CS6886 - Systems Engineering for DL

Assignment-3: Model Compression

Name: **Varun Sunderarajan**

Roll No.: **NS25Z318**

Initial setup

First clone the repository (https://github.com/DextroLaev/CS6886-Sys_for_dl-Assignment3) and setup pip environment with torch, torchvision, numpy and pillow. Next step is to runthe given command and see that everything is in order:

```
python test.py --weight_quant_bits 8 --activation_quant_bits 8
```

Output upon running the command:

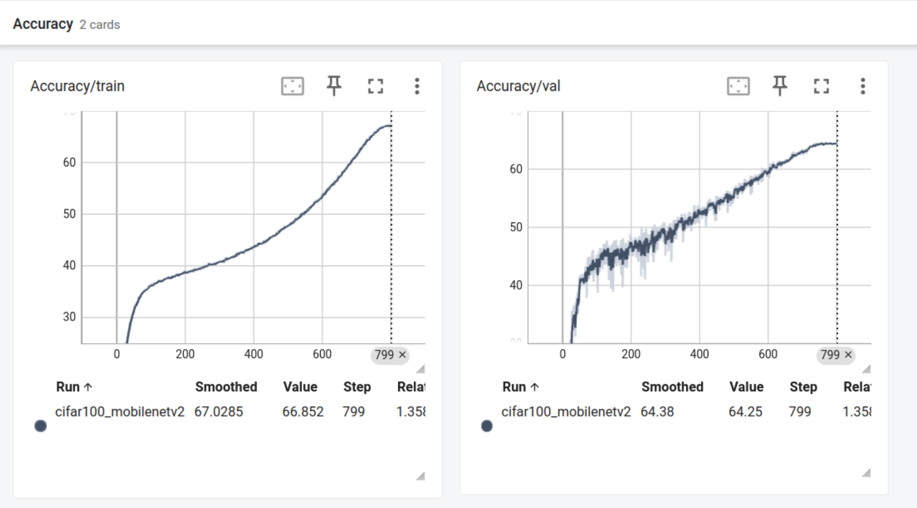
```
Test Acc=91.16%
Quantized Test Acc=91.13%
=== Compression Summary ===
FP32 model size:   128.35 MB
Quantized size:    32.14 MB (weights=8-bit)
Compression ratio: 3.99x
===========================
```

This is on **VGG16** architecture trained on **CIFAR-10** dataset. The architecture needs to be changed to **MobileNet-v2**, training needs to be done **CIFAR-100**. After that, weight_quant_bits and activation_quant_bits needs to be changed in order to get optimal dataset.

1. Training Baseline: MobileNet-v2 on CIFAR-10 dataset

- There is existing `dataloader.py` python file for loading **CIFAR-10**. This has been used modified for **CIFAR-100**. This dataset has been downloaded in `.tar.gz` format and processed.
- The `main.py` is meant for training **VGG.py**. It has been copied as `train_MobileNet.py` and appropriate adjustments have been made. model configuration details are as follows:
  - Width multiplier: 1.0
  - Dropout: 0.5
  - For BN, default `nn.BatchNorm2d` is used
- Training strategy is as follows:
  - Optimizer: `torch.optim.SGD`
  - LR schedule: `torch.optim.lr_scheduler.CosineAnnealingLR`
  - Regularization: `lr=0.05, weight_decay=5e-4, momentum=0.9`
  - Epochs: `800`
  - Batchsize: `256`

  - Dropout: `0.3`
- At the end of 800 epochs it appeared that the validation loss was saturated around 64%. The training loss was increasing marginally, but to 66%-67%, but also beginnnig to saturate.



Run ↑	Smoothed	Value	Step	Rela
● cifar100_mobilenetv2	67.0285	66.852	799	1.35

Run ↑	Smoothed	Value	Step	Rela
● cifar100_mobilenetv2	64.38	64.25	799	1.35

Figure 1: train/val accuracy

Loss decrease also appears to have saturated.

Overall result looks reasonable given that this is on **CIFAR-100**.

