UDACITY

‹ Back to Deep Learning

Dog Breed Classifier

REVIEW

HISTORY

Meets Specifications

Congratulations passing the project. I hope you learned a lot and enjoyed it. Keep the good work and happy learning.

Files Submitted

The submission includes all required files.

Step 1: Detect Humans

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

Great job.

The submission opines whether Haar cascades for face detection are an appropriate technique for human detection.

Great answer and discussion! 👏

Additional references

- YOLO
- ArcFace
- Reddit - [D] What is the current state of art in face recognition?

Step 2: Detect Dogs

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

Nice.

Note that this part of the code is very similar to the one to detect human faces. Because of that, you could have created a function to do this.

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

The submission specifies a CNN architecture.

Awesome architecture. Very nice use of Dropout.

Suggestion

- Another thing you could try is Leaky ReLU activation

The submission specifies the number of epochs used to train the algorithm.

epochs = 5 ✌️

The trained model attains at least 1% accuracy on the test set.

Test accuracy: 6.3397% 👍

Step 5: Create a CNN to Classify Dog Breeds

The submission downloads the bottleneck features corresponding to one of the Keras pre-trained models (VGG-19, ResNet-50, Inception, or Xception).

The submission specifies a model architecture.

Nice use of Dropout here too. Also, Inception is one of the best models for this problem.

The submission details why the chosen architecture succeeded in the classification task and why earlier attempts were not as successful.

👍 The main reason to get a great accuracy here is transfer learning 😃

The submission compiles the architecture by specifying the loss function and optimizer.

The submission uses model checkpointing to train the model and saves the model weights with the best validation loss.

The submission loads the model weights that attained the least validation loss.

Accuracy on the test set is 60% or greater.

Test accuracy: 81.6986% ✌️

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 6: Write Your Algorithm

The submission uses the CNN from Step 5 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.

Good job.

Suggestion

- You could show an image of the dog that the human resembles.
- You could also use listdir to iterate through a list of files.

Step 7: Test Your Algorithm

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

Excellent suggestions. About Data augmentation a suggest this paper is a must read, really, it is very interesting.

DOWNLOAD PROJECT

RETURN TO PATH