UDACITY Logout

‹ Return to "Deep Learning" in the classroom DISCUSS ON STUDENT HUB

Dog Breed Classifier

REVIEW | CODE REVIEW | HISTORY

Meets Specifications

Hi

Excellent work!! You passed the Dog Breed Classifier project.
AI is an awesome field
The different areas are:

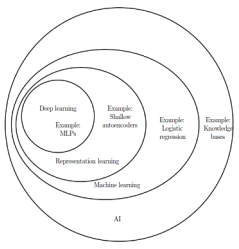

Figure 1.4: A Venn diagram showing how deep learning is a kind of representation learning, which is in turn a kind of machine learning, which is used for many but not all approaches to AI. Each section of the Venn diagram includes an example of an AI technology.

To keep learning more about AI:
9 Deep Learning papers
MIT AGI: Building machines that see, learn, and think like people
YOLO Object Detection
⭐New Deep Reinforcement Learning Nanodegree⭐
The AlphaGo Movie

Keep up the good work!
Sincerely
Leticia

Files Submitted

The submission includes all required, complete notebook files.

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.

Human Face Detector Performance: 99.0%
Dog Face Detector Performance: 13.0%

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).

Great work implementing a dog detector using vgg16.

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

Human dog_detector Performance: 0.0%
Dog dog_detector Performance: 97.0%

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.

You used image augmentation over the training set only. Good definition for the validation and testing set.
You defined three different transformations:

```
train_transforms = transforms.Compose([#transforms.RandomRotation(30),
                                       transforms.RandomResizedCrop(224),
                                       transforms.RandomHorizontalFlip(),
                                       transforms.ToTensor(),
                                       standard_normalization])

valid_transforms = transforms.Compose([transforms.Resize(256),
                                       transforms.CenterCrop(224),
                                       transforms.ToTensor(),
                                       standard_normalization])

test_transforms = transforms.Compose([transforms.Resize(size=(224,224)),
                                      #transforms.CenterCrop(224),
                                      transforms.ToTensor(),
                                     standard_normalization])
```

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

You answered the question 3.

We use Image Augmentation to increase the detection of the model. We can still overfit. Overfit is related to the model depth, and training.

> The goal is to not only reduce overfitting via augmentation but also to augment data in a way such that to best improve the classifier

Related paper: http://cs231n.stanford.edu/reports/2017/pdfs/300.pdf

The submission specifies a CNN architecture.

You created a CNN for dog breed recognition
Using regularization methods, like Dropout, is always helpful when we are training a model that trends to overfit.

Dropout prevents overfitting and provides a way of approximately combining exponentially many different neural network architectures efficiently. The term "dropout" refers to dropping out units (hidden and visible) in a neural network. By dropping a unit out, we mean temporarily removing it from the network, along with all its incoming and outgoing connections.

The choice of which units to drop is random. In the simplest case, each unit is retained with a fixed probability p independent of other units, where p can be chosen using a validation set or can simply be set at 0.5, which seems to be close to optimal for a wide range of networks and tasks. For the input units, however, the optimal probability of retention is usually closer to 1 than to 0.5.

Related paper: Dropout: A Simple Way to Prevent Neural Networks from Overfitting

Answer describes the reasoning behind the selection of layer types.

You answered the question 4

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

You used CrossEntropyLoss loss and SGD Optimizer.
Pytorch provides great LR schedulers that worth trying: https://pytorch.org/docs/stable/optim.html

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

Test Accuracy: 11% (97/836)

Step 4: Create a CNN Using Transfer Learning

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

You used vgg16 pre-trained model. Pytorch provides many pre-trained models. Some work better than vgg16 on this problem. You can try Resnets or desnets models.
More information: https://pytorch.org/docs/stable/torchvision/models.html

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

You answered the question 5.

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.

Test Accuracy: 82% (686/836) excellent

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.

You wrote predict_breed_transfer for meeting specifications.

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.

You wrote the required algorithm. You can just use:
`if dog_detector(img_path) :`

Instead of
`if dog_detector(img_path) is True:`

Step 6: Test Your Algorithm

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

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

You provided excellent improvements in the answer to the question 6.
You can find more ideas: https://machinelearningmastery.com/improve-deep-learning-performance/

DOWNLOAD PROJECT

Rate this review