Canyu Hankins

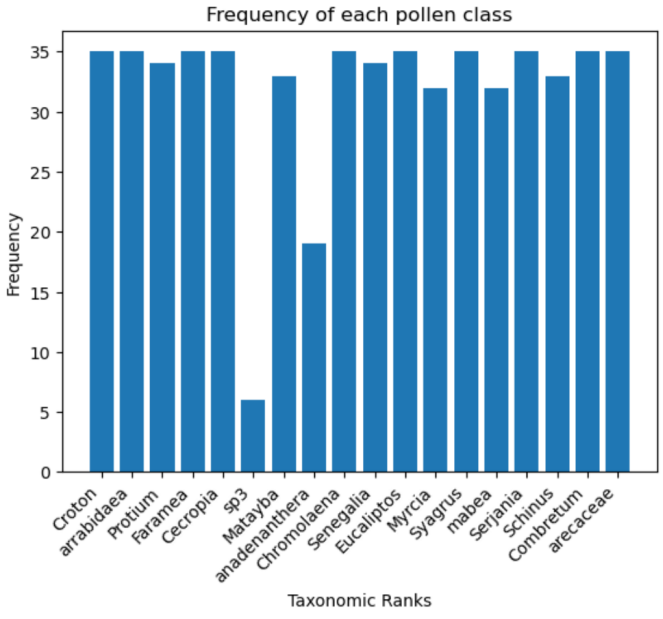
Pollen grain classification report

Introduction

This project is an image classification task to identify 18 different classes of pollen grain. The goal of this project is to use machine learning algorithms to do image classification and develop a model that can take pictures and automatically recognize the item correctly to create a database based on the pictures given. Therefore, classification for pollen grain is a great project to start with, and the same technique used in this project can be apply towards other similar task.

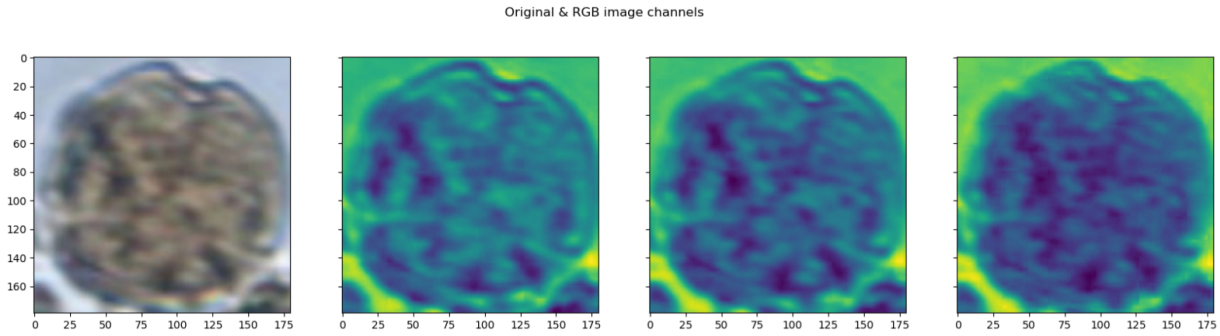
Dataset

The dataset comes in JPG format, and there are 573 images and 18 folders. The size of the dataset is ~18MB. The folders represent different taxonomic ranks of pollen grain, including: Croton, arrabidaea, Protium, Faramea, Cecropia, sp3, Matayba, anadenanthera, Chromolaena, Senegalia, Eucaliptos, Myrcia, Syagrus, mabea, Serjania, Schinus, Combretum, arecaceae. Most of them contains a similar amount of images inside each folder, around 30-35 images per folder, except for sp3 (6 images) and anadenanthera (19 images). Below is the frequency distribution.



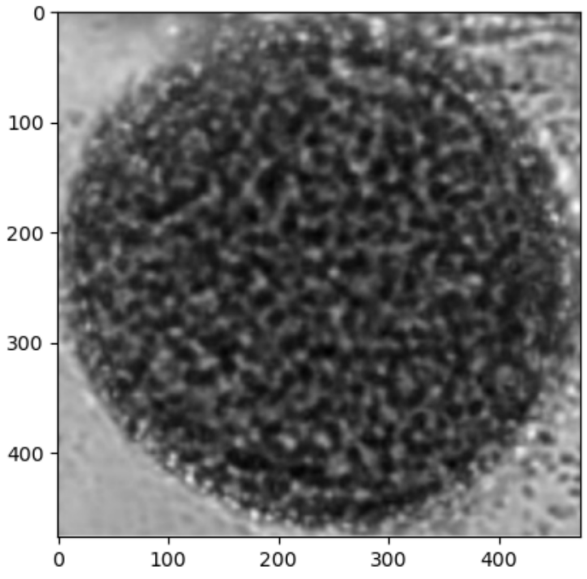
Data Preparation

The first step in the data preparation is to do data/image augmentation, since the dataset is somewhat unbalanced and small. Data augmentation is a technique to increase the training dataset by creating modifies copies based on the given dataset with existing data. In this case, I generated more images using RBG channels, adjusted contrast, and brightness, as well as applied rotation, and reflection, and also made each image the same size.



