Dog-breed Classifier

Simon Andres Soto Ochoa

June 15, 2021

Abstract

This is the Capstone Project for the end of the Machine Learning Engineer Nanodegree from Udacity. The prject consisted in the creation from scratch of a dog breed classifier, to observe the complexity of the topic and also the use of a pre trained oject.

1 Introduction

Computer vision is a fascinating area of Computer Science as it helps humans to do jobs at an incredible fast pace. This project in particular is going to work with classification. This is a task that a human can perform but at a slower pace and constraints. In particular, classification of a dog or not, or human or not is not a complicated task for humans but it is more complicated for computers as they only perceive a collection of data specifying the RGB encoding of each pixel. Even though computers have that limitations, there are available several classifiers online to do that in a very accurate way. The project goal is to create the next step, a dog-breed classifier. This is not a trivial task, as there are many several breed that look quite alike. For a human it requires a lot of time just to memorize the name of the breed, something that a computer can do better than us, but still, after learning names, the process of choosing correctly a specific breed is another story. This has been a topic in research, for example [LKJB12] , tries to overcome difficulties by identifying more features for a breed in the same pictures. Extracting more features from the image has shown results in the case of flowers [NZ08]. There are attempts only in the area of feature extraction to find missing dogs [LTY19].

Besides this topic there are more interesting projects to conduct, but as this project goes well, will give a model from which I can base other personal projects.

In this capstone project, you will leverage what you've learned throughout the Nanodegree program to solve a problem of your choice by applying machine learning algorithms and techniques. You will first define the problem you want to solve and investigate potential solutions and performance metrics. Next, you will analyze the problem through visualizations and data exploration to have a better understanding of what algorithms and features are appropriate for solving it. You will then implement your algorithms and metrics of choice, documenting the preprocessing, refinement, and postprocessing steps along the way. Afterward, you will collect results about the performance of the models used, visualize significant quantities, and validate/justify these

values. Finally, you will construct conclusions about your results, and discuss whether your implementation adequately solves the problem.

define Analyse implement results conclusions

2 Problem Definition

Classification of dog breeds is a complex topic even for humans. Therefore, we are going to construct an algorithm that can help in that process. Given a picture, determine if the picture contains a human of a dog. In the later case, determine a dog breed, and in the other case, determine the breed that the human resembles.

Restrictions: Picture must have ONLY a dog or a human. First part consists in identifying if there is a dog or a human, and the second part is to return the most likely breed or the breed that the human resemble the most. If both or none of the previous ones is found in the picture, we return an error, and ask for a new image.

Ideas of Solution: Use *Transfer Learning* from a Convolutional Neural Network (CNN) to be able to construct (not from scracht) without training our own architecture a dog classifier. We are going to define the architecture of the network but we will use a pre-trained model.

3 Problem Analysis

The data sets are publicly available. Here you can download the .zip file for the dogs pictures and here for the .zip file containing the pictures of humans.

The dog folder contains the train, validation, and test sets already in different folders in which each folder has a respective dog breed. This follows the regular standards on image classification in which each example has to be well referenced. Similar for humans, but in this case, we have just a folder named after one person and inside at least one picture of him/her. There are 8351 images of dogs and 13233 images of humans.

In our case, we have 133 different breeds (The number is the same for train, validation and test) a pre-trained model (VGG-16) for dog classification for the competition of ImageNet, and a model to identify human faces in pictures from OpenCV.

Basic figures over the amount of dogs pictures for each breed can be seen in 1.

As we saw in Section 3, for each breed, we have pictures in the train, validation, and test sets. The classes are no balanced but also not completely imbalance as it can be appreciated in figure 1. There is a difference among the top 4 breeds with more images and the bottom 4, but this does not make up for a significant difference. We decided during the process of executing the project to focus on accuracy instead of recall (as we stated in the proposal).

