INDIAN INSTITUTE OF TECHNOLOGY
JODHPUR

DEEP LEARNING

CSL7590
PROJECT REPORT

Instructure: Dr Mayank Vatsa

Submitted by: Shrikesh Jalan (M21MA012)

Searching for MobileNetV3 - Results

Dataset: MNIST Fashion

The model is trained on MobileNetV1, MobileNetV2, and MobileNetV3 are trained with fashion-MNIST dataset and the output is given below.

```
trainset = torchvision.datasets.FashionMNIST(root = "./data", train = True, download = True)
testset = torchvision.datasets.FashionMNIST(root = "./data", train = False, download = True)

Downloading http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/train-images-idx3-ubyte.gz to ./data/FashionMNIST/raw/train-images-idx3-ubyte.gz
HBox(children=(FloatProgress(value=1.0, bar_style='info', max=1.0), HTML(value='')))
Extracting ./data/FashionMNIST/raw/train-images-idx3-ubyte.gz to ./data/FashionMNIST/raw
Downloading http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/train-labels-idx1-ubyte.gz to ./data/FashionMNIST/raw/train-labels-idx1-ubyte.gz
HBox(children=(FloatProgress(value=1.0, bar_style='info', max=1.0), HTML(value='')))
Extracting ./data/FashionMNIST/raw/train-labels-idx1-ubyte.gz to ./data/FashionMNIST/raw
Downloading http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/t10k-images-idx3-ubyte.gz to ./data/FashionMNIST/raw/t10k-images-idx3-ubyte.gz
HBox(children=(FloatProgress(value=1.0, bar_style='info', max=1.0), HTML(value='')))

Extracting ./data/FashionMNIST/raw/t10k-images-idx3-ubyte.gz to ./data/FashionMNIST/raw
Downloading http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/t10k-labels-idx1-ubyte.gz to ./data/FashionMNIST/raw/t10k-labels-idx1-ubyte.gz
HBox(children=(FloatProgress(value=1.0, bar_style='info', max=1.0), HTML(value='')))
Extracting ./data/FashionMNIST/raw/t10k-labels-idx1-ubyte.gz to ./data/FashionMNIST/raw
Processing...
Done!
/usr/local/lib/python3.6/dist-packages/torchvision/datasets/mnist.py:480: UserWarning: The given NumPy array is not writeable, and PyTorch does not support
  return torch.from_numpy(parsed.astype(m[2], copy=False).view(*s)
```

MNIST FASHION data download

Training of MobileNetV1, MobileNetV2, and MobileNetV3:

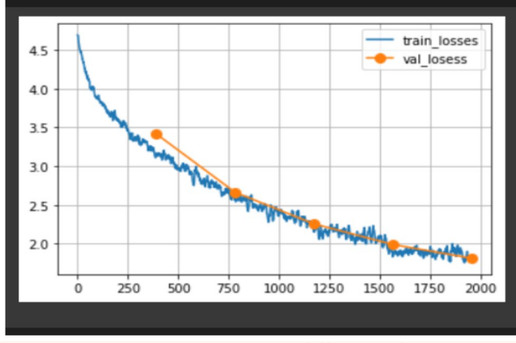
```
model = MobileNet1().to(device)
epochs = 5
lr = 1e-3
train_losses, val_losses = train(train_loader, val_loader, model, epochs, lr)

HBox(children=(FloatProgress(value=0.0, max=5.0), HTML(value='')))
EPOCH # 1
HBox(children=(FloatProgress(value=0.0, max=391.0), HTML(value='')))

train_loss: 3.6893 | train_acc: 0.1293
HBox(children=(FloatProgress(value=0.0, max=79.0), HTML(value='')))

  val_loss: 3.4165 |   val_acc: 0.1854
EPOCH # 2
HBox(children=(FloatProgress(value=0.0, max=391.0), HTML(value='')))

train_loss: 2.8927 | train_acc: 0.2681
HBox(children=(FloatProgress(value=0.0, max=79.0), HTML(value='')))

  val_loss: 2.6671 |   val_acc: 0.3182
EPOCH # 3
HBox(children=(FloatProgress(value=0.0, max=391.0), HTML(value='')))

train_loss: 2.4207 | train_acc: 0.3630
HBox(children=(FloatProgress(value=0.0, max=79.0), HTML(value='')))

  val_loss: 2.2628 |   val_acc: 0.4030
EPOCH # 4
HBox(children=(FloatProgress(value=0.0, max=391.0), HTML(value='')))

train_loss: 2.1123 | train_acc: 0.4347
HBox(children=(FloatProgress(value=0.0, max=79.0), HTML(value='')))
```

