Capstone Project
Data Science Nanodegree

Victor Wu
Dec 18, 2018

Definition

Project Overview

This project is to develop an algorithm for a Dog Identification App, which accept any image provided as input. The output is to classify dog's breed if a dog is found and provide a most resembling dog breed's estimation if a human is detected.

The algorithm of the classification method is related to deep learning and convolutional neural network.

Problem Statement

There are eight tasks in order to do classification for dog's identification.

1. Import Datasets
2. Detect Humans
3. Detect Dogs
4. Create a CNN to Classify Dog Breeds(from Scratch)
5. Use a CNN to Classify Dog Breeds (using Transfer Learning)
6. Create a CNN to Classify Dog Breeds (using Transfer Learning)
7. Write an Algorithm
8. Test the Algorithm

The final goal is to recognize the breed of a dog if there is a dog image and provide the most similar dog breed if it is a human picture.

Metrics

Accuracy is one of ways to measure how good model is. It includes both true positives and true negatives with equal weights.

$$\text{accuracy} = \frac{\text{true positives} + \text{true negatives}}{\text{dataset size}}$$

Analysis

Data Exploration

There are two image resources to provide most images for this project. One is ImageNet which is a large visual database designed for use in visual object recognition software research[1]. ImageNet is a very large and very popular dataset used for image classification and other vison tasks. Another dataset is Labeled Faces in the Wild that is a database of face photographs designed for studying the problem of unconstrained face recognition and contains more than 13,000 images of faces collected from the web[2].

[1] https://en.wikipedia.org/wiki/ImageNet
[2] http://vis-www.cs.umass.edu/lfw/index.html

Algorithms and Techniques

Harr-Cascade (Harr feature-based cascade classifiers) Detection in Open CV is applicable to detect humans among the images in the project.

In order to classify dog breed, we use Convolutional Neural Net of deep learning. Further more transfer learning is used here to leverage existing CNN methods with bottleneck features. The below is the bottleneck features applied in this project:

1. VGG-19 bottleneck features
2. ResNet-50 bottleneck features

Methodology

Data Preprocessing

The images are prepared in advance with training, validation and test dog images. In total, it is 133 dog categories and 8351 dog images. It is split with 6680 training dog images, 835 validation dog images and 836 test dog images. Regarding human dataset, there are 13233 total human images.

In this project, Keras CNNs need a 4D tensor as input, with shape (nb_samples, rows, columns, channels). We implement path_to_tensor function to take an image path as input and output a 4D array as required.

Implementation

There are seven steps to implement the project:

1. Detect Humans
   Firstly we use OpenCV to grayscale the image, secondly the detectMultiScale function executes the classifier stored in face_cascade and takes the grayscale image as a parameter.
2. Detect Dogs
   With the prepared data which pre-processed by path_to_tensor function, we leverage Keras CNN ResNet-50 API to recognize the dog image as long as the return value is in categories between 151 and 268 of ImageNet 1000 class id to human readable labels.
3. Create a CNN to Classify Dog Breeds(from Scratch)
   We create a CNN from scratch to classify dog breeds in order to attain 1% at least in 5 epochs. The below is model architecture.

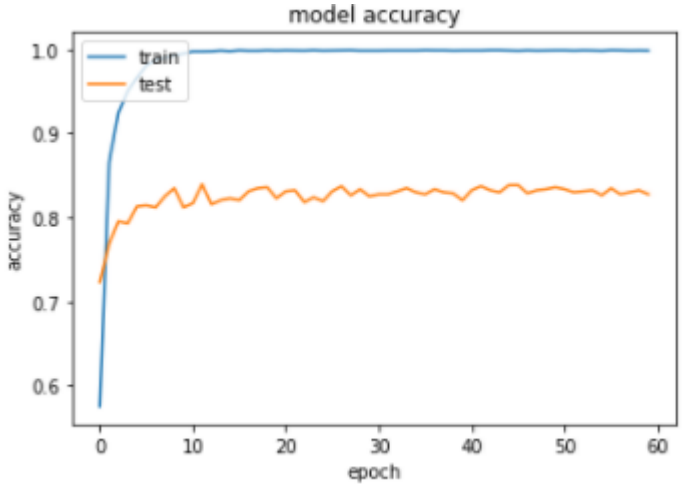
Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 223, 223, 16)	208
max_pooling2d_1 (MaxPooling2	(None, 111, 111, 16)	0
conv2d_2 (Conv2D)	(None, 110, 110, 32)	2080
max_pooling2d_2 (MaxPooling2	(None, 55, 55, 32)	0
conv2d_3 (Conv2D)	(None, 54, 54, 64)	8256
max_pooling2d_3 (MaxPooling2	(None, 27, 27, 64)	0
global_average_pooling2d_1 (	(None, 64)	0
dense_1 (Dense)	(None, 133)	8645

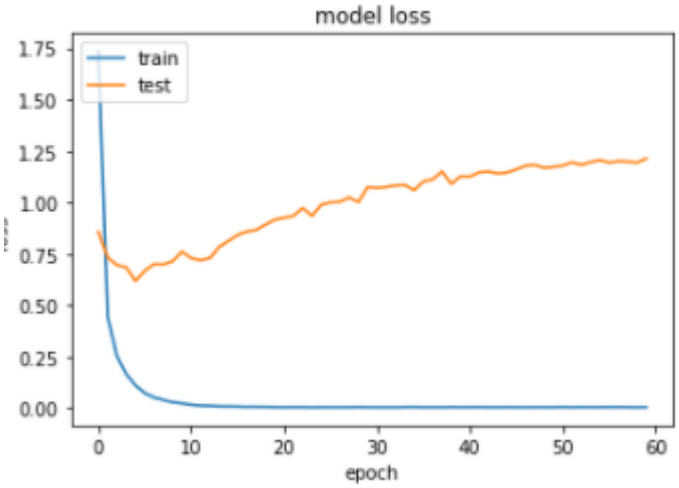
Total params: 19,189.0
Trainable params: 19,189.0
Non-trainable params: 0.0