3.1 First Steps

3.1.1 Human Face Classification

During the start of the process, we use the OpenCV Haarcascade Frontal Face Classifier. With this one we can achieve a 99% accuracy with the first 100 pictures of faces and

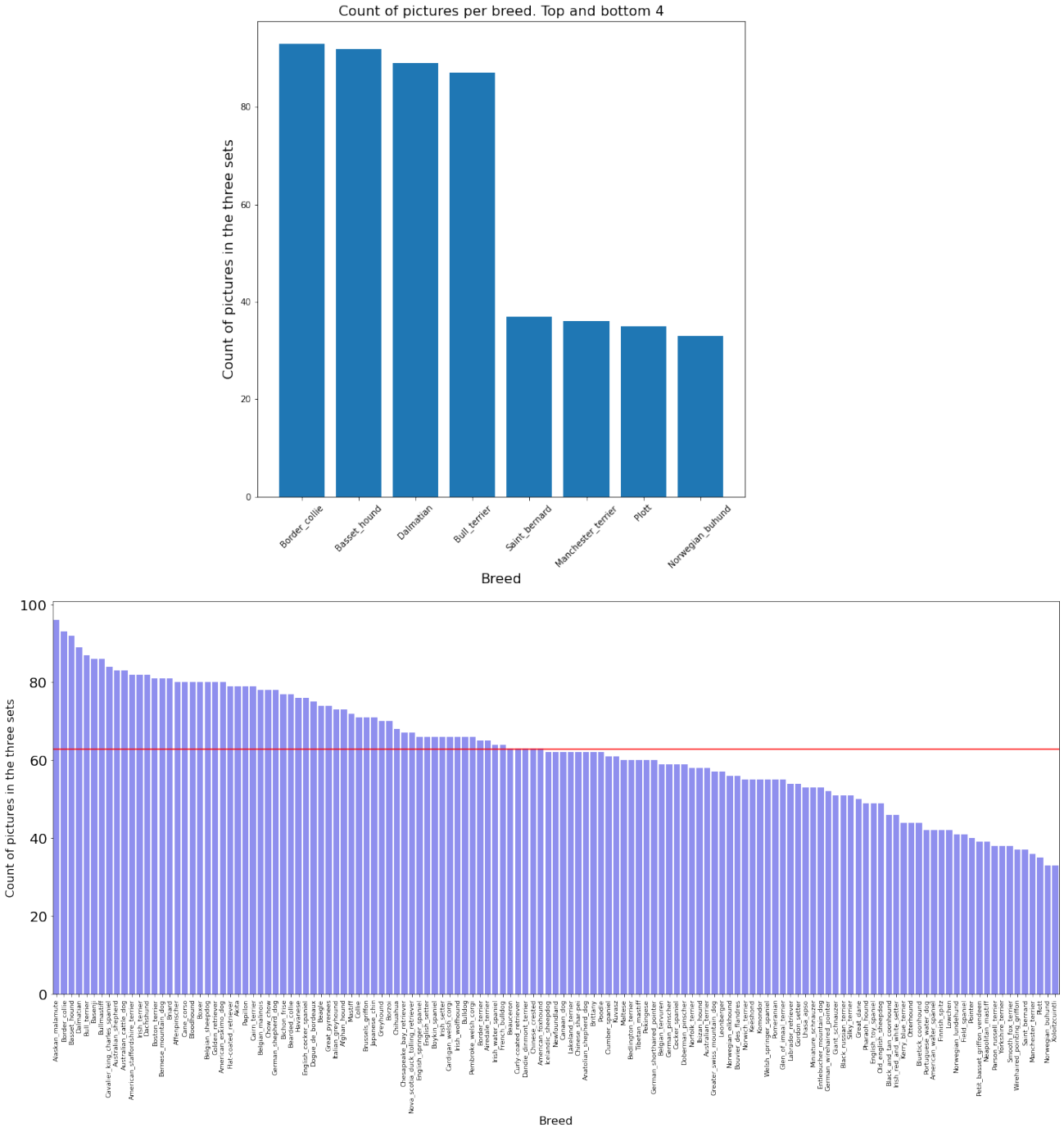

Figure 1: Number of pictures per breed adding up the three sets.

only mistakenly corrected 17% of the dog pictures. So, it is good if we have a human face, but not the best for a dog face.

