Abbas Shakiba

Machine Learning Engineer Nanodegree

Dog Breed Classifier

Capstone Project Report

I. Definition

Project Overview

Image classification is one of the most critical research areas in machine learning, and it has numerous applications in different fields. E.g., Face recognition, Health care, weather, education, industry, etc.

In this project, a dog breed classifier is built to determine a dog's breed from an image. It is a challenging problem since there are hundreds of different breeds in different sizes, colors. Additionally, the pictures' size, location of the dog in the photos, and the dog's angle add more complexity. Thus, a deep neural network model, known as Convolutional Neural Network, or CNN, is used.

Problem Statement

This project aims to classify dogs' pictures based on their breeds if the picture is a dog and flag the picture as human if the image is human. The project assigns a breed to that picture if humans' images have been detected to add more fun to this project. Thus, the two main tasks are the human face detector and the dog breed detector.

Metric

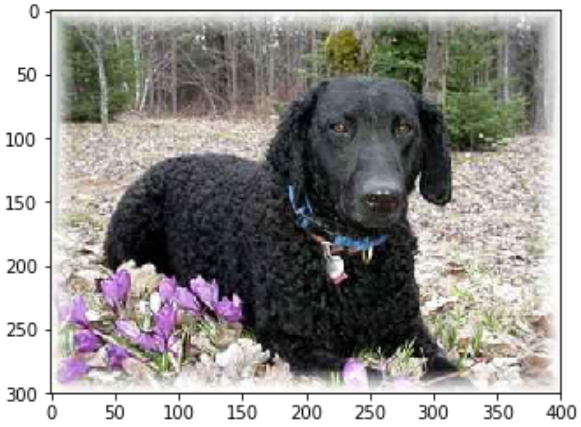
The Cross-Entropy loss function is applied for the model evaluation, which is the most common loss function for the multi-class classification. On average, there are 50 images for each breed, yet, some breeds have more pictures, and the data is only slightly imbalanced. Thus, to evaluate the model's performance on the test data set, I just cared about the model's accuracy for all the classes: the ratio of correctly predicted images over all the images.

II. Analysis

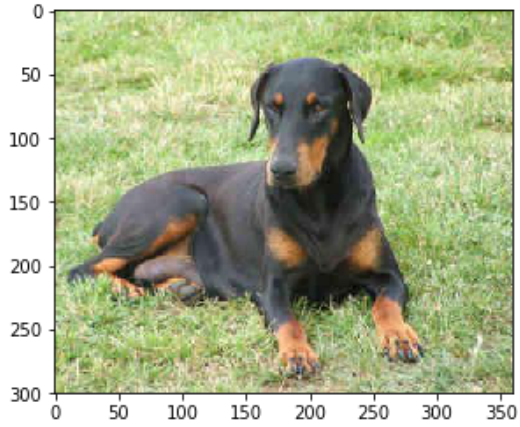
Data Exploration

The input for this problem is color pictures. There are two datasets of humans and dogs. The human dataset has 13233 pictures, and the dog image dataset has 8351 pictures. There are 133 different classes of breeds among dogs pictures. The dog data is split into three folders: Train (6680 pictures), Vali (835 pictures), and test (836) pictures.

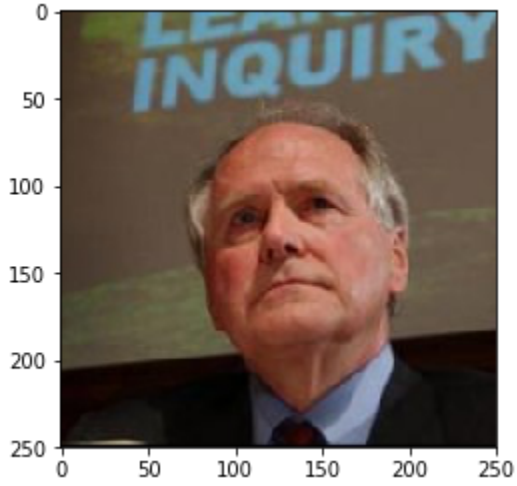
Exploratory Visualization



Curly-Coated Retriever



Doberman Pinscher



Algorithms and Techniques

Multi-class image classification is one of the most common problems in computer vision and has several applications. As mentioned previously, CNN is being used in this project. CNN used convolutional layers to create filters and, by adding depths to the input, extracts additional information, which eventually makes the model predict the class accurately.

Benchmark

The benchmark model trained on 100 epochs optimized by SGD (alpha=0.05) and Cross-Entropy as the loss function is helpful for multi-class classification problems. The test accuracy after 100 epochs was 31% (267/836), and the loss value was 2.739. Based on each epoch's loss values during the training part, it seems the model could perform better with the higher number of epochs.

