Capstone Proposal

Larry Hernandez
June 9, 2018

Domain Background

The problem I propose falls within the domain of Deep Learning for Image Classification. Given images of several types of object (i.e., cats), train a deep neural network that classifies each one of those images into exactly one of several sub-categories of that object (i.e., tuxedo cat). This is a common, contemporary problem which arises in many industries, including but not limited to e-commerce, robotics, and dating apps.

Deep convolutional neural networks (CNNs) are known to produce successful image classifiers, including handwritten digit recognition and skin cancer classification. In such cases, it is possible to use "transfer learning" to train a new CNN from an already existing CNN architecture. Rather than spending several hours developing a variety of neural network architectures from scratch, it is faster to take a pre-trained CNN and adapt it to the specific image classification problem at hand. This is because a pre-trained CNN already has the ability to recognize general patterns that are common for a variety of image objects. Subsequent training with the problem-specific data allows the CNN to learn the finer details that are relevant to the problem.

I am motivated to solve a problem in this domain because I enjoy image classification problems. I would also like to: (1) gain practice generating bottle-neck features with a pre-trained CNN and (2) use transfer learning in a scenario when the project data and the data that was used to train the established CNN model (aka ImageNet) do not have overlaps.

Problem Statement

This project is a Kaggle playground-level competition called Plant Seedlings Classification. Kaggle community members have the opportunity to develop algorithms that categorize images of plant seedlings into one of twelve classes. Some classes correspond to valuable crops and the rest correspond to (invaluable) weeds, which compete with crops for the same agricultural resources. If weeds are identified at an early growth stage—when they are seedlings—they can be physically removed, which then gives crops increased chances of prospering since they will not need to compete with the weeds for life-sustaining resources. Consequently, an effective image classifier installed on a smart phone can help farmers keep their plots (mostly) free of weeds and perhaps ultimately leading to increased crop yields.

Datasets and Inputs

The image data for this project are posted here on the Kaggle website. The images were acquired by researchers affiliated with University of Southern Denmark and Aarhus University in Denmark. Researchers utilized a dSLR camera to obtain 4,750 images of twelve different types of plant seedlings. Images of these seedlings were acquired over a 20 day period at 2-3 day intervals after they emerged from the soil in order to capture their appearance throughout several growth stages. The image classes are not balanced; the number of images for the plant classes ranges from 231 to 654. The images are in color, so they have red, green, and blue (RGB) channels, with the channels following that order. The widths and heights of the images vary. The plant seedlings are NOT segmented in these images, and on occasion the seedlings comprise

only a small portion of the images. That should not pose too much of a problem. The CNN can learn to ignore the non-plant portions of the images.

The images will be partitioned into one of three sets: training, validation, and test. The distribution of the classes in each of the three sets will reflect the distribution of the full data set. That is, if class 1 comprises 5% of the full data set, then it will comprise about 5% of the training set, about 5% of the validation set, and about 5% of the test set. The same will be true for all classes. The training set will contain approximately 2,852 images or 60 % of the 4,750 images, and the validation and test sets will each contain approximately 949 (or 20%) of the images.

Solution Statement

Deep CNNs are effective at learning patterns in images if given enough good quality training data and the appropriate architecture. For this project, a CNN can learn to recognize and distinguish the physical features of the various plant seedlings. To make model development easier, transfer learning will be utilized to train a new CNN specializing in the twelve plant seedling classes. The VGG19 architecture with pre-computed weights from training on the ImageNet database will be loaded. Several layers of the VGG19 model will be removed, since the training data and the ImageNet data do not contain overlapping classes. The number of model layers to discard will be determined through experimentation. In addition, a small neural network will be placed on top of the revised VGG19 network. It may consist of a flattened layer, a dense layer followed, a dropout layer, and finally another dense layer having twelve outputs, one for each class of plants. This architecture may change as the project progresses, but this is the starting idea. Keeping the pre-loaded VGG19 weights constant, the training and validation images will be passed through this combined CNN architecture to yield a new CNN that outputs probabilities for each of the twelve plant seedlings. The micro-averaged F1score will be used for evaluating the final CNN model. This is the official metric for the competition.

Benchmark Model

The benchmark model for this project will be a fairly shallow CNN that is trained from scratch. It will contain the following 14 layers: 2D convolution, 2D convolution, max pooling, 2D convolution, 2D convolution, dropout, max pooling, 2D convolution, 2D convolution, dropout, max pooling, flatten, dense layer, dense final output layer. Various values for the convolutional filter sizes, number of neurons for the dense layers, and the pool size will be tested to yield the optimal evaluation metric (described in the next section). This benchmark model is small and somewhat similar to the VGG19 model. It will likely perform worse than the VGG19-based model that will be developed via transfer learning. A pre-trained VGG19 model will have an advantage over the small CNN built from scratch, since it already has the capability of recognizing general patterns.

Evaluation Metrics

The evaluation metric for this problem has been defined in the competition. It is the micro-averaged F1 score. This metric is appropriate for classification problems having unbalanced classes. The mathematical expression for this metric is the following:

$$F1_{micro} = \frac{2Precision_{micro}Recall_{micro}}{Precision_{micro}+Recall_{micro}} \quad (1)$$

where

$$Precision_{micro} = \frac{\sum_{k \in C} TP_k}{\sum_{k \in C} TP_k + FP_k} \quad (2)$$

and

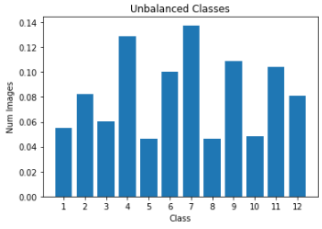
$$Recall_{micro} = \frac{\sum_{k \in C} TP_k}{\sum_{k \in C} TP_k + FN_k} \quad (3)$$

In these equations, TP represents the number of true positive predictions, FP represents the number of false positive predictions, and FN is the number of false negative predictions. In the summations, the index *k* represents the image classes. This micro-averaged F1-score is a much better metric than accuracy, which is a poor metric for non-balanced classes.

Project Design

The image data will first be assessed for quality. It is possible that some images are not useful—for example, some images do not contain a plant. In some cases the images may be very small, extremely blurry, or both. If it is clear that a human would not recognize the image as a plant or if the image is too blurry, then it will be discarded.

Second, the image data will be split into training, validation, and test sets in percentiles of 60%, 20%, and 20% of the full data set, respectively. The splits will maintain the relative proportion of each of the twelve classes. The figure below reveals the distribution of the classes in the full data set.



Once the image data have been partitioned into the three sets, they will be pre-processed for Keras. All images will be resized to 224x224. The images will also be re-scaled. For the VGG19 model, the keras function 'preprocess_input' will do this and also re-arrange the order of the color channels to be compatible with the order that is required by Keras.

After pre-processing, a CNN will be trained via transfer learning. First, a VGG19 architecture that has been pre-trained on the ImageNet database will be loaded. Several layers of the VGG19 model will be removed, since the training data and the ImageNet data do not contain overlapping classes. A global average pooling layer followed by a dense layer containing twelve output classes will be added. The pre-loaded VGG19 weights will be held constant, and the training and validation images will be fed through the VGG19 model to generate bottleneck features. The bottleneck features will then be fed through the aforementioned top-layer (see Solution Statement) for training.

I will test and compare several model parameters, such as number of pre-trained CNN layers to remove, number of densely connected neurons, etc. The CNN yielding the best score on the validation set will be chosen as the final model. The final model will then be used to make predictions with the test set, and a micro-averaged F1 score will serve as the final evaluation metric.