3.1.2 Dog Classification

For the dog pictures, we have first a pre processing of them in which we flipped horizontally, rotated , and resized. This is our stressed scenario and we got a 74% accuracy with VGG16 for dog classification, and only 1% of humans misclassified as dogs. Without stress, the accuracy can jump up to 99%. We decided later to try a ResNet 50 pre trained model. With this one, under the stressed conditions, we were able to achieve a 86% of accuracy. The last let us conclude that for the second part of the project, this model would perform quite well.

4 Implementation

For the purpose of this project, we work with convolutional neural networks as they are the ones referred in the relevant papers in the area. The best models so far make use of these. In our case, we have convolutional neural networks with padding and a small filter (3×3) followed by a pool layer to group pixels together. A good reference to understand these two is given in the link of the Course on Convolutional Neural Networks for Computer Vision Lastly, we have ReLU as an activation function for every layer. To perform the classification part, we went to the basic two layer fully connected Neural Network.

Regarding the size of the filters, I can only mentioned that in the paper of the VGG-16 model [SZ14], the recommendation is to use 3×3 as the size of the filter. This, they say, is to identify more details on each of the layers. With respect with the final architecture that we have, it was more a trial and error procedure instead of a more scientific approach. The architecture has the following elements:

- (conv1): Conv2d(3, 16, kernel size=(3, 3), stride=(1, 1), padding=(1, 1))
- (conv2): Conv2d(16, 32, kernel size=(3, 3), stride=(1, 1), padding=(1, 1))
- (conv3): Conv2d(32, 64, kernel size=(3, 3), stride=(1, 1), padding=(1, 1))
- (conv4): Conv2d(64, 128, kernel size=(3, 3), stride=(1, 1), padding=(1, 1))
- (conv5): Conv2d(128, 256, kernel size=(3, 3), stride=(1, 1), padding=(1, 1))
- (fc1): Linear(in features=12544, out features=512, bias=True)
- (fc2): Linear(in features=512, out features=133, bias=True)
- (dropout): Dropout(p=0.2)
- (pool): MaxPool2d(kernel size=2, stride=2, padding=0, dilation=1, ceil mode=False)
- (batch norm): BatchNorm1d(512, eps=1e-05, momentum=0.1, affine=True, track running stats=True)

After 30+ epochs, we were able to achieve a 13% of accuracy. This can imply that our architecture needed more training (but that can lead to overfitting) or we needed a better architecture, but that would make it too complex to follow or to explain.

We are not going to check or think about the identification of a human or a dog in a picture. These parts are going to be solved using already existing algorithms (OpenCV and VGG-16).

5 Results

5.1 Benchmark model

The benchmark model is going to be a trained neural network design by us. In this one, we are going to take ideas from different sources, for example [LKJB12], which is one of the first papers that classifies the 133 dog breeds (same number as in this project). In this paper, they use the recall (True Positive Rate) to measure the performance of the model. In our case, we decided to change this measure as it was already implemented and recommended in the notebook. In this case, we use the model that we created as a benchmark for our results. This is clearly a way to also show that there is better tools outside in the market than the ones you can do with simple elements, but still that you are able to construct a Network with the pretty decent results.

We decided based on the results that as the inception network is good at identifying not dogs (the errors in the short list was 0%) we first are going to check if there is a dog in the picture or not. With a negative answer we then ask if there is a human face in the picture. Lastly if none of those is satisfied, we get to the conclusion that there is a mistake with the picture.

This project started as a way to learn more about computer vision, in particular, Convolutional Neural Networks. To achieved this goal, we first used known algorithms to have a human and a dog classifier. After that first step, the real work started. We developed a simple Convolutional Neural Network to find the dog breed of different dog pictures, it was trained under the ecosystem in Udacity as the GPU mode help with the speed at training, and we achieve a decent 13% accuracy.

Some results of the app can be check in the Figures 2 and 3.

In the case for humans, identifying humans in pictures was not the most complicated task, with OpenCV was doable, and you can find funny comparisons made by the Network for the different people. You can see a sample of the work in the Figures 4,5.

The results that we got here, show that there is still room for improvement for the scratch model! Despite this, we were able to meet the requirements and the goals that we proposed and were specified in the notebook. The accuracy of the transfer learned model gave us the satisfaction of making an amazing job.