Exploratory Data Analysis (EDA)

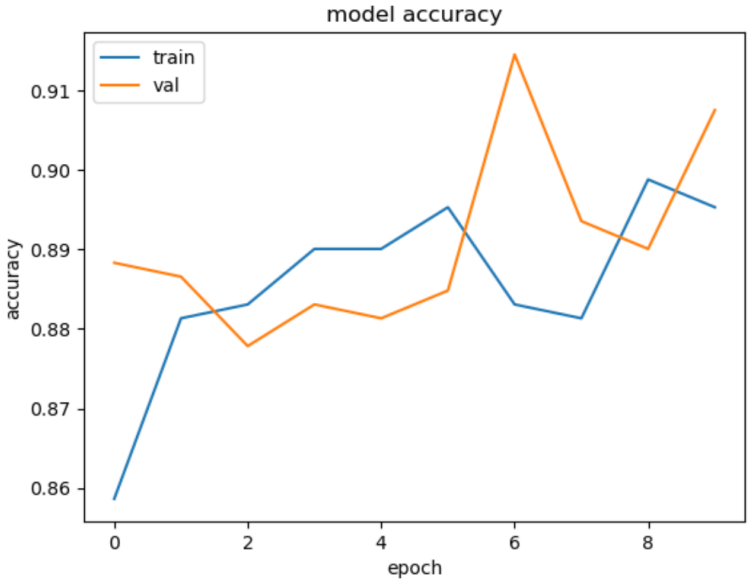
After the data has been prepared, the next step is to perform exploratory data analysis (EDA). EAD involves visualizing and summarizing the data to gain insights. For the pollen grain dataset, this involves plotting the image and converting them into image arrays, where we can check the average value of each pixel across all observations. Then we can compare the contrast and variability between average images.

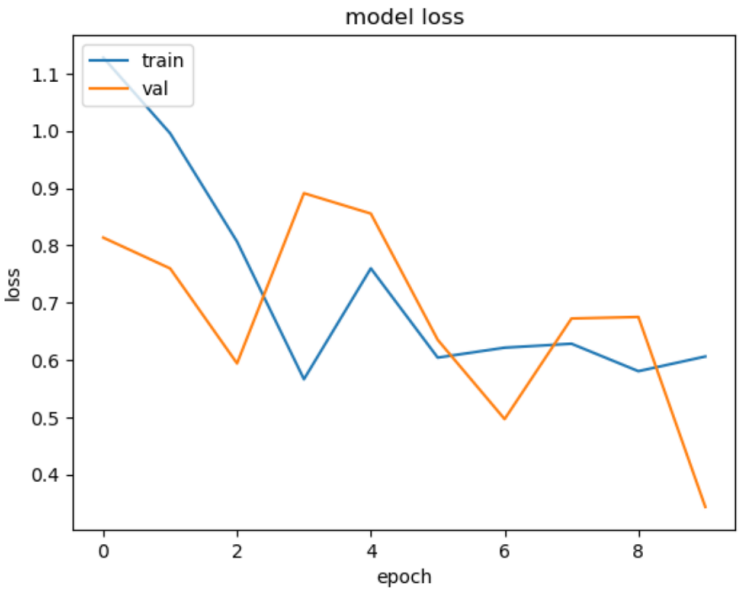


Model and Evaluation

The 2 models that I used are the base convolutional neural networks model and VGG16. The first model I tried is the base model, which is the sequential model, this model includes 3 layers of the convolutional network and an output layer, I trained this model with 5 epochs and the accuracy turn out to be ~6%. While training the model, the accuracy of the model fluctuates a lot, it would go up to about 10-12%, but it would go down to 6% when it's finished. This can be due to small dataset and high bias.

The second model I used is VGG16. VGG16 is one of the most popular model for image recognition tasks, and the 16 stands for 16 layers, which includes 13 layers of convolutional neural networks, 5 pooling layers and 3 dense layers. This model increases the accuracy drastically, with 3 epochs the model accuracy is ~82%, and with 10 epochs the accuracy is ~89%.





Future Work

The VGG16 model increases the overall model accuracy from ~6% to ~90%, which is very impressive, I'm planning on using another CNN model like ResNet and comparing ResNet with VGG and see the comparison between these 2 models. I also want to gather more data off the internet or kaggle, since transform learning sometimes works better for a larger dataset. Furthermore, I would also like to look into creating the database based on the result of the classification and working on the hardware inventory database. Overall, this is an interesting project to work on, and it's my first time dealing with image data, this gave me a strong understanding of image datasets and working with the os package in python.