

Dog Breed Classifier

Capstone Proposal

17-10-2020

Ahmed Gharib

Machine Learning Engineer Nanodegree

Udacity

Domain Background

Recognizing dogs' breeds according to specific physical characteristics is a challenging task for humans, since There are several breeds that most people have never seen before, or even heard of. Some are newly registered, and some are just less common in the United States.

To mention Some of these dogs, there are (Beauceron, Canaan Dog, Cesky Terrier, Chinook, Dandie Dinmont Terrier and Lagotto Romagnolo), and many more.

In the United States alone, the AKC's dog breed list currently includes 190 dog breeds. Worldwide, the FCI lists 360 officially recognized breeds. These don't include experimental breeds that have yet to achieve official status. Official lists also don't include mixed-breed dogs, not even "designer" cross breeds like the goldendoodle (a cross between a golden retriever and a poodle) or the puggle (a mix of beagle and pug).

But on the other hand some people in the world have no clue about any of the dog breeds and can't even tell the difference between them.

This may give an indication that the task of dogs breed classification is a complex one, since remembering all dog breeds and distinguishing between similar ones cannot be easily done by humans. From here comes the need for computers to do such a task, machine learning pleasures to assist.

The task includes image recognition and classification as machine learning techniques to aid computer vision. A Convolutional Neural Network (CNN) will be used for the task.


Problem Statement

It's an Image Classification problem where we need to classify images of dogs and define their breeds and if the image is for a human we classify it as a human and give the closest dog breed for this human image.

Datasets and Inputs

Datasets: the datasets are provided by udacity as 2 zipped folders containing RGB images for humans and dogs the size of the images vary and we will have to handle this problem before passing the images to our models.

1. Human Dataset:



LFW folder with total 13233 images of humans distributed in 5749 folders, but there is variation in the number of images for each folder.

Worth to mention that data is not balanced here, there's only one picture for some people, where other people have more than one. Images also have different backgrounds and different angles, but they all have the same size.

2. Dog Dataset:

Distributed in 3 main folders with a total of 8351 images. Each folder contains 133 folders corresponding to dog breeds.

1. Train: 6680 images 80%
2. Valid: 835 images 10%
3. Test: 836 images 10%

Images have different sizes and different backgrounds. Noteworthy that the number of images provided for each breed varies, so the data is not balanced;

Each folder contains 133 folders for each dog breed but also here the images are not equally distributed for each class or breed of dogs.

Solution Statement

We will need here to create 2 models

1. A model to predict if there is a human in the picture.
2. A model to predict the dog breed of a given picture.

And then Write an algorithm that accepts a file path to an image and first determines whether the image contains a human, dog, or neither. Then,

1. if a dog is detected in the image, return the predicted breed.
2. if a human is detected in the image, return the resembling dog breed.
3. if neither is detected in the image, provide output that indicates an error.

In order to implement this multiclass classification, a Convolutional Neural Network (CNN) will be used, which is a deep learning algorithm. The algorithm takes an image as input, assigns learnable weights to different aspects in the image, and the corresponding outputs are either of the options list above.

Benchmark Model

1. A Model created from scratch with a minimum accuracy of 10% that confirms that the model is working better than a random selection as we have 133 classes and a random model will perform an accuracy of less than 1%.
2. A Model created using transfer learning with a minimum accuracy of 60%.

Evaluation Metrics

Since the data is not balanced, accuracy may not be a fair indicator of the model's evaluation. And since it's a multiclass classification problem, the evaluation is going to be based upon multi-class log loss. The log loss is going to take in consideration the uncertainty of the prediction based upon how much it's different from the actual label. However, some would consider accuracy for model's evaluation, where accuracy : comparing the predicted labels for our images with the true labels and compute the percentage of our model accuracy as follow:

Accuracy = (Number of correctly predicted images/ Total Number of images) * 100%.

Project Design

1. Import Datasets

Importing the necessary datasets and libraries needed to be used, which are provided by Udacity. Download the [dog dataset](#). Unzip the folder and place it in this project's home directory, at the location /dogImages.

Download the [human dataset](#). Unzip the folder and place it in the home directory, at location /lfw.

Then perform image preprocessing before we input them to the models.

2. Detect Humans

In this section, we use OpenCV's implementation of [Haar feature-based cascade classifiers](#) to detect human faces in images.

3. Detect Dogs

In this section, we use a pretrained VGG-16 model to detect dogs in images.

4. Create a CNN to Classify Dog Breeds (from Scratch)

Now that we have functions for detecting humans and dogs in images, we need a way to predict breed from images. In this step, we will create a CNN that classifies dog breeds from scratch, then train, validate, and test that model.

5. Create a CNN to Classify Dog Breeds (using Transfer Learning)

We will now use transfer learning to create a CNN that can identify dog breed from images, then train, validate, and test the model. Our CNN must attain at least 60% accuracy on the test set.

6. Implementing the Algorithm to integrate the two detectors.

Implementing an algorithm that accepts a file path to an image and first determines whether the image contains a human, dog, or neither. Then,

- if a **dog** is detected in the image, return the predicted breed.
- if a **human** is detected in the image, return the resembling dog breed.
- if **neither** is detected in the image, provide output that indicates an error.

7. Testing the Algorithm

In this section, we will take the new algorithm for a spin! What kind of dog does the algorithm think that *you* look like? If you have a dog, does it predict your dog's breed accurately? If you have a cat, does it mistakenly think that your cat is a dog?

References

1. [Deep Learning Specialization at Coursera](#)
2. [PyTorch documentation](#)
3. [Intro to Deep Learning with PyTorch Free Course at Udacity](#)
4. [10 dog breeds probably never hear of \(Article\)](#)
5. [How many dog breeds are there \(Article\)](#)



6. [Deep Neural Network for Dog Breeds Identification \(Research Paper\)](#)