5/14/2020

Dog Breed Classifier with CNN

Capstone Project

Joel Dsouza

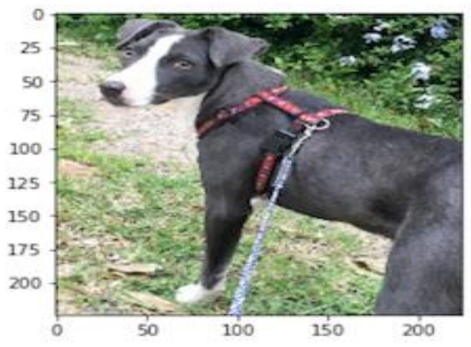
Project Overview

The goal of the project is to build a machine learning model that can be used within web app to process real-world, user-supplied images.

The algorithm has to perform two tasks:

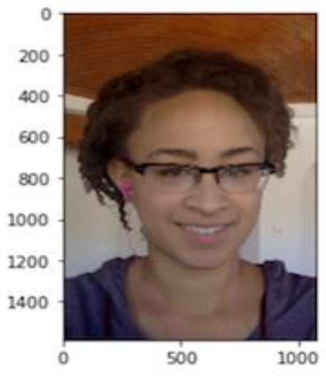
1. Given an image of a dog, the algorithm will identify an estimate of the canine's breed.

```
hello, dog!
your predicted breed is ...
American Staffordshire terrier
```



2. If supplied an image of a human, The code will identify that it is a human and find the corresponding dog breed that the human is associated with

```
hello, human!
```



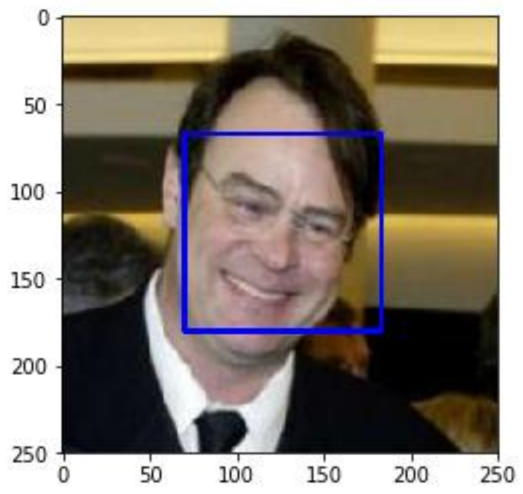
```
You look like a ...
Chinese_shar-pei
```

Capstone Project Joel Dsouza

Machine Learning Engineer Nanodegree

Convolutional Neural Networks are used to classify the breeds of dogs and humans. The solution involves three steps:

1. To detect human images, we can use existing algorithm like OpenCV's implementation of Haar feature based cascade classifiers.
2. To detect dog-images we will use a pretrained VGG16 model.
3. After the image is identified as dog/human, we can pass this image to an CNN model which will process the image and predict the breed that matches the best out of 133 breeds.