MobileNet1 training

```
start = time.time()
val_acc = get_acc(model, val_loader)
end = time.time()

print('Batch_size={}, epochs={}, lr={}'.format(128, epochs, lr))
print('Val accuracy =', val_acc)
print("Process validation time: {:0.4f} s".format(end - start) )

HBox(children=(FloatProgress(value=0.0, max=79.0), HTML(value='')))

Batch_size=128, epochs=5, lr=0.001
Val accuracy = 0.5061
Process validation time: 16.1220 s
```

MobileNet1 Validation

```
plot_losses(train_losses, val_losses)
```



MobileNet1 losses

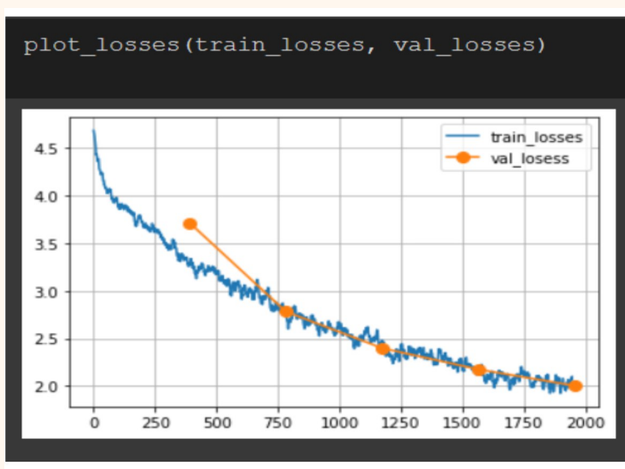
```
model = MobileNetV2().to(device)
epochs = 5
lr = 1e-3
train_losses, val_losses = train(train_loader, val_loader, model, epochs, lr)

HBox(children=(FloatProgress(value=0.0, max=5.0), HTML(value='')))
EPOCH # 1
HBox(children=(FloatProgress(value=0.0, max=391.0), HTML(value='')))

train_loss: 3.7707 | train_acc: 0.1111
HBox(children=(FloatProgress(value=0.0, max=79.0), HTML(value='')))

  val_loss: 3.7133 |   val_acc: 0.1397
EPOCH # 2
HBox(children=(FloatProgress(value=0.0, max=391.0), HTML(value='')))

train_loss: 3.0537 | train_acc: 0.2323
HBox(children=(FloatProgress(value=0.0, max=79.0), HTML(value='')))

  val_loss: 2.7829 |   val_acc: 0.2914
EPOCH # 3
HBox(children=(FloatProgress(value=0.0, max=391.0), HTML(value='')))

train_loss: 2.6039 | train_acc: 0.3165
HBox(children=(FloatProgress(value=0.0, max=79.0), HTML(value='')))

  val_loss: 2.4011 |   val_acc: 0.3595
EPOCH # 4
HBox(children=(FloatProgress(value=0.0, max=391.0), HTML(value='')))
```

MobileNet2 Training

```
start = time.time()
val_acc = get_acc(model, val_loader)
end = time.time()

print('Batch_size={}, epochs={}, lr={}'.format(128, epochs, lr))
print('Val accuracy =', val_acc)
print("Process validation time: {:0.4f} s".format(end - start) )

HBox(children=(FloatProgress(value=0.0, max=79.0), HTML(value='')))

Batch_size=128, epochs=5, lr=0.001
Val accuracy = 0.4506
Process validation time: 16.6421 s
```

