UDACITY

Return to "Deep Learning" in the classroom

DISCUSS ON STUDENT HUB

Dog Breed Classifier

REVIEW

HISTORY

Meets Specifications

Files Submitted

The submission includes all required files.

all files 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.

Good. The percentage of the first 100 images in human_filenames and in dog_filenames were deteced and correctly returned. Ideally, we would like to have 100% of human images with a detected face and 0% of dog images with a detected face but the algorithm falls short as shown in the answers with 99% of human and 11% of dog images.

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

You are right. To improve the accuracy of face recognition, one can try HOG(Histograms of Oriented Gradients) or some deep learning algorithms, such as YOLO(Real-Time Object Detection algorithm), FaceNet and so on. Besides, one can use imgaug to enhance and expand training sets to increase the diversity of training sets.

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.

Awesome! The percentage of the first 100 images in human_filenames and in dog_filenames with a detected dog were detected and correctly returned. 2% of human_filenames detected a dog and 100% of dog_filenames detected a dog. From observations, we notice that the results get better as the algorithm is modified.

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

The submission specifies a CNN architecture.

good architecture. Here are some articles to build a high performing architectures. May be you can try them in your free time

- This stackexchange question has some ideas on how to improve for architecture for better accuracy.
- Architectures
- Keras Tutorial: The Ultimate Beginner's Guide to Deep Learning in Python.
- Deep learning for complete beginners: convolutional neural networks with keras.
- You can use batch normalisation to further improve the performance of your model. BatchNorm avoids "internal covariate shift" as it minimises the effect of weights and parameters in successive forward and back pass on the initial data normalisation done to make the data comparable across features as explained here
- keras tutorial – build a convolutional neural network in 11 lines.
- Image Classification using Convolutional Neural Networks in Keras

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

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

awesome job with achieving accuracy more than 1%

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).

Using Keras pre-trained model, Inception, the submission downloads the bottleneck features for training, validating and testing.

The submission specifies a model architecture.

Suggestions

Even though the first architecture chosen worked out as expected, it would be good to try out or make other attempts with different architectures and find out the reason they aren't working the way they are supposed to be or why they work better. Try doing this next time.I believe it will enhance your knowledge and skills in making better decisions and architectures in the future.

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

If you havent done please read these articles to get more insight into the different architectures that you can leverage as transfer learning here:

- ImageNet: VGGNet, ResNet, Inception, and Xception with Keras
- ResNet, AlexNet, VGGNet, Inception: Understanding various architectures of Convolutional Networks
- VGG 16 - VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION
- Xception : Deep Learning with depthwise separable Convolutions

Bonus study material

- Le Net 5 - Gradient Based Learning Applied to Document Recognition
- Alex Net - ImageNet Classification with Deep Convolutional Neural Networks
- GoogLeNet - Going Deeper with Convolutions

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

Good job. 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.

Model checkpointing was used to train the model and save the model that attains the best validation loss.

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

The model weights with the least validation loss were loaded as required. Nice.

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.

A function that takes a file path to an image as input and returns the dog breed that was predicted by the CNN has been included and well implemented in the submission.

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.

Tests were made using more than 6 images with at least 2 dogs and 2 humans with a brilliant explanation of what was expected as output and what was gotten as output. Very good.

Step 7: Test Your Algorithm

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

Great points. 

DOWNLOAD PROJECT

RETURN TO PATH