Dataset Import

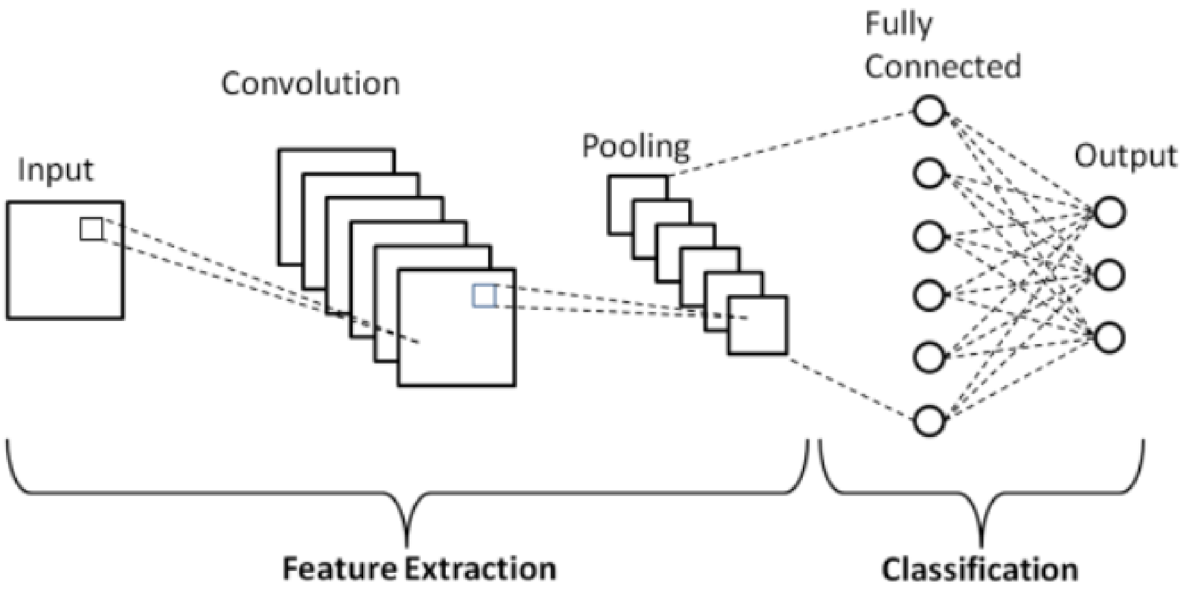
- Download the dog dataset here : https://s3-us-west-1.amazonaws.com/udacity-aind/dog-project/dogImages.zip

- Download the human dataset here : https://s3-us-west-1.amazonaws.com/udacity-aind/dog-project/lfw.zip

Problem Statement

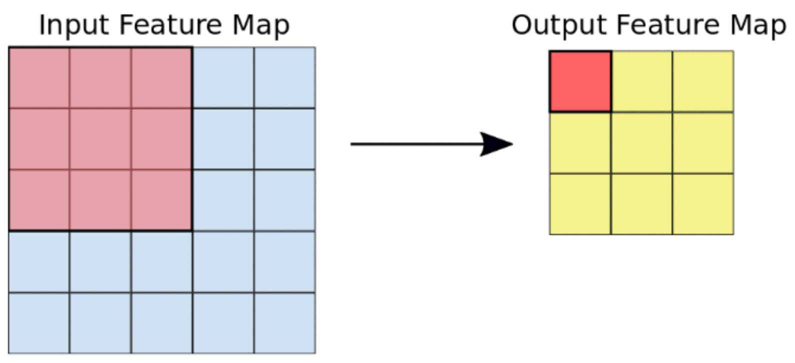
The goal of this project is to create an app that can classify the images of dogs and predict their corresponding breeds. This can be particularly helpful in identifying stray dogs and helping them find foster homes.

Algorithms and Techniques



- What are convolutions in the context of images?

  An image can be expressed as a matrix of RGB values. To complete the convolution operation, we need an image and a filter. the convolution involves superimposing the filter onto the image matrix, adding the product of the values from the filter and the values from the image matrix, which will generate a convoluted layer.



Convolution operation

- How are convolutions integrated into a neural network?
    - Convolution layer: Convo layer is sometimes called feature extractor layer because features of the image are get extracted within this layer. First of all, a part of image is connected to Convo layer to perform convolution operation as we saw earlier and calculating the dot product between receptive field and the filter. Result of the operation is single integer of the output volume. Then we slide the filter over the next receptive field of the same input image by a Stride and do the same operation again. We will repeat the same process again and again until we go through the whole image. The output will be the input for the next layer. Convo layer also contains ReLU activation to make all negative value to zero.
    - Fully Connected Layer(FC): Fully connected layer involves weights, biases, and neurons. It connects neurons in one layer to neurons in another layer. It is used to classify images between different category by training.
    - Softmax / Logistic Layer: Softmax or Logistic layer is the last layer of CNN. It resides at the end of FC layer. Logistic is used for binary classification and softmax is for multi-classification.
    - Output Layer: Output layer contains the label which is in the form of one-hot encoded.

