Readme

PyTorch implementation of 'On Minimizing Diagonal Block-wise Differences for Neural Network Compression'.

There are four different models:

- Lenet5 on MNIST
- VGG16 on Cifar10
- VGG16 on Cifar100
- AlexNet on Cifar100

Requirements

- python3.6+
- tqdm
- numpy
- Pytorch 0.4.1, torchvision
- scipy
- scikit-learn

Usage

Training model with different compression algorithms

- Lenet5 on MNIST (/code/lenet5_mnist)
    - DeepC (/code/lenet5_mnist/1_deep_compression)

```
$ python main.py -mm d -tm 4 -b conv=8,fc=5 -ep 230 -reep 20 -sd model_deepcomp
ression
or
$ sh run.sh
```

    - FC+Q (/code/lenet5_mnist/1_deep_compression)

```
$ python main.py -mm d -tm 3 -b conv=8,fc=5 -ep 230 -reep 20 -sd model_fcq -lm
 model_deepcompression/model_initial_end.ptmodel
```

- MPDC+Q (/code/lenet5_mnist/2_mpd_compression)

```
$ python main.py -mm d -tm 5 -b conv=8,fc=5 -p fc1=10,fc2=10 -ep 230 -qep 10 -a
l 0.0 -sd model_mpd
or
$ sh run.sh
```

- MESA (/code/lenet5_mnist/3_mesa_compression)

```
$ python main.py -mm d -tm 5 -b conv=8,fc=5 -p fc1=10,fc2=10 -ep 230 -qep 20 -a
l 0.0    -sd model_mesa_0.0
$ python main.py -mm d -tm 5 -b conv=8,fc=5 -p fc1=10,fc2=10 -ep 230 -qep 20 -a
l 0.0001 -sd model_mesa_0.0001
$ python main.py -mm d -tm 5 -b conv=8,fc=5 -p fc1=10,fc2=10 -ep 230 -qep 20 -a
l 0.001  -sd model_mesa_0.001
$ python main.py -mm d -tm 5 -b conv=8,fc=5 -p fc1=10,fc2=10 -ep 230 -qep 20 -a
l 0.01   -sd model_mesa_0.01
$ python main.py -mm d -tm 5 -b conv=8,fc=5 -p fc1=10,fc2=10 -ep 230 -qep 20 -a
l 0.1    -sd model_mesa_0.1
or
$ sh run.sh
```

- VGG16 on Cifar10 (/code/vgg16_c10)
  - DeepC (/code/vgg16_c10/1_deep_compression)

```
$ python main.py -mm d -tm 4 -b conv=8,fc=5 -ep 250 -reep 20 -sd model_deepcomp
ression
or
$ sh run.sh
```

  - FC+Q (/code/vgg16_c10/1_deep_compression)

```
$ python main.py -mm d -tm 3 -b conv=8,fc=5 -ep 250 -reep 20 -sd model_fcq -lm
 model_deepcompression/model_initial_end.ptmodel
```

  - MPDC+Q (/code/vgg16_c10/2_mpd_compression)

```
$ python main.py -mm d -tm 5 -b 5 -p fc1=8,fc2=8,fc3=10 -ep 250 -qep 10 -al 0.0
 -sd model_mpd
```

```
or
$ sh run.sh
```

- MESA (/code/vgg16_c10/3_mesa_compression)

```
$ python main.py -mm d -tm 5 -b 5 -p fc1=8,fc2=8,fc3=10 -ep 250 -qep 10 -al 0.0
    -sd model_mesa_0.0
$ python main.py -mm d -tm 5 -b 5 -p fc1=8,fc2=8,fc3=10 -ep 250 -qep 10 -al 0.0
001 -sd model_mesa_0.0001
$ python main.py -mm d -tm 5 -b 5 -p fc1=8,fc2=8,fc3=10 -ep 250 -qep 10 -al 0.0
01  -sd model_mesa_0.001
$ python main.py -mm d -tm 5 -b 5 -p fc1=8,fc2=8,fc3=10 -ep 250 -qep 10 -al 0.0
1   -sd model_mesa_0.01
$ python main.py -mm d -tm 5 -b 5 -p fc1=8,fc2=8,fc3=10 -ep 250 -qep 10 -al 0.1
    -sd model_mesa_0.1
or
$ sh run.sh
```

- VGG16 on Cifar100 (/code/vgg16_c100)
  - DeepC (/code/vgg16_c100/1_deep_compression)

```
$ python main.py -mm d -tm 4 -b conv=8,fc=5 -ep 300 -reep 10 -lr=0.1 -sd model_
deepcompression
or
$ sh run.sh
```

  - FC+Q (/code/vgg16_c100/1_deep_compression)

```
$ python main.py -mm d -tm 3 -b conv=8,fc=5 -ep 300 -reep 10 -lr=0.1 -sd model_
fcq -lm model_deepcompression/model_initial_end.ptmodel
```

  - MPDC+Q (/code/vgg16_c100/2_mpd_compression)

```
$ python main.py -mm d -tm 5 -b 5 -p fc1=10,fc2=10,fc3=10 -lr 0.1 -ep 300 -qep
10 -al 0.0 -sd model_mpd
or
$ sh run.sh
```

