix of numbers that represent each pixel, it is important that the relation between the pixels (values) remains even after the image is processed through the network. To save this spatial relation between pixels, convolution neural networks are used that have different mathematical operations stacked on top of each-other to create layers of the network.

In the first part of this section it has been briefly explained the mathematical operations that serve as building blocks for the updated model architecture.[2]

A. Convolution

Convolution is a mathematical operation that takes two functions ($K_{i,j}^{(l)}$ and $Y_j^{(l-1)}$) to produce a third one ($Y_i^{(l)}$). It is the process of adding each pixel of the image to its local neighbors, weighted by the kernel/filter of m × n size. In each layer, there is a certain number of filters. The number of filters p that is applied in one stage is equivalent to the depth of the volume of output feature maps. Each filter detects a particular feature at every location on the input. The output $Y_i^{(l)}$ layer l consists of $p^{(l)}$ feature maps of size $m^{(l)} \times n^{(l)}$. Thus the i^{th} feature map is computed as:

$$Y_i^{(l)} = B_i^{(l)} + \sum_{i=0}^{p} Ki,\, j^{(l)} \times Y_j^{(l-1)}$$

where $B_i^{(l)}$ is bias matrix and $K_{i,j}^{(l)}$ is the kernel connecting the feature map j of previous layer (l - 1) with i^{th} feature map in current layer l.[2]

B. Activation

It is a function that takes the feature map $Y_j^{(l-1)}$ generated by the convolution and creates the activation map as output $Y_i^{(l)}$. It serves as a gate, to let certain part of the map elements pass while others not. This is strictly element-wise operation:

$$Y_i^{(l)} = f(Y_i^{(l-1)})$$

where *f* is the function that have been used as a multiplier.

C. Maxpooling

Max pooling is a sample-based discretization process. The objective is to down-sample an input representation (image, hidden-layer output matrix, etc.), reducing its dimensionality and allowing for assumptions to be made about features contained in the sub-regions binned.

This is done to in part to help over-fitting by providing an abstracted form of the representation. As well, it reduces the computational cost by reducing the number of parameters to learn and provides basic translation invariance to the internal representation.

Max pooling is done by applying a max filter to (usually) non-overlapping subregions of the initial representation.[3]

D. Flatten

Convolutional Neural Network whose initial layers are Convolution and Pooling layers. These layers have multidimensional tensors as their outputs. If we wanted to use a Dense(a fully connected layer) after the convolution layers, it is needed to 'unstack' all this multidimensional tensor into a very long 1D tensor. Flatten is used to achieve this.

E. Conv2D

A 2D convolution layer means that the input of the convolution operation is three-dimensional, for example, a color image which has a value for each pixel across three layers: red, blue and green. However, it is called a "2D convolution" because the movement of the filter across the image happens in two dimensions. The filter is run across the image three times, once for each of the three layers.

After the convolution ends, the features are down sampled, and then the same convolutional structure repeats again. At first, the convolution identifies features in the original image, then it identifies sub-features within smaller parts of the image. Eventually, this process is meant to identify the essential features that can help classify the image.[4]

