UDACITY

< Back to Deep Learning

Dog Breed Classifier

REVIEW

CODE REVIEW

HISTORY

Meets Specifications

Keen Learner,
Congratulations!🎉👏
Excellent work on this project! You made remarkable models and showcased a good understanding of the underlying concepts. Keep working this hard and you will make a great expert in this field. Have fun and enjoy learning with us. Good luck moving forward!

Pro Tips

The following links may be of great interest:

- How to improve my test accuracy using CNN in Tensorflow.
- A Guide to TF Layers: Building a Convolutional Neural Network.
- Recurrent Neural Networks.

Files Submitted

The submission includes all required, complete notebook files.

Step 1: Detect Humans

The submission returns the percentage of the first 100 images in the dog and human face datasets that include a detected, human face.

Step 2: Detect Dogs

Use a pre-trained VGG16 Net to find the predicted class for a given image. Use this to complete a `dog_detector` function below that returns True if a dog is detected in an image (and False if not).

The submission returns the percentage of the first 100 images in the dog and human face datasets that include a detected dog.

Step 3: Create a CNN to Classify Dog Breeds (from Scratch)

Write three separate data loaders for the training, validation, and test datasets of dog images. These images should be pre-processed to be of the correct size.

Answer describes how the images were pre-processed and/or augmented.

The submission specifies a CNN architecture.

Answer describes the reasoning behind the selection of layer types.

Choose appropriate loss and optimization functions for this classification task. Train the model for a number of epochs and save the "best" result.

The trained model attains at least 10% accuracy on the test set.

Step 4: Create a CNN Using Transfer Learning

The submission specifies a model architecture that uses part of a pre-trained model.

The submission details why the chosen architecture is suitable for this classification task.

Excellent work here! This submission answers question 5 completely. Good intuition on transfer learning concerning the similarities of the datasets.👍

Question 5: Outline the steps you took to get to your final CNN architecture and your reasoning at each step. Describe why you think the architecture is suitable for the current problem.

Answer:

1. I first made a research about how to choose the best model for features extraction and I found this paper : https://arxiv.org/abs/1805.08974 Based on that, I decided to ive a try with a ResNet type model as they seems to be among the best to extract features that can be re-used for other classifier.
2. I tried first with a ResNet-14 with one fully connected layer, but it gave me "only" 53% when testing.
3. I tried the, a ResNet-34 that gave me a 57% accuracy on test
4. When I tried ResNet-50, but it was a bit harder for my hardware to handle it, so I moved back to ResNet-34
5. After adding 1 fully connected layer, I achieved to have an accuracy over 60%

The base model (ResNet-34) has already been trained to extract patterns, features from an image and this training has been done on a huge quantity (ImageNet). It also has been designed to be very efficent on training using "tricks" (here in the ResNet architecture, using residual block that shortcuts some layers and helps each layer to contribute to the output). And that allows such dense model to train well and give great performance. Then we can re-use the model as a features extractor using the convolution layers, and simply adapt and train the fully connected layers on our problem. I think that this architecture is suitable because the dog images dataset is not too different from the ImageNet used for the pre-training of the model. Also, our dataset is quiet small, so using the pretrained model will simplify the problem. This will allow the model to rearrange the way that the features extracted are used and then focus on training the model extracting the maximum value from our small dataset.

Train your model for a number of epochs and save the result wth the lowest validation loss.

Accuracy on the test set is 60% or greater.

The submission includes a function that takes a file path to an image as input and returns the dog breed that is predicted by the CNN.

Step 5: Write Your Algorithm

The submission uses the CNN from the previous step to detect dog breed. The submission has different output for each detected image type (dog, human, other) and provides either predicted actual (or resembling) dog breed.

Step 6: Test Your Algorithm

The submission tests at least 6 images, including at least two human and two dog images.

Splendid work testing your algorithm on several images of dogs, humans and other objects(fowl, mango) getting excellent dog breed classifications for them. You did well on detecting and classifying dogs from humans! Excellent results!👏

Submission provides at least three possible points of improvement for the classification algorithm.

Good job with the interesting suggestions for possible improvement to the project.

Question 6: Is the output better than you expected :) ? Or worse :(? Provide at least three possible points of improvement for your algorithm.

Answer: (Three possible points for improvement)

On the dog breed prediction, I found the output quiet good. On the "dog breed app" itself, I found several issues that I've tried to assess.

1. If a picture includes more than one person, the dog prediction is not very interesting. So I've tried to assess that using the "face detector" to cut into the image and predict a dog breed for each face found.
2. Some dogs are seen as human, it's quiet often (18%). So I've tried to assess that with a double check that it's not a dog when a human face is found. I'm using an optionnal part I've done re-using an exercice from the courses that detect if it's a dog or a cat. I've changed it to detect if it's a dog or a human.
3. If several dogs are on the image, it gives an average dog breed. An improvement would be to detect where is the dog or dogs in an image and crop that part.
4. My knowledge about dog breeds was very low before this project, so I'm quiet impressed of the result of the breed classification, but not very sensitive of the result on human faces as I don't understand how true or flase or funny it is. Maybe that giving 1 or 2 pictures of the dog breed beside the human face detected could be an improvement.
5. I thought that adding a "style transfer" part to generate a human face content with a "dog breed style" could be funny. That would make the app a lot heavier and slow, but more impressive.
6. I think I will build an app that detect what is someone totem animal. But that's not really an improvement, more another project.

Remark

I encourage you to go ahead and build the app as you desire, this will give you an opportunity to learn even more and perfect your skills. Good luck!

DOWNLOAD PROJECT

RETURN TO PATH