Project Overview

Our team sought to analyze the effectiveness of three machine learning libraries for image classification. Our goal was to compare the differences in classification accuracy among the three models we created versus a pre-trained model called “Mobile Net”. The images used in the classification were art datasets from five distinct styles, or genres, of artwork. We trained our models on a dataset that was composed of 4,400 images, while our validation dataset was composed of 1,900 images. Our sample size that we tested the models on was significantly smaller. This included 10 images per genre, which totaled 50 images.

Comparison Overview

0 – Drawing, 1 – Engraving, 2 – Iconography , 3 – Painting , 4 – Sculpture

The results shown above are three different confusion matrices that depict the results of our 50 test images. We used the same batch size for every model although, the Tensorflow and Keras models both had a total of 30 epochs, while the pre-trained model of Image Net only ran for 10 epochs. This unfortunately was due to our machine limitations, in terms of processing power, and time constraints. Analyzing the models above, Tensorflow had the highest accuracy when classifying images in our test dataset. This was surprising as Keras had the highest validation accuracy. The results showed our team the importance of data validation especially when feeding a model test data that is different from the original trained dataset.

Image Net Comparison

The results that are shown here were derived from a pre-trained convolutional neural network in Image Net. The model was downloaded from Keras.io, from a list of 10 other pre-trained models. We used 10 epochs and a batch size of 64. As discussed on the Comparison Overview page, we had to suffice for 10 epochs due to time constraints and processing power. Even with the limited epoch size, the model had a validation accuracy of approximately 70 to 80 percent. This is similar to our other models we created, but if we ran the pre-trained model for 30 epochs we predict a better result.

ScikitLearn Comparison

The results that are shown here were derived from the standard vector machine in SKLearn. The weighted average of this model was 72 percent, while the accuracy was approximately 73 percent. A few of the challenges that our team faced were resizing and loading truncated images for processing. We also had difficulty running tests on our datasets, since we had multiple directories that we needed to read in. In order to combat these issues, we utilized “SKimage.io import imread\_collection,” which allowed iteration through multiple directories. We also used “SKimage.transform” for our scaling issues. In addition, we also needed to wrap all of our data in “rgb2gray” to reduce the number of dimensions of certain images. The dimensions need to match for every image in order for the model to run correctly.

Tensorflow Comparison

The results that are shown here were derived from the convolutional neural network in TensorFlow. We used 30 epochs and a batch size of 64. The accuracy of this model was approximately 75 percent for training and 65 percent for validation accuracy. The challenges we faced in building this model were dataset size, training and validation split, overfitting, and debugging the model for maximum efficiency. By adding “dropouts,” which delete a specified amount of data, and shuffling we were able to decrease the amount of overfitting to get more accurate results. The other issues were combatted by moving data from training to validation in a 70 to 30 split. This increased our model efficiency and allowed for more accurate reporting.

Keras Comparison

The results that are shown here were derived from the convolutional neural network in Keras. We used 30 epochs and a batch size of 64. The value accuracy of this model was approximately 68 to 73 percent, while the training accuracy was 74 percent. Our team found that Keras was the most user-friendly in terms of building, compiling, and running the model. Having dealt with the challenges in TensorFlow, we were more prepared when working with Keras. This resulted in our team not facing many, if any at all, challenges when using this library. We also found that this machine library seemed to provide more functionality in terms of image processing.