UDACITY

PROJECT

Dog Breed Classifier

A part of the Deep Learning Nanodegree Program

PROJECT REVIEW

CODE REVIEW

NOTES

SHARE YOUR ACCOMPLISHMENT!

Requires Changes

1 SPECIFICATION REQUIRES CHANGES

😃 Awesome work. You have acquired all the important concepts from this project. You only need to make one modification and then you are ready to go. Wish you all the best for the upcoming projects!

Files Submitted

The submission includes all required files.

😃 Good job including all the 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.

Very good, you rightly detect the percentage of human faces in the human face and dog dataset.

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

Good rationale. You are correct we can use deep learning methods for face detection; data augmentation too will help. There are certain major issues with Haar Cascade face detection method, you can learn more about them here:

- Object Detection : Face Detection using Haar Cascade Classfiers
- Object detection using Haar-cascade Classifier

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.

😃 Very cool, detecting 100% images in dog dataset as dog

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

The submission specifies a CNN architecture.

Simple yet efficient CNN architecture. Great work in explaining the architecture layer wise. You can understand more about the different layers of CNN from this link

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

Good work

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

19.4976% accuracy with the scratch CNN. Good. The rubric required at least 1% but your result outperforms. Great job!

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).

Good work extracting the bottleneck features.

The submission specifies a model architecture.

Good job

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

You have chosen a good architecture. However, the project guidelines require you to answer the following question as well:

> **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.

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

Good choice choosing categorical cross entropy loss and RMSprop optimizer. You can learn more about how the choice of optimizers affect the performance through this blog post: An Overview of gradient descent optimizers

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

👍 good work in saving the model weights with the best validation loss.

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

Good job

Accuracy on the test set is 60% or greater.

Great in getting 81.6986% accuracy

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.

Nice work with implementing the function.

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.

It is good that you put dog detector before human face detector, because the accuracy of dog detector is higher. 😃

Step 7: Test Your Algorithm

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

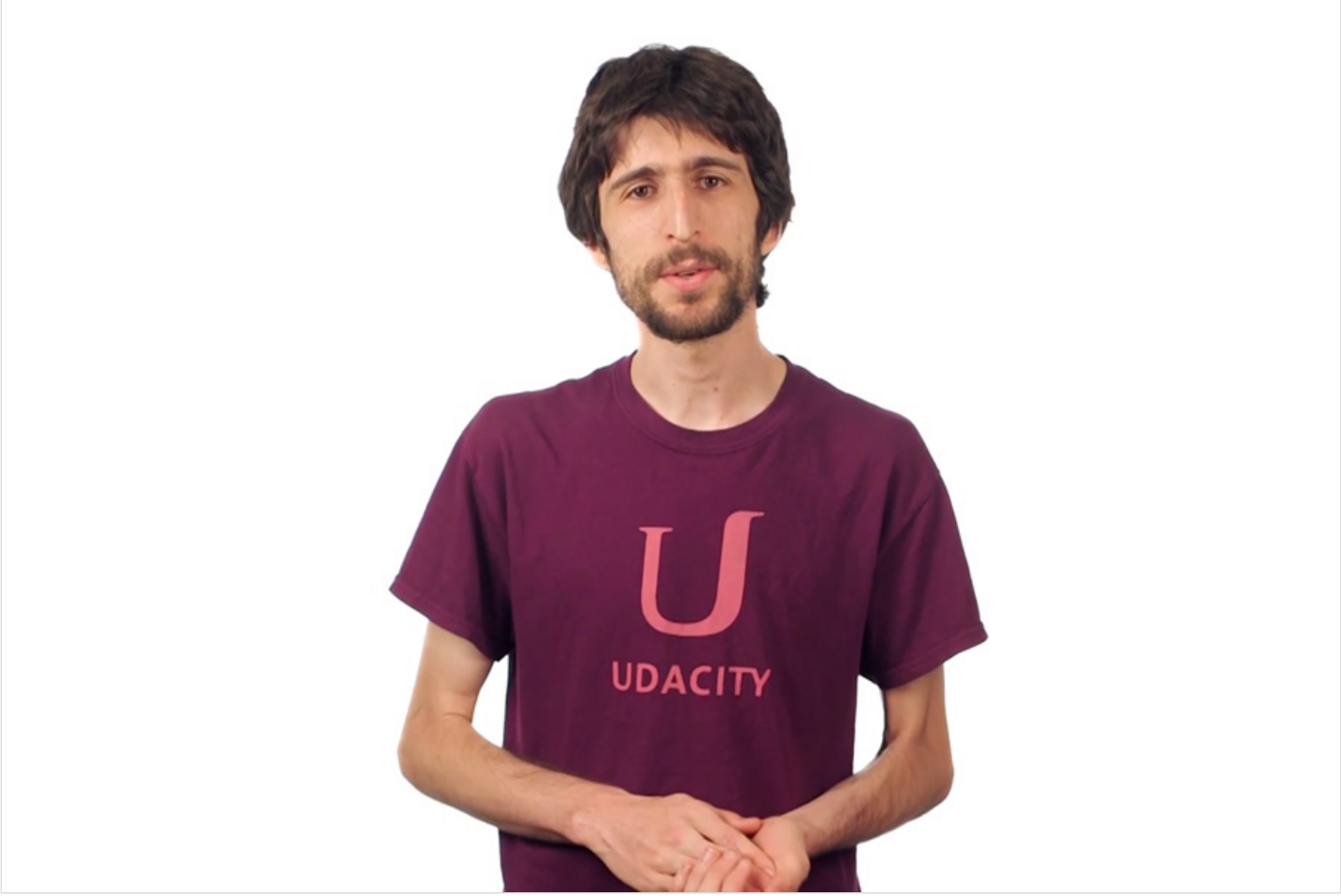
Fabulous job testing your algorithm, and suggesting points for improvement.

Tip

Besides these directions, I would recommend you to look at "how to fine tune a pretrained model" from this Keras post

RESUBMIT

DOWNLOAD PROJECT



Best practices for your project resubmission

Ben shares 5 helpful tips to get you through revising and resubmitting your project.

Watch Video (3:01)

RETURN TO PATH

Rate this review

Student FAQ