This code can be used in production for web apps or even an app in which the user can enter a picture and get the breed of the dog that they looked at.

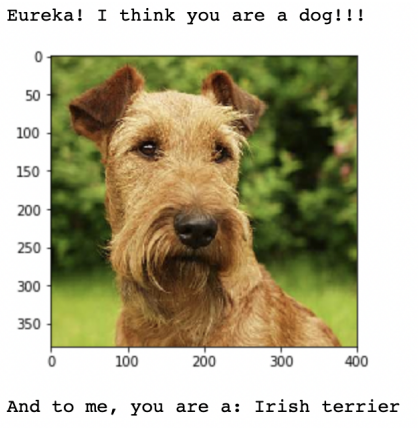

Figure 2: A well identified Irish Terrier

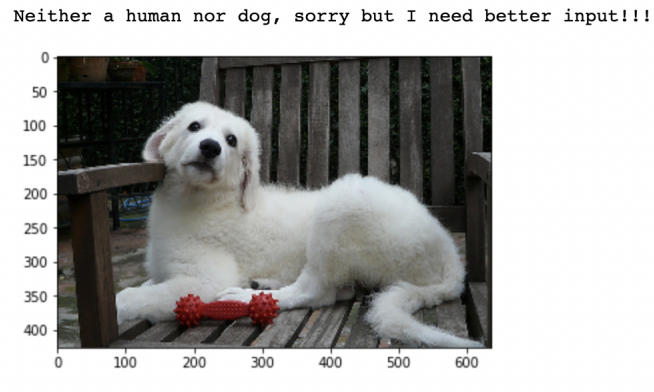

Figure 3: Problems that always appear

6 Conclusions, Ideas and Future work

- Creating a model that satisfy the desired requirements is not an easy task and it requires a lot of time to train and to test the different ideas you might have in your head.
- Differentiating objects is an easy task for the different models that are already produced. In the case of the pre trained models, they were able to differentiate with high accuracy between humans and dogs. Creating a dog breed classifier is a complete different story, there are many small features that cannot been depicted in the pictures. As mentioned in the notebook, even for humans it is a complicated task.
- We can give more training epochs to the algorithm so it might perform better. But there is a chance to over-fitting.
- The state of the art algorithms that are currently available in the market make it easy to construct your own dog breed classifier but their architecture is way more complex. This makes it difficult to follow their ideas. Even more, the complexity also translates into more training time, VGG-16 needed one week to do its training.

Figure 4: Sylvester Stallone as an American water spaniel

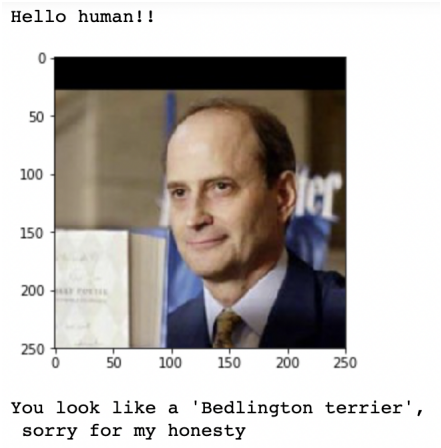

Figure 5: A cute guess. A Bedlington terrier

- Transfer learning is an interesting field that can be explore more in the future, this will let you not to do everything from scratch but to use the already built power, made by people doing research in those areas, for our own purposes. Obviously there should be a domain investigation to use the best models for the tasks that we are interested at the moment.
- The human eyes can construct in the mind 3D models of what we see, and this deepness is something that I cannot imagine that a computer can do.
- Development of an app for smartphones in which you can take the picture and get a guess about the breed of that dog.

References

[LKJB12] Jiongxin Liu, Angjoo Kanazawa, David Jacobs, and Peter Belhumeur. Dog breed classification using part localization. In *European conference on computer vision*, pages 172–185. Springer, 2012.

[LTY19] Kenneth Lai, Xinyuan Tu, and Svetlana Yanushkevich. Dog identification using soft biometrics and neural networks. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019.

[NZ08] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729. IEEE, 2008.

[SZ14] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.