UDACITY

‹ Return to "Deep Learning" in the classroom

DISCUSS ON STUDENT HUB

Dog Breed Classifier

REVIEW

HISTORY

Meets Specifications

Awesome submission ! 🎉 You've done a great job of implementing a Convolutional Neural Network. Your understanding and clarity of concepts of how CNNs work is vividly depicted here.

If you are curious/want to learn more, I recommend you to take a look at the links mentioned below:
CNN's for Visual Recognition
Building an Image Classifier
Large Scale Image Recognition using CNN's
Transfer Learning
CNN Tricks
Keep learning ! ✌🏾

Files Submitted

The submission includes all required, complete notebook files.

All the necessary files are included 👍🏽

Step 1: Detect Humans

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

Step 2: Detect Dogs

Use a pre-trained VGG16 Net to find the predicted class for a given image. Use this to complete a `dog_detector` function below that returns True if a dog is detected in an image (and False if not).

Good work using VGG16 to write a dog detector.

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

Your values for true positives and false positives are acceptable. Superb work in getting the correct percent of dogs and humans!

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

Write three separate data loaders for the training, validation, and test datasets of dog images. These images should be pre-processed to be of the correct size.

Its good that you implemented separate data loader for training, validation, and test datasets. Nice work! You may be able to calculate means dynamically with something like np.mean(train_set.train_data, axis=(0,1,2))/255): https://discuss.pytorch.org/t/normalization-in-the-mnist-example/457/12

Answer describes how the images were pre-processed and/or augmented.

The submission specifies a CNN architecture.

Good job discussing the architecture of your CNN.

Some suggestions for improvement:

- 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
- This stackexchange question has some ideas on how to improve for architecture for better accuracy.

Answer describes the reasoning behind the selection of layer types.

Choose appropriate loss and optimization functions for this classification task. Train the model for a number of epochs and save the "best" result.

Good work on using `crossentropy` loss and `adam` optimizer.
Further reading: Overview of Gradient descent optimizers

The trained model attains at least 10% accuracy on the test set.

Step 4: Create a CNN Using Transfer Learning

The submission specifies a model architecture that uses part of a pre-trained model.

The submission details why the chosen architecture is suitable for this classification task.

Train your model for a number of epochs and save the result wth the lowest validation loss.

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.

Step 5: Write Your Algorithm

The submission uses the CNN from the previous step 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.

Given your output from the initial detection results you might want to swap the logic here, since `dog_detector` had fewer false positives than `face_detector` it should be run first.

Step 6: Test Your Algorithm

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

Good set of sample images, some humans and dogs to test the algorithm.

Submission provides at least three possible points of improvement for the classification algorithm.

Includes good ideas for next direction to take this for improvement.
A resource if you care to read about other general improvements.
https://machinelearningmastery.com/improve-deep-learning-performance/

DOWNLOAD PROJECT

RETURN TO PATH