Machine Learning Engineer Nanodegree

Capstone Proposal

Shivaadith Anbarasu

June 21st, 2020

Dog Breed Classifier using CNN

Domain Background

Classification of animals is a famous problem in both the field of zoology/ecology and in computer vision using machine learning. Use of deep learning frameworks and computer vision is becoming increasingly popular in environmental conservation. Similarly, dog breed classification is quite a famous problem especially amongst beginners. The aim is to differentiate between human and dog faces initially, and then classify dogs based on breed – if human faces are detected then we provide a close approximation of a dog breed. Having been personally involved in animal classification projects on Zooniverse, where we manually classify different species across various terrains, this project is of great interest to me. The datasets on Zooniverse are very large and deep learning methods will be well suited for classification problems on their platform. The dog breed classification model is similar to the problem stated above.

Problem Statement

The aim here is to build a model that can be used to classify user input images into various breeds of dogs. Since the user input can contain various types of images, it is important to distinguish between images that contain dogs and those that do not. For this there will be two detectors created, one is a face detector which detects human faces and the other is the dog detector. We also require our model to provide an approximate resemblance of dog breeds even in the case of the input image being a human face. Any image not belonging to either of these groups will output a message mentioning that the image is outside the scope of the application.

After development, the model can be deployed using services from AWS. Web applications can be created and hosted using their services, which makes it accessible to anyone with an internet connection. Mobile applications can also be created to perform the same tasks.

Datasets and Inputs

The data inputs must in the form of images. The datasets used for training our model is obtained from Udacity. The dataset contains images of humans as well as dogs. With 13,233 images of humans and 8351 images of dogs, there seems to be a reasonable amount of variation present in the dataset. This, coupled with appropriate tuning methods, could provide satisfactory results.

The model will also be tested with images outside the provided datasets in order to obtain evaluations metrics.

Solution Statement

Convolutional Neural Networks (CNN) will be used to classify breeds of dogs. A CNN is built from scratch and trained with a certain proportion of images from the given dataset. Transfer learning along with the VGG16 model is then used to obtain satisfactory results.

Initially, OpenCV's Haar cascade classifier will be used to identify human faces. A VGG16 model pretrained on datasets from ImageNet will be used to identify images of dogs. The choices being made here are a result of various other experiments performed on similar datasets. This experience will serve as a personal benchmark using which decisions can be made in case the current model underperforms.

Benchmark Model

The CNN created from scratch serves as the benchmark to this project. The minimum test accuracy required is 10% which provides a fair start line. The model can then be improved upon using various hyperparameter tuning methods. The model obtained through transfer learning requires a test accuracy of 60% minimum. This value can be used to obtain accurate results, which can further be improved using similar hyperparameter tuning methods. We can also use better classifiers for face detection along with a greater diversity of images in training datasets - this would certainly produce much better results.

Evaluation Metrics

The logarithmic loss method can be applied here to evaluate the model. Accuracy score will not provide the best results due to uncertainty of predictions and imbalance in the dataset - both of which log loss method is well suited for. This is especially true in the case of multi-class classification problems. Log loss measures the differences between the predictions and the actual labels. The greater the difference, the greater the loss. Our aim is to minimize this loss value and bring it as close to zero as possible. The cross-entropy criterion from PyTorch can provide this evaluation metric.

Project Design

Step 1: Data pre-processing

- Import necessary libraries and download the required datasets.
- Segregate the data sets into appropriate folders and make it easy to access.
- Split data into training, validation and testing sets

Step 2: Human-face detection

- Use OpenCV's implementation of Haar feature-based cascade classifiers to detect human faces in the images.
- Perform an assessment on a section the downloaded datasets to see how many faces the classifier can identify correctly.

Step 3: Dog-face detection

- Download and implement a pre-trained VGG16 model to detect dog faces. (the model is trained on ImageNet)
- As in the previous step, perform an assessment on this classifier as well in order to obtain an idea of how accurate it can be. This can serve as a valuable benchmark in creating our own model from scratch.

Step 4: Creating a CNN from scratch

- Create data loaders for training, validation and testing sets
- Define the basic model architecture. The architecture is influenced by the research paper Very deep convolutional networks for Large-Scale image recognition by Karen Simonyan & Andrew Zisserman.

- Define the loss function and the optimiser to be used. Train, validate and test the model.

Step 5: Create a CNN to classify Dog breeds using Transfer Learning:

- Copy data loaders from the previous step or create new ones with tuned parameters based on the performance observed with the previous model
- Use transfer learning to obtain a pretrained VGG16 model and implement the architecture.
- Define the loss function and the optimiser. Train, validate and test the new model.
- Compare the performance of the two models. There should be a significant improvement in this stage.

Step 6: Write an algorithm to predict breed of dog

- The algorithm uses both the dog detector and the face detector.
- If a dog is identified in the input image, the appropriate breed is returned
- If human is detected, then the closest resembling dog breed is returned.
- If the image contains neither of the above, then an error message is printed out.

References

1) Reference papers: https://arxiv.org/abs/1409.1556v6
2) ImageNet: http://www.image-net.org
3) OpenCV's Haar cascades classifier: https://docs.opencv.org/trunk/db/d28/tutorial_cascade_classifier.html
4) Project repo: https://github.com/udacity/deep-learning-v2-pytorch/tree/master/project-dog-classification
5) Log Loss: http://wiki.fast.ai/index.php/Log_Loss#Multi-class_Classification, https://en.m.wikipedia.org/wiki/Cross_entropy
6) Cross-entropy: http://neuralnetworksanddeeplearning.com/chap3.html
7) Inspiration: https://www.zooniverse.org/projects