  - MESA (/code/vgg16_c10/3_mesa_compression)

```
$ python main.py -mm d -tm 5 -b 5 -p fc1=10,fc2=10,fc3=10 -lr 0.1 -ep 300 -qep
10 -al 0.0    -sd model_mesa_0.0
$ python main.py -mm d -tm 5 -b 5 -p fc1=10,fc2=10,fc3=10 -lr 0.1 -ep 300 -qep
10 -al 0.0001 -sd model_mesa_0.0001
$ python main.py -mm d -tm 5 -b 5 -p fc1=10,fc2=10,fc3=10 -lr 0.1 -ep 300 -qep
10 -al 0.001  -sd model_mesa_0.001
$ python main.py -mm d -tm 5 -b 5 -p fc1=10,fc2=10,fc3=10 -lr 0.1 -ep 300 -qep
10 -al 0.01   -sd model_mesa_0.01
$ python main.py -mm d -tm 5 -b 5 -p fc1=10,fc2=10,fc3=10 -lr 0.1 -ep 300 -qep
10 -al 0.1    -sd model_mesa_0.1
or
$ sh run.sh
```

- AlexNet on Cifar100 (/code/alexnet_c100)
    - DeepC (/code/alexnet_c100/1_deep_compression)

```
$ python main.py -mm d -tm 4 -b conv=3,fc=5 -lr 0.1 -ep 300 -reep 20 -sd model_
deepcompression
or
$ sh run.sh
```

    - FC+Q (/code/alexnet_c100/1_deep_compression)

```
$ python main.py -mm d -tm 3 -b conv=3,fc=5 -lr 0.1 -ep 300 -reep 20 -sd model_
fcq -lm model_deepcompression/model_initial_end.ptmodel
```

    - MPDC+Q (/code/alexnet_c100/2_mpd_compression)

```
$ python main.py -mm d -tm 5 -b 5 -p fc1=10,fc2=10,fc3=4 -al 0.0 -lr 0.1  -ep 3
00 -qep 10 -sd model_mpd
or
$ sh run.sh
```

    - MESA (/code/alexnet_c100/3_mesa_compression)

```
$ python main.py -mm d -tm 5 -b 5 -p fc1=10,fc2=10,fc3=4 -al 0.0    -lr 0.1  -e
p 300 -qep 10 -sd model_mesa_0.0
$ python main.py -mm d -tm 5 -b 5 -p fc1=10,fc2=10,fc3=4 -al 0.0001 -lr 0.1  -e
p 300 -qep 10 -sd model_mesa_0.0001
```

```
$ python main.py -mm d -tm 5 -b 5 -p fc1=10,fc2=10,fc3=4 -al 0.001  -lr 0.1  -e
p 300 -qep 10 -sd model_mesa_0.001
$ python main.py -mm d -tm 5 -b 5 -p fc1=10,fc2=10,fc3=4 -al 0.01   -lr 0.1  -e
p 300 -qep 10 -sd model_mesa_0.01
$ python main.py -mm d -tm 5 -b 5 -p fc1=10,fc2=10,fc3=4 -al 0.1    -lr 0.1  -e
p 300 -qep 10 -sd model_mesa_0.1
or
$ sh run.sh
```

Doing Huffman coding

Put the trained model (checkpoint file/ptmodel file) to the corresponding algorithm folder in the folder (.../compression_rate_calculate/model/).

- Lenet5 on MNIST (/code/lenet5_mnist/compression_rate_calculate/src)

```
python main.py -lmd ../model/deepc/model_quantized_retrain10.ptmodel -lmq ../model/fc_
q/model_quantized_retrain10.ptmodel -lmm  ../model/mesa/checkpoint_quantized_re_alpha_
0.0_100.tar -lmp ../model/mpd/checkpoint_initial_p_alpha_0.0_200.tar -lmi ../model/fc/
model_initial_end.ptmodel -p fc1=10,fc2=10 -b conv=8,fc=5 -tm d -sd ../output_compress
ion
```

- VGG16 on Cifar10 (/code/vgg16_c10/compression_rate_calculate/src)

```
python main.py -lmd ../model/deepc/model_quantized_retrain10.ptmodel -lmq ../model/fc_
q/model_quantized_retrain10.ptmodel -lmm ../model/mesa/checkpoint_quantized_re_alpha_
0.0_10.tar -lmi ../model/fc/model_initial_end.ptmodel -lmp ../model/mpd/checkpoint_ini
tial_p_alpha_0.0_100.tar -p fc1=8,fc2=8,fc3=10 -b conv=8,fc=5 -tm d -sd ../output_comp
ression
```

- VGG16 on Cifar100 (/code/vgg16_c100/compression_rate_calculate/src)

```
python main.py -lmd ../model/deepc/model_quantized_retrain10.ptmodel -lmq ../model/fc_
q/model_quantized_retrain10.ptmodel -lmm ../model/mesa/checkpoint_quantized_re_alpha_
0.0_5.tar -lmp ../model/mpd/checkpoint_initial_p_alpha_0.0_250.tar -lmi ../model/fc/mo
del_initial_end.ptmodel -p fc1=8,fc2=8,fc3=10 -b conv=5,fc=5 -tm d -sd ../output_compr
ession
```

- AlexNet on Cifar100 (/code/alexnet_c100/compression_rate_calculate/src)

```
$ python main.py -lmd ../model/deepc/model_quantized_retrain50.ptmodel -lmq ../model/fc_q/model_quantized_retrain50.ptmodel -lmm ../model/mesa/checkpoint_quantized_re_alpha_0.0_0.tar -lmp ../model/mpd/checkpoint_initial_p_alpha_0.0_200.tar -lmi ../model/fc/model_initial_end.ptmodel -p fc1=10,fc2=10,fc3=4 -b conv=8,fc=5 -tm d -sd ../output_compression
```

code modified from https://github.com/mightydeveloper/Deep-Compression-PyTorch