Flower classification Presentation

A short synopsis of Lab 5

Jean-Sebastien Roger[1]

[1]Florida Atlantic University

Biostatistics, Fall 2020

Abstract

In this experiment, we were tasked with tuning a model to classify flower images with their correct genus. Our classifications consisted of 102 different species of flower each with their own distinct shape, size, and color. Below are a few of the more popular labels:

- Sunflower
- Pink primrose
- English marigold
- Snapdragon
- Daffodil
- Petunia
- Rose
- Lotus
- Magnolia

Our experiment was successful in the sense that, we were able to load images, properly transform and normalize our images into tensors, re-tune a ResNet-18 model using our training set, and correctly classify the majority of our validation set. The next slide has numerous examples of the images used in training.

These images below have been cropped, transformed, and normalized

Examples

Procedure

Using python code, we:

1. Read in our dataset of images
2. Crop each image and transform it into a tensor and then normalize the resulting tensor
3. Load in a pre-trained model, ResNet-18
4. Create and use a function to train the model on flower training set and validation set
5. Incorporate the trained model into another function for visualization
6. Prove the model's flexibility by reading a flower image from online and predicting the species

Model Training

The model training step was the most intensive section of this lab. In training we had to provide both the training and validation datasets and with each iteration our training function aimed to optimize the model. Within our training function we had three key tuning tools:

- Criterion - This allowed us to calculated the loss in each iteration
- Optimizer - This optimized our model allowing us to apply stochastic gradient descent
- Scheduler - This was what controlled the progression of the descent based on our learning rate and optimizer

In the end, we were left with a model_tuning function which we could call in a single line:

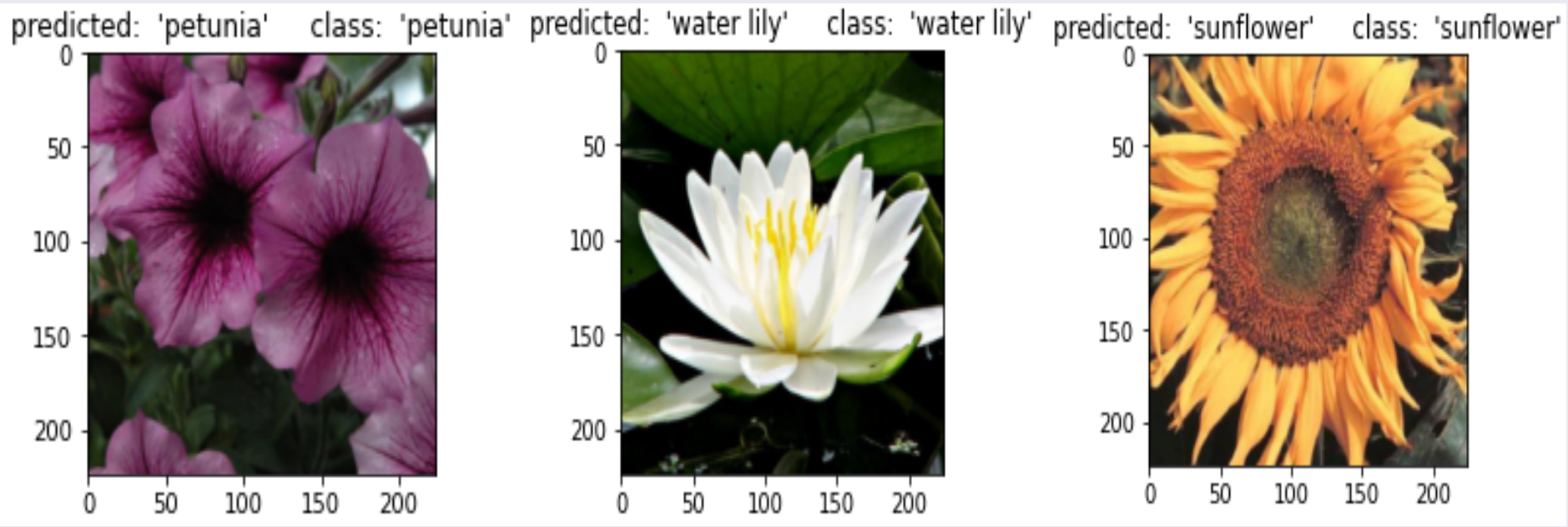
Final model train function call

$$model = train\_model(model, num\_epochs = 3) \tag{1}$$

Visualization

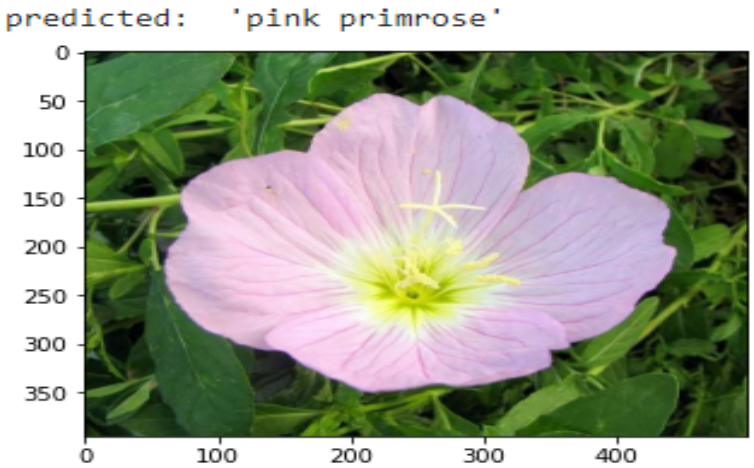
Our final visualization stitched everything together, allowing us to present the image, the prediction, and the label all in one visual

Example prediction results



Conclusion

From this experiment, we were able to train a pre-built model and validate its accuracy for our own image classification task, we were able to achieve a validation accuracy of around 96%. With the level of detail needed to distinguish certain characteristics within images and associate these characteristics with given labels, simple models (like that in lab 4) built on a single threshold do not hold up to the test.



Our final task within this lab was to ensure that our model could now be used on images outside of the typical datasets used to train and validate. We loaded the image above from an amazon URL and also produced a classification which also turned out to be and accurate classification.