The second model, the one with Transfer Learning, has 25 epochs only, optimized by SGD (alpha=0.01), and uses the Cross-Entropy loss function, which has an accuracy of 84% (704/836).

III. Methodology

Data Preprocessing

The input was resized (3,258,258), CenterCrop (224), and normalized before using them in the model. Additionally, to reduce the chance of overfitting and gain more information from the training dataset, horizontally flipped and rotation (0,30) are applied.

Implementation

For Human Detection Haar Cascades classifier is used, one of the common OpenCV algorithms for humans' face detection.

The first model has five convolutional layers, each doubling the following layer's depth by adding more filters. Each of the layers has a kernel with the size of (3,3) and having one stride and padding. A 2D max-pooling is applied to reduce the size of each convolutional layer. Eventually, the network will be flattened and followed by a four-layer

fully connected network with a drop-out in between each layer. The last layer of the network has 133 nodes which represents the number of classes.

The second model for taking advantage of well-known CNN models, Transfer Learning, is applied to transfer weight and filters from the VGG16 model. Similar to the first model, the network flattened, followed by a four-layer fully connected network with drop-outs in between.

Refinement

The benchmark model already has better performance, almost 1/3, than random classification, 1/133. Consequently, the transfer learning model performs better than the benchmark model since the model is already trained. Hyperparameter tuning is a way to increase the accuracy of moth models.

IV. Results

Model Evaluation and Validation

The benchmark model trained on 100 epochs optimized by SGD (alpha=0.05) and Cross-Entropy as the loss function is helpful for multi-class classification problems. The test accuracy after 100 epochs was 31% (267/836), and the loss value was 2.739. Based on each epoch's loss values during the training part, it seems the model could perform better with the higher number of epochs.

The second model, the one with Transfer Learning, has ten epochs only, optimized by SGD (alpha=0.01), and uses the Cross-Entropy loss function, which has an accuracy of 84% (704/836).

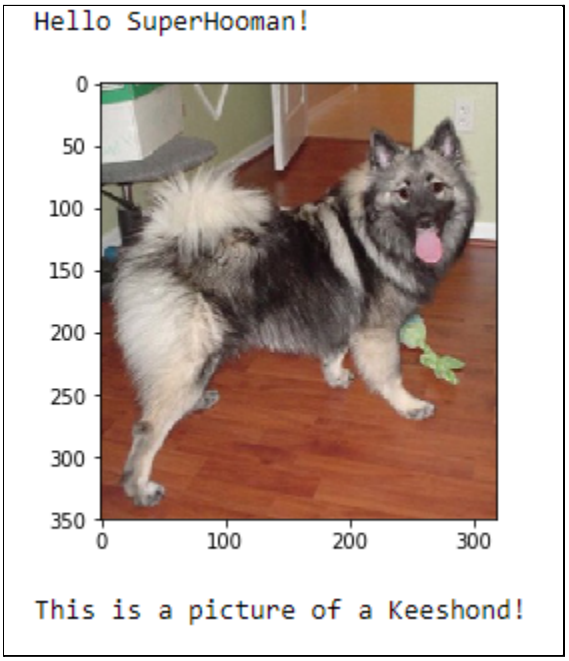
Justification

The second model, with Transfer Learning, overperforms the benchmark model by far. Although there is still room for improvement for the benchmark model, the transfer learning accuracy and speed are unbeatable.

V. Conclusion

Free-Form Visualization

Below are some of the results from the final algorithm:



Reflection

There are seven steps to complete this project:
Step 1: Import datasets and data exploration
Step 2: Detect humans' faces by Haar Cascade classifier, OpenCV
Step 3: Detect dogs pictures by pre-trained ImageNet VGG16 model
Step 4: Create a CNN from scratch to classify dog breeds. Creating image loaders, Net class, loss function, optimizer, train, and test.
Step 5: Create a CNN using Transfer Learning (Pre-trained VGG16) and creating image loaders, Net class, loss function, optimizer, train, and test.
Step 6: Implement the main function to detect the picture as a human or a dog and return the breed of the dog if the picture is a dog or resembling the dog breed if the picture is human.
Step 7: Test the model and the main with random pictures

Improvement

For both models, there is room for improvement:

- Using a more sophisticated pre-trained algorithm, preferably on dog images. e.g., ResNet50
- Hypertuning; find the values for the parameters to get the highest accuracy.
- Tuning the number of epochs, adding more data augmentation, etc.