Binary Classification of Tree Leaves using Convolutional Neural Network

Final Report

Drew Bostrom July 2021

Summary

Images of oak and maple leaves were extracted from cell phone video, taken by the author. Images were processed for input into a deep learning model built from one of three pre-trained models available by the keras module. The model was tuned for best optimizer and best learning rate as well as selecting for the most accurate pre-trained model. The final model was able to correctly identify images from a holdout set with 98% accuracy.

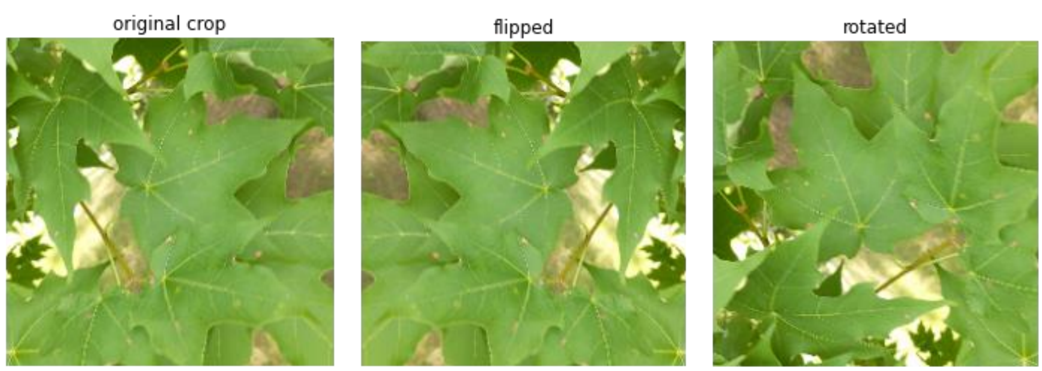
Problem Statement

Can images extracted from a few videos be used to develop a convolutional neural network (CNN) model that can accurately identify a leaf as oak or maple?

Data

Images used for model training were extracted from video taken on a Google Pixel phone. The speed of the camera is 30 frames per second and each frame is 1920 pixels x 1080 pixels. Every 10^{th} frame was extracted for further processing. Approximately 8 minutes of video were taken of both maple and oak branches. This process resulted in about 2800 near-photographic quality images of leaves.

Extracted frames were then further processed by cropping to a square in the center of the image and resizing to (224, 224, 3), the input suggested by the pretrained deep learning models. Finally, each image was then augmented in one of two ways: rotating 90 degrees counterclockwise and flipping left to right. The augmentation step resulted in 7000 images for model training, 3500 of each tree type.



To facilitate training, the 7000 images were grouped into 10 chunks with corresponding labels. Each chunk contained only maple or only oak, but the images within each tree type were randomized before chunking so they came from all videos, not just one or two. For each round of training, one chunk from of maple images and one chunk of oak images were combined and randomized. Therefore, the training data was balanced and randomized at all times during training.

Deep learning model with Convolution Neural Network (CNN)

Pretrained models

The deep learning classification model was built with the keras module. Three pretrained models were imported from the keras applications library: NASNetMobile, DenseNet169, and InceptionResNetV2. The parameters of these models were frozen—the weights of the model made untrainable. The last layer of the model was replaced with one containing just two nodes, one for each of the classes in the problem—maple and oak.

Training was performed on the Paperspace Gradient platform with a C7 instance featuring 12 CPUs and 35 GB RAM.

Hyperparameter tuning

Each model was tuned for two parameters: learning rate and optimizer. The learning rates tried were 0.1 and 0.001. The optimizers used were Adam, RMSprop, and SGD (stochastic gradient descent).

Other parameters used in the model were average pooling, binary crossentropy for the loss function, accuracy for the model metric, 3 epochs per chunk of training, and 28 images per batch.

Model selection

The model using the DenseNet169 pretrained model proved to be the best model by a couple of percentage points of accuracy. The RMSprop optimized proved to be superior for all runs and models. The ideal learning rate was a little less clear. High learning rates results in fast training, but progress was a bit unstable. In the end, a schedule was chosen that ramped learning rate down from 0.1 to 0.0001.

A summary of results is shown below.

Pretrained Model	Learning Rate	Best Accuracy (%)	Best Optimizer
NASNetMobile	0.1	93	RMSProp
	0.001	94	RMSProp
DenseNet169	0.1	97	RMSProp
	0.001	96	RMSProp, Adam
InceptionResNetV2	0.1	94	RMSProp
	0.001	89	RMSProp, Adam

Further training

The final model (DenseNet169, RMSProp) was trained from scratch with three rounds of decreasing learning rates (0.1, 0.001, 0.0001) with 3 epochs for each chunk of data.

Results

The final model was able to achieve 99% on the internal accuracy measure and 98% accuracy on the validation set by the end of training. However, earlier chunks did slightly worse, achieving 96 or 97% accuracy. Results seemed to vary a bit, then, depending on the particular data set.

To test how the model would do with unseen data, an evaluation set of data was created from videos not used in training. From these videos, images were extracted and processed the same as was done for training, resulting in 500 images for each of the tree types. On this evaluation set, the model classified the images with 98% accuracy. This shows that the model is generalizable, performing well on unseen data.

Conclusion

The results of this project show that images can be extracted from video to train a deep learning model. This finding can be useful for those who want to generate new data. Video cameras mounted to drones or car could generate an enormous number of images for model development.

Recommendations

- Use the model with confidence to correctly classify maple or oak leaves with 98% accuracy. Images of leaves should be high-quality – up close with good lighting. For higher accuracy, many images from the same tree can be inputted.
- More rounds of training can be performed on the same data to improve accuracy.
- The NASNetMobile performed well. If faster speed or smaller model size is desired, a model can be built from this pretrained model with good results.

Further Work

- Analyze what the specific weaknesses of the model are and address them with more data. Collect a wider range of images that include poor lighting (glare, high contrast), bad angles, and varying distances.
- Create a multiclass classifier that includes specific species of maple, oak, and other trees. This process can begin with a trinary model that is maple, oak, and none-of-the-above.
- Try unfreezing the model and training all parameters.
- Try training a model with black and white or grayscale images, including images with advanced processing, such as edge detection.