UDACITY

Back to Deep Learning Nanodegree

Dog Breed Classifier

审阅

HISTORY

Requires Changes

还需满足 4 个要求 变化

With such low epochs and such high dropout it might just not be enough epochs to get to a higher percentage. Possibly try moving them after the first and second batch normalization but not the last one. Also with such a low starting accuracy it might just be that the architecture needs to be improved before it can handle dropout. It also is unlikely that there is any overfitting at that level.

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.

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

Please add an answer to Question 2. It looks like you missed it.

> 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.

Please add some more details to better answer the question asked. Getting a such a certain test accuracy and even a really low one isn't a general reason why CNN architectures work well for image classification tasks. Please provide descriptions and reasoning through the layers and parameters used in the design or describe why you think that CNN architectures generally should work well for the image classification task.

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

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

Test accuracy: 2.2727%

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).

Or try out all four. Great idea.

The submission specifies a model architecture.

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 add some details to address the second part of the question asked. Why is this problem suitable for using transfer learning on 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.

VGG19 Test accuracy: 55.8612%
ResNet50 Test accuracy: 82.7751%
Inception Test accuracy: 85.6459%
Xception Test accuracy: 81.9378%

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.

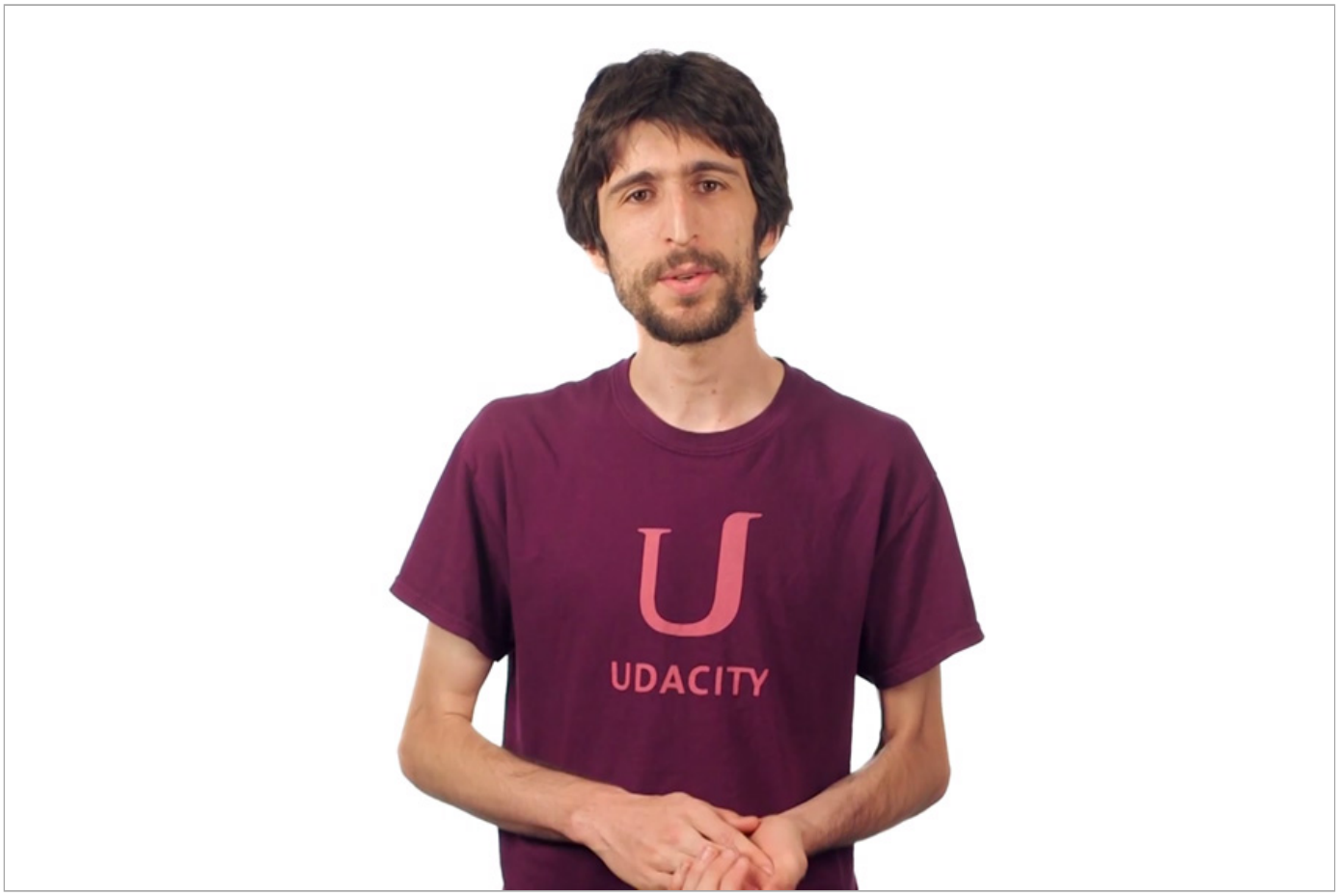
Step 7: Test Your Algorithm

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

The six images should be separate from those included in the training/validation sets.

重新提交

下载项目



重新提交项目的最佳做法

Ben 与你分享修改和重新提交的 5 个有益的小贴士。

观看视频 (3:01)

返回 PATH

学员 FAQ