- Why are convolutional neural networks good at image problems in general?

Unlike fully connected layers, convolutional layers have a much smaller set of parameters to learn. This is due to:

- parameter sharing
- sparsity of connection

Parameter sharing refers to the fact that one feature detector, such as vertical edges detector, will be useful in many parts of the image. Then, the sparsity of connections refers to the fact that only a few features are related to a certain output value.

Therefore, we can train on smaller datasets and greatly reduce the number of parameters to learn, making CNNs a great tool for computer vision tasks.

- What hyperparameters exist and what do they mean?

The convolutional layer has four hyperparameters:

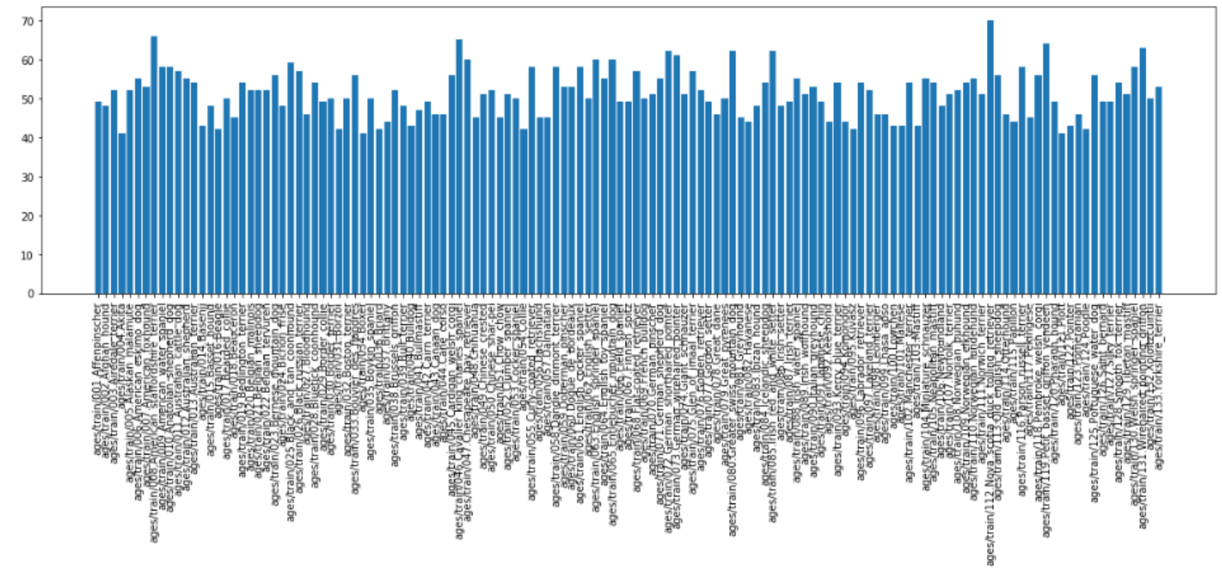
1. The number of filters K
2. **The size F filters:** each filter is of dimensions F×F×D pixels.
3. **The S step** with which you drag the window corresponding to the filter on the image. For example, a step of 1 means moving the window one pixel at a time.
4. **The Zero-padding P**: add a black contour of P pixels thickness to the input image of the layer. Without this contour, the exit dimensions are smaller. Thus, the more convolutional layers are stacked with P=0, the smaller the input image of the network is. We lose a lot of information quickly, which makes the task of extracting features difficult.

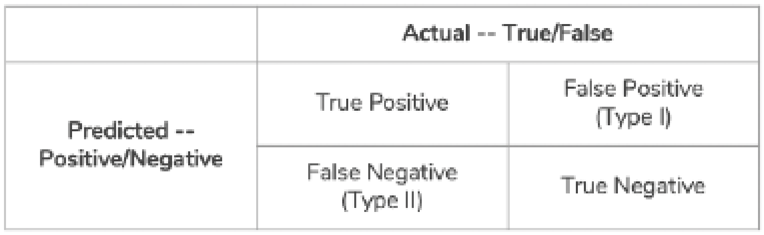
The pooling layer has two hyperparameters:

1. **The size F of the cells:** the image is divided into square cells of size F×F pixels.
2. **The S step:** cells are separated from each other by S pixels.

