Neural network for artistic style transfer (NST) in PyTorch

August 11, 2021

[42]:
```
import pystiche
from pystiche import demo, enc, loss, ops, optim
from pystiche.image import show_image
from pystiche.misc import get_device, get_input_image

print(f"I'm working with pystiche=={pystiche.__version__}")
device = get_device()
print(f"I'm working with {device}")
```

```
I'm working with pystiche==1.0.0.post0
I'm working with cuda
```

[43]:
```
multi_layer_encoder = enc.vgg19_multi_layer_encoder()
print(multi_layer_encoder)
```

```
VGGMultiLayerEncoder(
  arch=vgg19, framework=torch
  (preprocessing): TorchPreprocessing(
    (0): Normalize(mean=('0.485', '0.456', '0.406'), std=('0.229', '0.224',
'0.225'))
  )
  (conv1_1): Conv2d(3, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (relu1_1): ReLU()
  (conv1_2): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (relu1_2): ReLU()
  (pool1): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1,
ceil_mode=False)
  (conv2_1): Conv2d(64, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (relu2_1): ReLU()
  (conv2_2): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (relu2_2): ReLU()
  (pool2): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1,
ceil_mode=False)
  (conv3_1): Conv2d(128, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (relu3_1): ReLU()
  (conv3_2): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (relu3_2): ReLU()
  (conv3_3): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
```

```
  (relu3_3): ReLU()
  (conv3_4): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (relu3_4): ReLU()
  (pool3): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1,
ceil_mode=False)
  (conv4_1): Conv2d(256, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (relu4_1): ReLU()
  (conv4_2): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (relu4_2): ReLU()
  (conv4_3): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (relu4_3): ReLU()
  (conv4_4): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (relu4_4): ReLU()
  (pool4): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1,
ceil_mode=False)
  (conv5_1): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (relu5_1): ReLU()
  (conv5_2): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (relu5_2): ReLU()
  (conv5_3): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (relu5_3): ReLU()
  (conv5_4): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (relu5_4): ReLU()
  (pool5): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1,
ceil_mode=False)
)
```

[44]:
```
content_layer = "relu4_2"
content_encoder = multi_layer_encoder.extract_encoder(content_layer)
content_weight = 1e0
content_loss = ops.FeatureReconstructionOperator(
content_encoder, score_weight=content_weight)

print(content_loss)
```

```
FeatureReconstructionOperator(
  (encoder): VGGMultiLayerEncoder(layer=relu4_2, arch=vgg19, framework=torch)
)
```

```
C:\Users\Todor Velichkov\AppData\Roaming\Python\Python37\site-
packages\pystiche\ops\__init__.py:29: UserWarning: The class
pystiche.ops.FeatureReconstructionOperator is deprecated since pystiche==1.0 and
will be removed in a future release. It was renamed and moved to
pystiche.loss.FeatureReconstructionLoss. See
https://github.com/pystiche/pystiche/issues/436 for details
  warnings.warn(msg)
```

[47]:
```
style_layers = ("relu1_1", "relu2_1", "relu3_1", "relu4_1", "relu5_1")
style_weight = 1e3

def get_style_op(encoder, layer_weight):
    return ops.GramOperator(encoder, score_weight=layer_weight)

style_loss = ops.MultiLayerEncodingOperator(
multi_layer_encoder, style_layers, get_style_op, score_weight=style_weight,)

print(style_loss)
```

```
MultiLayerEncodingOperator(
  encoder=VGGMultiLayerEncoder(arch=vgg19, framework=torch), score_weight=1000
  (relu1_1): GramOperator(score_weight=0.2)
  (relu2_1): GramOperator(score_weight=0.2)
  (relu3_1): GramOperator(score_weight=0.2)
  (relu4_1): GramOperator(score_weight=0.2)
  (relu5_1): GramOperator(score_weight=0.2)
)
```

```
C:\Users\Todor Velichkov\AppData\Roaming\Python\Python37\site-
packages\pystiche\ops\__init__.py:29: UserWarning: The class
pystiche.ops.MultiLayerEncodingOperator is deprecated since pystiche==1.0 and
will be removed in a future release. It was renamed and moved to
pystiche.loss.MultiLayerEncodingLoss. See
https://github.com/pystiche/pystiche/issues/436 for details
  warnings.warn(msg)
C:\Users\Todor Velichkov\AppData\Roaming\Python\Python37\site-
packages\pystiche\ops\__init__.py:29: UserWarning: The class
pystiche.ops.GramOperator is deprecated since pystiche==1.0 and will be removed
in a future release. It was renamed and moved to pystiche.loss.GramLoss. See
https://github.com/pystiche/pystiche/issues/436 for details
  warnings.warn(msg)
```

[48]:
```
criterion = loss.PerceptualLoss(content_loss, style_loss).to(device)
print(criterion)
```

```
PerceptualLoss(
  (content_loss): FeatureReconstructionOperator(
    (encoder): VGGMultiLayerEncoder(layer=relu4_2, arch=vgg19, framework=torch)
  )
  (style_loss): MultiLayerEncodingOperator(
    encoder=VGGMultiLayerEncoder(arch=vgg19, framework=torch), score_weight=1000
    (relu1_1): GramOperator(score_weight=0.2)
    (relu2_1): GramOperator(score_weight=0.2)
    (relu3_1): GramOperator(score_weight=0.2)
    (relu4_1): GramOperator(score_weight=0.2)
    (relu5_1): GramOperator(score_weight=0.2)
```

```
    )
)
```

[61]:
```
images = demo.images()
images.download()
size = 500
```

[95]:
```
# Local Images

from pystiche.image import read_image

content_image = read_image("Ship.jpg", size=size, device=device)
```

[96]:
```
#content_image = images["bird1"].read(size=size, device=device)

show_image(content_image, title="Content image")
```

[97]:
```
style_image = images["paint"].read(size=size, device=device)
show_image(style_image, title="Style image")
```

[98]:
```
criterion.set_content_image(content_image)
criterion.set_style_image(style_image)
```

[99]:
```
starting_point = "random"
input_image = get_input_image(starting_point, content_image=content_image)
```

[100]:
```
starting_point = "content"
input_image = get_input_image(starting_point, content_image=content_image)
show_image(input_image, title="Input image")
```

[101]:
```
output_image = optim.image_optimization(input_image, criterion, num_steps=500)
```

```
HBox(children=(HTML(value='Image optimization'), FloatProgress(value=0.0,␣
↪max=500.0), HTML(value='')))
```

[102]:
```
show_image(output_image, title="Output image")
```