II. PROCESS

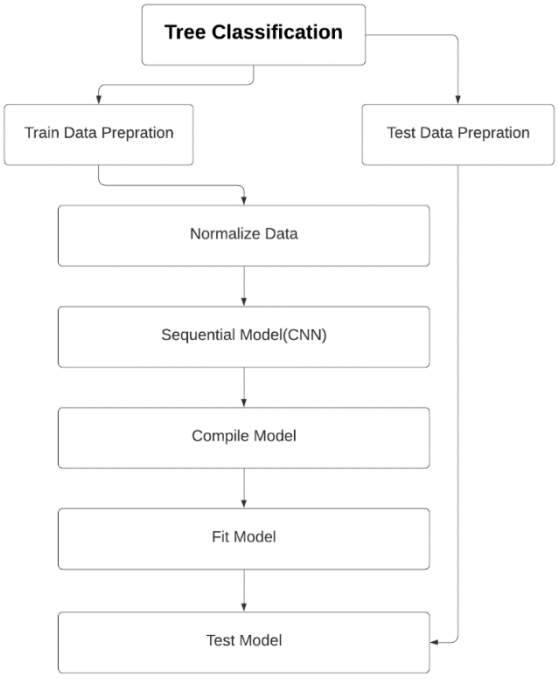

Fig4.1: Flow diagram of tree classification

For the classification of tree, the dataset is imported and categories where mentioned. In initial steps, tree image dataset is converted into array for easy understanding for the machine. The dataset is split into train data and test data and stored as a pickle file.

```
In [4]: #normalize
        X = X/255.0

In [5]: X

Out[5]: array([[[[0.81568627],
                 [0.81568627],
                 [0.82352941],
                 ...,
                 [0.84705882],
                 [0.84705882],
                 [0.84705882]],

                [[0.82352941],
                 [0.82352941],
                 [0.81960784],
                 ...,
                 [0.8627451 ],
                 [0.84705882],
                 [0.85098039]],

                [[0.82352941],
                 [0.82352941],
                 [0.82352941],
```

Fig4.2: Normalized data

The modules from the TensorFlow were imported and the data is normalized (X/255) because the data what we use contains gray scale values. A model is defined in our case we defined a sequential model. CNN need layers so we are using conv2D in the first and second hidden layers. In first layer the shape of layer is defined and the activation is 'Relu'-Rectifier Linear. In second layer also the activation is 'Relu'. Flatten is used to convert the 3D feature into 1D feature. Later dense layer and output layer is added with 64nodes in dense and Sigmoid as the activation in the output layer.

```
In [7]: x

Out[7]: array([[[[214],
                 [213],
                 [216],
                 ...,
                 [216],
                 [215],
                 [216]],

                [[214],
                 [215],
                 [214],
                 ...,
                 [216],
                 [218],
                 [215]],

                [[213],
                 [216],
                 [216],
```

Fig4.3: Train data

Configuring the model with the loss function, Optimizer and metrics then fitting the model with the parameters such as batch_size, epochs and the model is trained.

5.RESULT

In this section we present the result obtained by the model used, their detection speed after training and the number of images the models are tested. The model was trained with 40 images collected in *Jönköping*, *Sweden* and tested with 4 different images which were not included in the training dataset. Finally the model is tested in 39ms/sample and the test loss of the sample is 0.00048 and it predicts the tree accurately.

```
# Evaluating the model with the input evaluate function
results = model.evaluate(x_test, y_test, batch_size=128)
#printing the results like Loss and Accuracy
print('test loss, test acc:', results)

4/4 [==============================] - 0s 39ms/sample - loss: 4.8038e-04 - acc: 1.0000
test loss, test acc: [0.0004803756892215460 5, 1.0]
```

Fig4.4: Predicted data

6.CONCLUSION

In this work, we proposed and evaluated an approach for the detection of tree species based on CNN with high resolution images captured by smartphone. In the experiment carried out on a dataset comprising 40 images, it achieved the accurate result. Applying deep learning models to natural landscapes opens new opportunities in ecology, forestry, and

land management. Despite a lack of high-quality training data, deep learning algorithms can be deployed for tree prediction while using unsupervised detection to produce generated trees for pretraining the neural network.

The experimental results indicate that tree species with CNN-based detection algorithms constitute a promising approach towards the development of tree species, as well for demography monitoring, which is fundamental to integrate economic development and nature conservation. Future works will investigate the application of the proposed techniques considering other tree species. Real-time tree detection using embedded devices will also be investigated.

Current limitation of our platform is the compute power needed for the system to run. Because most neural networks have many layers, especially CNNs, the most suitable to run the models are the CUDA/OpenCL capable devices. Our continuation of this work will include more images of different tree species, at different growth stages, with different training systems.

7.ACKNOWLEDGMENT

We acknowledge Beril Sirmacek for the guidance and idea of this project.

8.REFERENCES

[1] https://www.researchgate.net/publication/312147459_Plant_Species_Identification_Using_Computer_Vision_Techniques_A_Systematic_Literature_Review

[2] https://www.frontiersin.org/articles/10.3389/fpls.2019.00611/full

[3] https://www.quora.com/What-is-max-pooling-in-convolutional-neural-networks

[4] https://missinglink.ai/guides/keras/keras-conv2d-working-cnn-2d-convolutions-keras

[5] https://www.mdpi.com/2072-4292/11/11/1309/htm