Metrics

We checked for class imbalance and notice all classes are more or less equally balanced. Hence we use Accuracy and Log Loss as the two evaluation metrics.





	Actual -- True/False	
Predicted -- Positive/Negative	True Positive	False Positive (Type I)
	False Negative (Type II)	True Negative

Accuracy *= Number of items correctly classified/ All classified items*
Accuracy *= TP+TN/Total*

Data Exploration

For this project, the input format must be of image type, because we want to input an image and identify the breed of the dog. The dataset has pictures of dogs and humans.

- ***Dog images dataset:*** The dog image dataset has 8351 total images which are sorted into train (6,680 Images), test (836 Images) and valid (835 Images) directories. Each of this directory (train, test, valid) have 133 folders corresponding to dog breeds. The images are of different sizes and different backgrounds, some images are not full-sized. The data is not balanced because the number of images provided for each breed varies. Few have 4 images while some have 8 images.
- ***Human images dataset:*** The human dataset contains 13233 total human images which are sorted by names of human (5750 folders). All images are of size 250x250. Images have different background and different angles. The data is not balanced because we have 1 image for some people and many images for some.

Benchmark

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% if we don't consider unbalanced data for our dog images. We then compare the accuracy of our transferred learning model and see if the accuracy beats the benchmark score or not.

Refinement

The CNN created from scratch have accuracy of 13%, Though it meets the benchmarking, the model can be significantly improved by using transfer learning. To create CNN with transfer learning, I have selected the Resnet architecture which is pre-trained on ImageNet dataset. The last convolutional output of Resnet is fed as input to our model. We only need to add a fully connected layer to produce 133-dimensional output (one for each dog category). The model performed extremely well when compared to CNN from scratch. **The model accuracy improved to 67%**

Model Evaluation and Validation

Human Face detector: The human face detector function was created using OpenCV's implementation of Haar feature based cascade classifiers. 98% of human faces were detected in first 100 images of human face dataset and 17% of human faces detected in first 100 images of dog dataset.

Dog Face detector: The dog detector function was created using pre-trained VGG16 model. 100% of dog faces were detected in first 100 images of dog dataset and 1% of dog faces detected in first 100 images of human dataset.

CNN using transfer learning: The CNN model created using transfer learning with ResNet101 architecture was trained for 5 epochs, and the final model produced an accuracy of 81% on test data. The model correctly predicted breeds for 680 images out of 836 total images.

Accuracy on test data: 81% (680/836)

Justification

I think the model performance is better than expected. The model created using transfer learning have an accuracy of 67% compared to the CNN model created from scratch which had only 13% accuracy.

Improvement

The model can be improved by adding more training and test data, currently the model is created using only 133 breeds of dog. Also, by performing more image augmentation, we can avoid overfitting and improve the accuracy. I have tried only with ResNet architecture for feature extraction, May be the model can be improved using different architecture.

References

1. https://towardsdatascience.com/introduction-to-resnets-c0a830a288a4
2. https://towardsdatascience.com/an-overview-of-resnet-and-its-variants-5281e2f56035
3. https://towardsdatascience.com/implementing-a-resnet-model-from-scratch-971be7193718
4. https://towardsdatascience.com/understanding-and-visualizing-resnets-442284831be8
5. https://towardsdatascience.com/build-your-first-cnn-fb3aaad77038
6. https://towardsdatascience.com/convolutional-neural-networks-from-the-ground-up-c67bb41454e1
7. https://towardsdatascience.com/a-guide-to-convolutional-neural-networks-from-scratch-f1e3bfc3e2de
8. https://towardsdatascience.com/training-a-convolutional-neural-network-from-scratch-2235c2a25754