MobileNet2 Validation

```
plot_losses(train_losses, val_losses)
```



MobileNet2 Losses

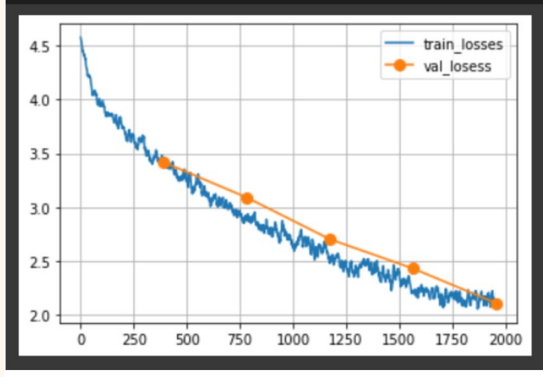
```
model = MobileNetV3Small().to(device)
epochs = 5
lr = 1e-3
train_losses, val_losses = train(train_loader, val_loader, model, epochs, lr)

HBox(children=(FloatProgress(value=0.0, max=5.0), HTML(value='')))
EPOCH # 1
HBox(children=(FloatProgress(value=0.0, max=391.0), HTML(value='')))

train_loss: 3.8013 | train_acc: 0.1053
HBox(children=(FloatProgress(value=0.0, max=79.0), HTML(value='')))

  val_loss: 3.4138 |   val_acc: 0.1716
EPOCH # 2
HBox(children=(FloatProgress(value=0.0, max=391.0), HTML(value='')))

train_loss: 3.1493 | train_acc: 0.2164
HBox(children=(FloatProgress(value=0.0, max=79.0), HTML(value='')))

  val_loss: 3.0889 |   val_acc: 0.2381
EPOCH # 3
HBox(children=(FloatProgress(value=0.0, max=391.0), HTML(value='')))

train_loss: 2.7333 | train_acc: 0.2957
HBox(children=(FloatProgress(value=0.0, max=79.0), HTML(value='')))

  val_loss: 2.7053 |   val_acc: 0.3126
EPOCH # 4
HBox(children=(FloatProgress(value=0.0, max=391.0), HTML(value='')))

train_loss: 2.4096 | train_acc: 0.3647
```

MobileNet3 Training

```
start = time.time()
val_acc = get_acc(model, val_loader)
end = time.time()

print('Batch_size={}, epochs={}, lr={}'.format(128, epochs, lr))
print('Val accuracy =', val_acc)
print("Process validation time: {:0.4f} s".format(end - start) )

HBox(children=(FloatProgress(value=0.0, max=79.0), HTML(value='')))

Batch_size=128, epochs=5, lr=0.001
Val accuracy = 0.4372
Process validation time: 16.0766 s
```

MobileNet3 Validation

```
plot_losses(train_losses, val_losses)
```

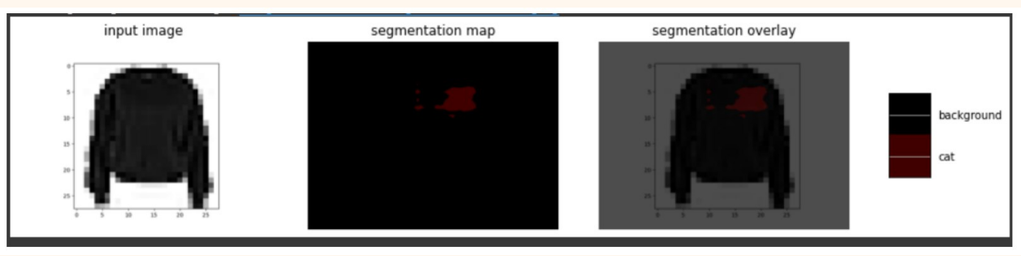


MobileNet3 losses

```
SAMPLE_IMAGE = 'image1'
IMAGE_URL = 'https://i.stack.imgur.com/CAZAB.png'

def run_visualization(url):
  """Inferences DeepLab model and visualizes result."""
  try:
    f = urllib.request.urlopen(url)
    jpeg_str = f.read()
    original_im = Image.open(BytesIO(jpeg_str))
  except IOError:
    print('Cannot retrieve image. Please check url: ' + url)
    return

  print('running deeplab on image %s...' % url)
  resized_im, seg_map = MODEL.run(original_im)

  vis_segmentation(resized_im, seg_map)


image_url = IMAGE_URL or _SAMPLE_URL % SAMPLE_IMAGE
run_visualization(image_url)
```

Test visualization

Segmentation Map and overlayed Output:



Output of MNIST Fashion data link

MobileNetV3 was announced as a new state-of-the-art in mobile classification, detection, and segmentation. Multiple network architectural search techniques, as well as innovations in network design, are being used to produce the next generation of mobile models, as outlined in this study. MobileNetV3 demonstrates how to adapt nonlinearities such as swish and apply squeeze and excite in a quantization-friendly and efficient manner, presenting them as useful tools in the mobile model domain, and I have applied the same nonlinearity in the architecture.