UDACITY

‹ Return to "Deep Learning" in the classroom

Dog Breed Classifier

REVIEW

CODE REVIEW

HISTORY

Meets Specifications

Keen Learner,
Congratulations!🎉
Excellent work on this project! You have made remarkable models and showcased good knowledge of the concepts covered. We wish that you keep studying this hard so you will be able to do great things with the knowledge gained. Have fun learning and good luck as you move forward.

Pro Tips

The following links may be of interest for further learning.

- How to improve my test accuracy using CNN in Tensorflow.
- How To Improve Deep Learning Performance.
- A Guide to TF Layers: Building a Convolutional Neural Network.
- A Beginner's Guide to Recurrent Networks and LSTMs.
- Recurrent Neural Networks.
- Understanding LSTM Networks.

Files Submitted

The submission includes all required files.

All necessary files are present in this submission. Great!

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.

Superb work in getting the number of images in the first 100 images of dogs and humans faces datasets with detected human faces!

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

Thanks for your opinion here.
Also, there are more complex algorithms like CNNs that can clearly detect humans from images with obscured faces or taken from different angles, producing much better and logical classifications, eliminating the option of requesting clear images from users. Keep up the good work!👏

Pro Tips

Please, you can read more about the Haar cascades algorithm in links below:

- OpenCV Face Detection in Images using Haar Cascades with Face Count.
- Face Detection using Haar Cascades.
- Here is a nice OpenCV documentation that discusses Haar cascades in detail.
- You can check out a brief yet informative explanation of Haar Cascades in 2 minutes in the link provided.

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.

Superb work in getting the number of images in the first 100 images of dogs and humans faces datasets with a detected dog!

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

The submission specifies a CNN architecture.

This CNN network is deep enough to recognize complex patterns in the images in the dataset and consequently producing accurate predictions. You could add dropout layers to reduce overfitting. Nice work using `relu()` (`softmax` for the last layer) activations to introduce non-linearity in the model. Great job!👏

Pro Tips

Please, the following links may be of great interest to you for further research:

- Keras Tutorial: The Ultimate Beginner’s Guide to Deep Learning in Python.
- Deep learning for complete beginners: convolutional neural networks with keras.
- 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 number of epochs was specified to be 6. Training with more epochs will give even better accuracy. Good job!

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

Excellent job!

Admonition

- We could also use keras.callbacks.EarlyStopping to get the best performance of the network.
- I want to share this discussion about choosing the optimal number of epochs as it may help you out in the future.

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).

The submission downloads the bottleneck features and answers question 4 properly.

Pro Tips

- This article provides a clear and in-depth discussion about the different pre-trained networks from Keras. Please check it out if you want to learn more about them.

The submission specifies a model architecture.

The model architecture was specified in the project notebook. Great job using transfer learning and adding a dropout layer to reduce overfitting.

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

Nice!

Pro Tips

Transfer learning is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task
Here are some good documents you can check out for more insight on transfer learning.

- 1n Convolutional Neural Networks for Visual Recognition
- When you should use Transfer Learning
- Transfer Learning Introduction

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

The model was compiled with the `rmsprop` optimizer and `categorical_crossentropy` loss function was specified. 👍

Pro Tips

- Loss functions and optimization algorithms for deep learning models may help you understand further the uses of these functions and algorithms. This is a nice article that would help expand your knowledge about them.

The submission uses model checkpointing to train the model and saves the model weights with the best validation loss.

Model checkpoints were used to train and save the model weights with the validation loss as required!

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

Awesome! Indeed the submission loads the model weights that attained the least validation loss. 👍

Accuracy on the test set is 60% or greater.

Excellent test accuracy! Keep up the great work!👏

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 `dog_breed_classify()` is correctly defined to take a file path as input and return the breed predicted by the CNN. Great job!

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.

The implemented algorithm uses the CNN to detect dog breeds. Furthermore, it has different outputs for each detected image type and provides either predicted actual (or resembling) dog breed. Nicely done!🎉

Step 7: Test Your Algorithm

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

Nice job testing the algorithm on two human and two dog images, getting very good classifications. The predicted dog breeds for the images are excellent. Impressive results!👏
You also did very well by answering question 6 explicitly, making good points for improvement of your algorithm. I encourage you to try out some of the improvement points when you have the chance.

DOWNLOAD PROJECT

RETURN TO PATH

Rate this review