SAVE AND LOAD THE MODEL

In this section we will look at how to persist model state with saving, loading and running model predictions.

**import** torch

**import** torchvision.models **as** models

Saving and Loading Model Weights

PyTorch models store the learned parameters in an internal state dictionary, called state\_dict. These can be persisted via the torch.save method:

**model** **=** **models.vgg16(pretrained=True)**

**torch.save(model.state\_dict(),** 'model\_weights.pth'**)**

To load model weights, you need to create an instance of the same model first, and then load the parameters using load\_state\_dict() method.

**model** **=** **models.vgg16()** *# we do not specify pretrained=True, i.e. do not load default weights*

**model.load\_state\_dict(torch.load(**'model\_weights.pth'**))**

**model.eval()**

NOTE

be sure to call model.eval() method before inferencing to set the dropout and batch normalization layers to evaluation mode. Failing to do this will yield inconsistent inference results.

Saving and Loading Models with Shapes

When loading model weights, we needed to instantiate the model class first, because the class defines the structure of a network. We might want to save the structure of this class together with the model, in which case we can pass model (and not model.state\_dict()) to the saving function:

**torch.save(model,** 'model.pth'**)**

We can then load the model like this:

**model** **=** **torch.load(**'model.pth'**)**

NOTE

This approach uses Python [pickle](https://docs.python.org/3/library/pickle.html) module when serializing the model, thus it relies on the actual class definition to be available when loading the model.

Related Tutorials

[Saving and Loading a General Checkpoint in PyTorch](https://pytorch.org/tutorials/recipes/recipes/saving_and_loading_a_general_checkpoint.html)

**Total running time of the script:** ( 0 minutes 9.808 seconds)