UDACITY

Return to Classroom

Landmark Classification & Tagging for Social Media

REVIEW

HISTORY

Meets Specifications

Congrats

You have successfully passed the project. The set of steps are well coded. Training and testing are parts have been executed rightly. The model is performing pretty well with the test images.

I have shared some extra resources to help you make it better.

Keep Learning! Deep Learning!

Files Submitted

✓

The submission includes the required notebook file and HTML file. When the HTML file is created, all the code cells in the notebook need to have been run so that reviewers can see the final implementation and output.

Step 1: Create a CNN to Classify Landmarks (from Scratch)

✓

✓

The submission randomly splits the images at `landmark_images/train` into train and validation sets. The submission then creates a data loader for the created train set, a data loader for the created validation set, and a data loader for the images at `landmark_images/test`.

Implementation of transforms is acceptable. The SubsetRandomSampler is used rightly to split the training data into train and valid parts.

Set of transforms are also rightly coded to preprocess data. 👌

✓

Answer describes each step of the image preprocessing and augmentation. Augmentation (cropping, rotating, etc.) is not a requirement.

✓

The submission displays at least 5 images from the train data loader, and labels each image with its class name (e.g., "Golden Gate Bridge").

The code to plot images in training data loader is all correct. The purpose of plotting training images was to see the changes in image due to transformations that have been performed while preprocessing.

✓

The submission chooses appropriate loss and optimization functions for this classification task.

The loss function and SGD optimizer are used precisely. ✅

Suggestions (Important)

I encourage you to try using StepLR Scheduler:
https://www.deeplearningwizard.com/deep_learning/boosting_models_pytorch/lr_scheduling/

✓

The submission specifies a CNN architecture.

Perfect

The network is precisely coded. You have used the set of 5 Convolution layers with ReLu and Pooling layers.
The classifier part of the network is coded rightly using 2 Linear layers. Dropout helps to reduce the chances of over-fitting.

✓

Answer describes the reasoning behind the selection of layer types.

✓

The submission implements an algorithm to train a model for a number of epochs and save the "best" result.

✓

The submission implements a custom weight initialization function that modifies all the weights of the model. The submission does not cause the training loss or validation loss to explode to `nan`.

✓

The trained model attains at least 20% accuracy on the test set.

Good Job

The model is well trained and fetching good results with the test data too:

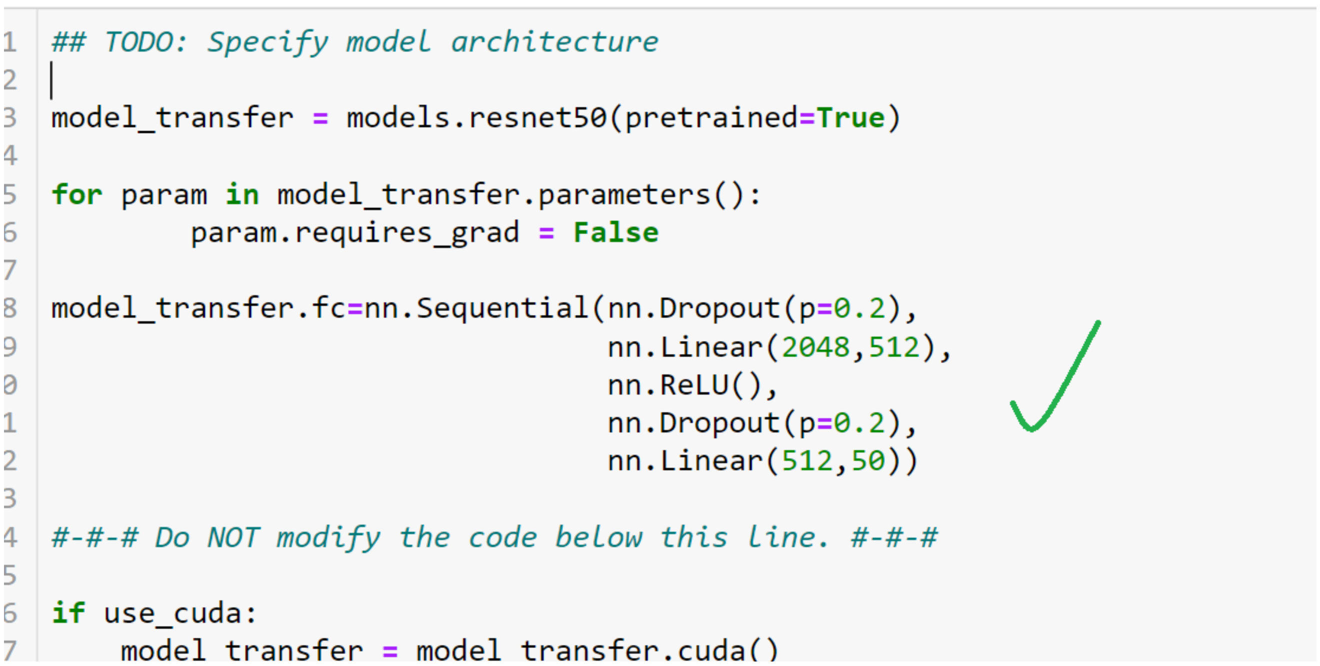
```
Test Accuracy: 34% (435/1250)
```

Step 2: Create a CNN to Classify Landmarks (using Transfer Learning)

✓

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

The ResNet50 is loaded perfectly. You have replaced the final fully connected layer with a new set of layers. This helps to build a customize the model.



```
## TODO: Specify model architecture

model_transfer = models.resnet50(pretrained=True)

for param in model_transfer.parameters():
        param.requires_grad = False

model_transfer.fc=nn.Sequential(nn.Dropout(p=0.2),
                                nn.Linear(2048,512),
                                nn.ReLU(),
                                nn.Dropout(p=0.2),
                                nn.Linear(512,50))

#-#-# Do NOT modify the code below this line. #-#-#

if use_cuda:
    model_transfer = model_transfer.cuda()
```

✓

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

✓

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

✓

Accuracy on the test set is 60% or greater.

Fantastic

The model's performance is good. It's well trained and working perfectly with the testing data.

```
Test Accuracy: 72% (906/1250)
```

Step 3: Write Your Landmark Prediction Algorithm

✓

The submission implements functionality to use the transfer learned CNN from Step 2 to predict top k landmarks. The returned predictions are the names of the landmarks (e.g., "Golden Gate Bridge").

✓

The submission displays a given image and uses the functionality in "Write Your Algorithm, Part 1" to predict the top 3 landmarks.

✓

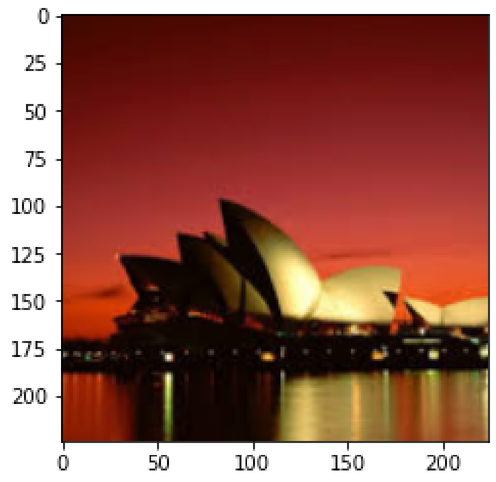
The submission tests at least 4 images.

Superb

It's great that you have tested with a diverse set of images. I liked that you have printed the actual landmark too along with the predicted landmarks. This helps to validate the performance of model.

```
[21]:  suggest_locations('test2.jpg')
```

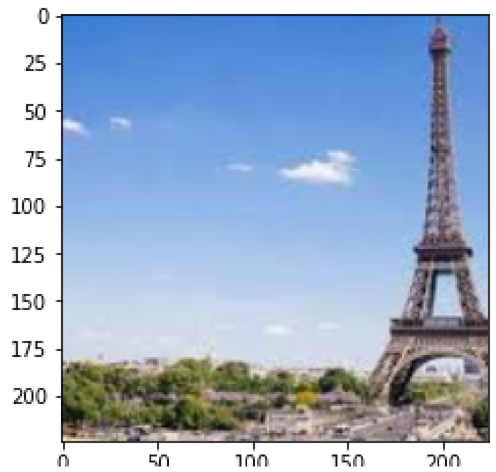
```
Is this picture of the
Sydney Opera House, Forth Bridge, Sydney Harbour Bridge?
```



✓

```
[22]:  suggest_locations('test3.jpg')
```

```
Is this picture of the
Eiffel Tower, Vienna City Hall, Gateway of India?
```



✓

✓

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

Good to see that you are aware of the things that can be worked on.
These are some of the important steps. You should definitely try working on these points.

It would be better if you try using other transfer learning models like ResNet152, VGG16 with batch normalization. These can help you see an improvement in the results.

While considering to train the model for more epochs, it would be better if you use EarlyStopping too.

DOWNLOAD PROJECT

RETURN TO PATH

Rate this review

START