Dog Breed Classifier (CNN)

Vindhya Avvari

Udacity - Capstone Project Proposal

Machine Learning Engineer Nano Degree Program

I. DOMAIN BACKGROUND

The convolutional neural network (CNN) is a class of **deep learning neural network**. CNNs represent a huge breakthrough in image recognition. They're most commonly used to analyze visual imagery and are frequently working behind the scenes in image classification. They can be found at the core of everything from Facebook's photo tagging to self-driving cars.

Image classification is the process of taking an **input** (like a picture) and outputting a **class** (like "dog" or "cat") or a **probability** that the input is a particular class ("there's a 90% probability that this input is a dog or cat). Computer Vision researchers have come up with a data-driven approach to design the algorithm to classify images into distinct categories. Instead of trying to specify what every one of the image categories of interest look like directly in code, they provide the computer with many examples of each image class and then develop learning algorithms that look at these examples and learn about the visual appearance of each class. In other words, they first accumulate a training dataset of labeled images, then feed it to the computer in order for it to get familiar with the data.

Given that fact, the complete image classification pipeline can be formalized as follows:

- Our input is a training dataset that consists of *N* images, each labeled with one of *K* different classes.

- Then, we use this training set to train a classifier to learn what every one of the classes looks like.

- In the end, we evaluate the quality of the classifier by asking it to predict labels for a new set of images that it has never seen before. We will then compare the true labels of these images to the ones predicted by the classifier.

Convolutional Neural Networks (CNNs) is the most popular neural network model being used for image classification problem. The big idea behind CNNs is that a local understanding of an image is good enough. The practical benefit is that having fewer parameters greatly improves the time it takes to learn as well as reduces the amount of data required to train the model. Instead of a fully connected network of weights from each pixel, a CNN has just enough weights to look at a small patch of the image. It's like reading a book by using a magnifying glass; eventually, you read the whole page, but you look at only a small patch of the page at any given time.

Consider a 256 x 256 image. CNN can efficiently scan it chunk by chunk — say, a 5 × 5 window. The 5 × 5 window slides along the image (usually left to right, and top to bottom), as shown below. How "quickly" it slides is called its **stride length**. For example, a stride length of 2 means the 5 × 5 sliding window moves by 2 pixels at a time until it spans the entire image.

A **convolution** is a weighted sum of the pixel values of the image, as the window slides across the whole image. Turns out, this convolution process throughout an image with a weight matrix produces another image (of the same size, depending on the convention). Convolving is the process of applying a convolution. Each convolutional layer typically generates many alternate convolutions, so the weight matrix is a tensor of 5 × 5 × n, where n is the number of convolutions.

II. PROBLEM STATEMENT

To experiment, derive and develop an algorithm that could be used as part of a mobile or web app. In this project any user-supplied image acts as an input and is passed to the model. The trained model will then perform the logic to identify the image whether it is dog's image or human's image. If a dog is detected in the image, it will provide an estimate of the dog's breed. If a human is detected, it will provide an estimate of the dog breed that is most resembling.

III. DATASETS AND INPUTS

The project deals with identifying dogs and humans from the input images hence the input should be of dog images and human images. For this project the two input datasets have been provided in the zip format by Udacity. And each zip file corresponds to dog images and human images. In total there are 13233 human images and 8351 dog images.

The folder containing dog images already has 3 sets of images corresponding to test, train and validation sets. This data was collected for 133 different breeds of dogs. On the other hand, there were 5749 folders for each human data comprising multiple pictures of an individual. These folders have been sorted by the person name. The file paths for both the human (LFW) dataset and dog dataset are stored in the numpy arrays named as human_files and dog_files for further use in the project.

IV. SOLUTION STATEMENT

In this project there are two ways that we would implement the solution of predicting the breed from images. One of them is to build a Convolutional Neural Network from scratch and the other is by using transfer learning.

There are multiple steps involved in implementing the CNN from scratch. To find if the picture is human or not we will use the OpenCV model which provides many pre-trained face detectors. Before using any of the face detectors, it is standard procedure to convert the images to grayscale. The *detectMultiScale* function executes the classifier stored in *face_cascade* and takes the grayscale image as a parameter.

As a next step we would detect dogs. And to find if the dog is on a picture we will use a pre-trained VGG-16 model. We will create our CNN model using transfer learning because we need a lot fewer images this way and we still can get great results.

V. BENCHMARK MODEL

For our benchmark model, we will use the Convolutional Neural Networks (CNN) model created from scratch with an accuracy of more than 10%. This should be enough to confirm that our model is working because random guess would be 1 in 133 breeds which are less than 1. So, I will be comparing the 1st model (built from scratch) to validate the performance against the optimized model built using transfer learning.

VI. EVALUATION METRICS

The problem we try to solve is a classification problem. I am planning to evaluate the performance using the test set accuracy score. After training is complete, I would be testing it by loading couple of input images which the model has never seen, never been trained on. Hence, validating the results by comparing it to the actual breed name would give an idea of its performance.

VII. PROJECT DESIGN

I. Import Datasets:

1. Download the dog dataset. Unzip the folder and place it in this project's home directory, at the location /dogImages.

2. Download the human dataset. Unzip the folder and place it in the home directory, at location /lfw.

II. Detect Humans:

1. We will use the OpenCV model to get faces from the image and that will detect whether an image of human's or not. To do this we will implement Haar feature-based cascade classifiers. These are the steps involved :

a) initialize the pre-trained face detector
b) load the image
c) convert image to grayscale
d) find the faces in the image
e) return true if face is recognized otherwise return false

III. Detect Dogs

1. We will use the pre-trained model VGG16 for executing these steps below.

a) define VGG16 model
b) load and pre-process the image
c) input an image to the VGG16 model
d) model return index is from 0 to 999 (dog indices are between 151 and 268 -inclusive)

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

1. We will create CNN from scratch to classify dog Breeds

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

1. The prediction algorithm is designed using Transfer learning which would determine the dog's breed from the input images.

VI. Write your Algorithm

1. Given an image of a dog, your algorithm will identify an estimate of the canine's breed. If supplied an image of a human, the code will identify the resembling dog breed.

VII. Test Your Algorithm

1. Input random images that the model has not seen yet and determine how accurately it is able to predict the results.

References

1. https://towardsdatascience.com/wtf-is-image-classification-8e78a8235acb
2. https://en.wikipedia.org/wiki/Convolutional_neural_network
3. https://www.kaggle.com/c/dog-breed-identification/overview/description
4. https://github.com/udacity/dog-project
5. https://www.kdnuggets.com/2018/04/right-metric-evaluating-machine-learning-models-1.html
6. https://www.analyticsvidhya.com/blog/2020/02/learn-image-classification-cnn-convolutional-neural-networks-3-datasets/
7. https://becominghuman.ai/understanding-the-basics-of-cnn-with-image-classification-7f3a9ddea8f9
8. https://towardsdatascience.com/the-4-convolutional-neural-network-models-that-can-classify-your-fashion-images-9fe7f3e5399d
9. https://towardsdatascience.com/wtf-is-image-classification-8e78a8235acb