

Files submitted

Criteria	Specification
----------	---------------

Submission Files	The submission includes all required files.
------------------	---

Part 1 - Development Notebook

Criteria	Specification
----------	---------------

Package Imports	All the necessary packages and modules are imported in the first cell of the notebook
-----------------	---

Training data augmentation	torchvision transforms are used to augment the training data with random scaling, rotations, mirroring, and/or cropping
----------------------------	---

Data normalization	The training, validation, and testing data is appropriately cropped and normalized
--------------------	--

Data loading	The data for each set (train, validation, test) is loaded with torchvision's ImageFolder
--------------	--

Data batching	The data for each set is loaded with torchvision's DataLoader
---------------	---

Pretrained Network	A pretrained network such as VGG16 is loaded from torchvision.models and the parameters are frozen
--------------------	--

Feedforward Classifier	A new feedforward network is defined for use as a classifier using the features as input
------------------------	--

Training the network	The parameters of the feedforward classifier are appropriately trained, while the parameters of the feature network are left static
----------------------	---

Validation Loss and Accuracy	During training, the validation loss and accuracy are displayed
------------------------------	---

Testing Accuracy	The network's accuracy is measured on the test data
------------------	---

Saving the model	The trained model is saved as a checkpoint along with associated hyperparameters and the <code>class_to_idx</code> dictionary
------------------	---

Loading checkpoints	There is a function that successfully loads a checkpoint and rebuilds the model
---------------------	---

Image Processing	The <code>process_image</code> function successfully converts a PIL image into an object that can be used as input to a trained model
------------------	---

Class Prediction	The <code>predict</code> function successfully takes the path to an image and a checkpoint, then returns the top K most probable classes for that image
------------------	---

Sanity Checking with matplotlib	A matplotlib figure is created displaying an image and its associated top 5 most probable classes with actual flower names
---------------------------------	--

Part 2 - Command Line Application

Criteria	Specification
Training a network	<code>train.py</code> successfully trains a new network on a dataset of images
Training validation log	The training loss, validation loss, and validation accuracy are printed out as a network trains
Model architecture	The training script allows users to choose from at least two different architectures available from torchvision.models
Model hyperparameters	The training script allows users to set hyperparameters for learning rate, number of hidden units, and training epochs
Training with GPU	The training script allows users to choose training the model on a GPU
Predicting classes	The <code>predict.py</code> script successfully reads in an image and a checkpoint then prints the most likely image class and it's associated probability
Top K classes	The <code>predict.py</code> script allows users to print out the top K classes along with associated probabilities
Displaying class names	The <code>predict.py</code> script allows users to load a JSON file that maps the class values to other category names
Predicting with GPU	The <code>predict.py</code> script allows users to use the GPU to calculate the predictions