UDACITY

‹ Return to Classroom

Dog Breed Classifier

REVIEW

HISTORY

Requires Changes

2 specifications require changes

Hey there, great job working on implementing the Dog Breed Classifier!

There are a couple of sections left where you have probably just missed the question, please make sure to answer them and resubmit. Amazing progress so far, a little touch left and you will be good to go 👍

Have a good one!
Cheers!

Files Submitted

The submission includes all required files.

All files are present!

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.

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

Please make sure to answer the **Question 2**:
"This algorithmic choice necessitates that we communicate to the user that we accept human images only when they provide a clear view of a face (otherwise, we risk having unneccessarily frustrated users!). In your opinion, is this a reasonable expectation to pose on the user? If not, can you think of a way to detect humans in images that does not necessitate an image with a clearly presented face?"

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.

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

The submission specifies a CNN architecture.

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

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

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).

✓ Xception loaded

The submission specifies a model architecture.

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

Please make sure to answer **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.

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.

Amazing results! Pure guessing produces the results of only 0.75%, your model 112+ times more accurate than that with 84% 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.

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.

Great implementation of the pipeline. For further improvement I would encourage you to show the predicted dogs breed photo next to the input image, that way you can see what the algorithm predicts without looking up for the breed separately.

If you would like to improve the face detection part of the algorithm, I would recommend you to check out the following project by Adam Geitgey: https://github.com/ageitgey/face_recognition.

Step 7: Test Your Algorithm

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

Testing ✓

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