4. Use a CNN to Classify Dog Breeds (using Transfer Learning)
   We create a CNN with VGG-16 bottleneck features to achieve test accuracy 40% at least.
5. Create a CNN to Classify Dog Breeds (using Transfer Learning)
   In this step, I choose the bottleneck features of ResNet50 to increase the accuracy 60% above.
6. Write an Algorithm
   We use the CNN of step 5 to implement an algorithm in order to classify whether the image is a dog or human, and further identify breed it is or it is most similar with.
7. Test the Algorithm
   The step is to apply the algorithm we achieved in previous step for a test.

Refinement

The refinement is to observe its accuracy and loss when we train and test the model.





Result

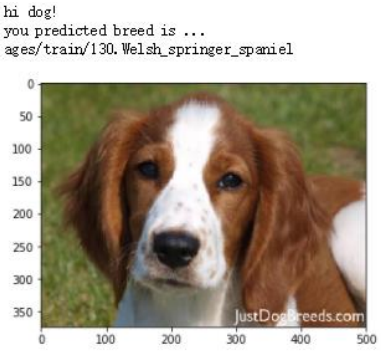
Model Evaluation and Validation

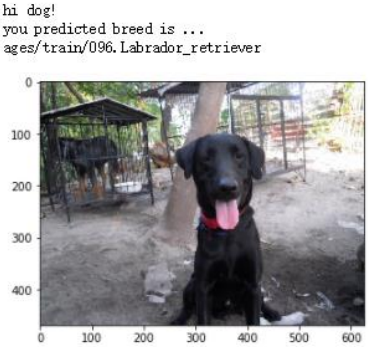
The accuracy of validation test in the model from scratch is 2.2727% that meets the goal (larger than 1%). The validation test accuracy of transfer learning with CNN using VGG16 bottleneck features is 40.4306%. And the accuracy of transfer learning with ResNet50 prediction is 81.1005% that is greater than 60%(the goal required to meet)

Conclusion

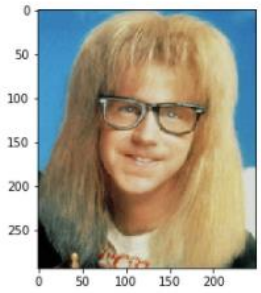
Free-Form Visualization

The visualization is as follows

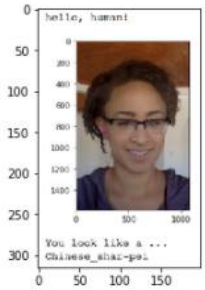




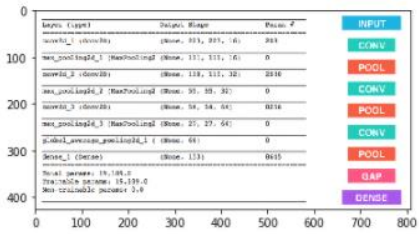
hi human!
You look like a ...
ages/train/002.Afghan_hound



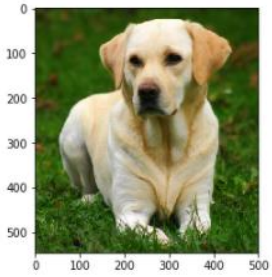
hi human!
You look like a ...
ages/train/064.English_toy_spaniel



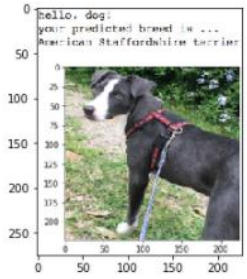
wrong picture...



hi dog!
you predicted breed is ...
ages/train/096.Labrador_retriever

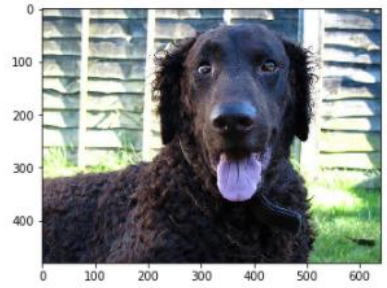


wrong picture...

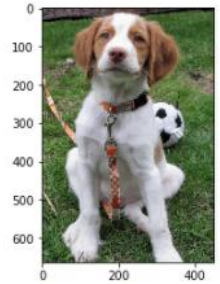


hi dog!
you predicted breed is ...
ages/train/035.Boykin_spaniel

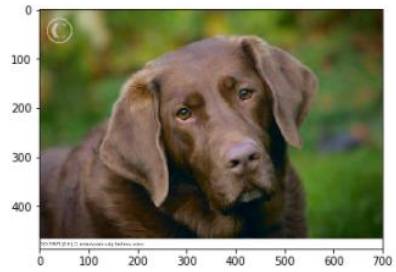
hi dog!
you predicted breed is ...
ages/train/055.Curly-coated_retriever



hi dog!
you predicted breed is ...
ages/train/037.Brittany



```
hi dog!
you predicted breed is ...
ages/train/047.Chesapeake_bay_retriever
```



Improvement

The improvement point is to adapt the shape of input so that the application can recognize the dog and further classify the breed in the below situation.

```
wrong picture...
```