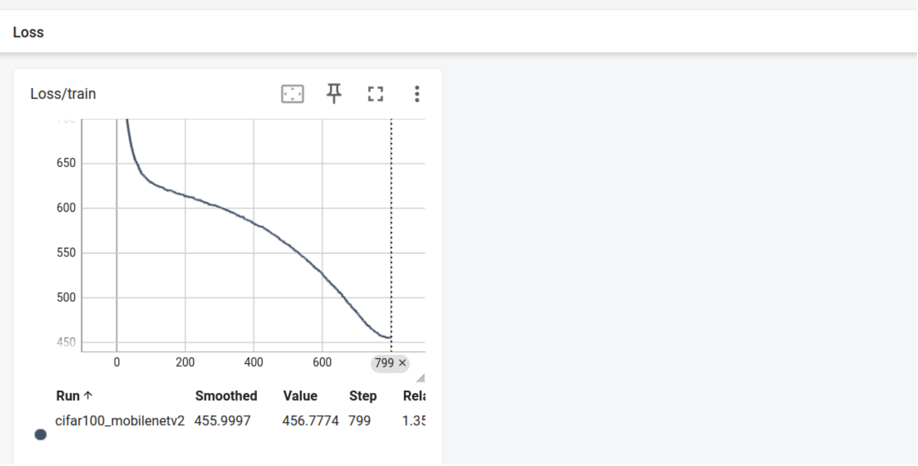
2. Model Compression Implementation

- Quantization technique is used for model The existing `quantize.py` has been used. In `test.py` the previous `VGG` model has been replaced by `mobilenetv2`.
- Compression ratio has been calculated as per the given code which will be seen upon running `test.py`.

3. Compression Results

- Compression pipeline at different levels of compression: Compression is being done with different levels of quantization `[1, 2, 4, 8]` for both weight and activation. It has been done as follows:

```
for w in 1 2 4 8 16; do
  for a in 1 2 4 8 16; do
    python test.py --weight_quant_bits $w --activation_quant_bits $a
  done
done
```



	Run ↑	Smoothed	Value	Step	Rel
●	cifar100_mobilenetv2	455.9997	456.7774	799	1.35

Figure 2: loss

- Evaluation and comparism of accuracy in these settings. The data manually entered upon running the code is as follows:

```
data = {
    'activation_quant_bits': [1, 2, 4, 8, 16],
    'weight_quant_bits': [1, 2, 4, 8, 16],
    'compression_ratio': [26.10, 14.42, 7.61, 3.91, 1.99],
    'model_size_mb': [0.34, 0.62, 1.18, 2.29, 8.97],
    'quantized_acc': [1.0, 1.0, 18.63, 64.29, 64.25],
    'label': ['A', 'B', 'C', 'D', 'E']
}
```

Note actually 25 experiments took place; whichever the `weight_quant_bits` is set to, everything else appeared same.

It appears that `activation_quant_bits=8` appears most optimal. The model is small compared to 16bit and accuracy of 4bit is too low.

4. Compression Analysis

1. Compression ratio of *model*: 3.91x
2. Compression ratio of *weights*: ~4.00x
3. Compression ratio of *activation*: ~0.00x
4. Final approximated *model size* (MB) after compression: 2.29 MB

The function `print_compression` has been modified and ratio of weights calculated as follows:

```
quant_weights_size = sum(p.numel() for p in model.parameters()) * (weight_bits / 8)
```

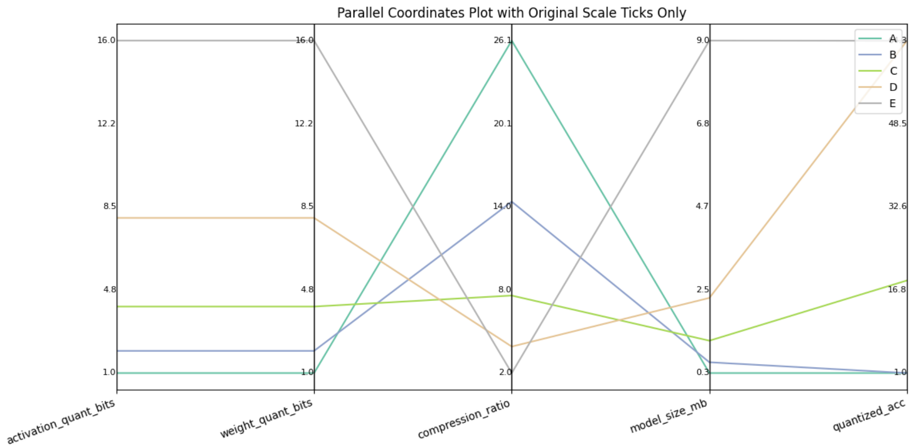


Figure 3: Wandb plot

The remaining ones have been taken as activation

`quant_activations_size = quant_size - quant_weights_size`

This is just an approximation as it can't be zero. But activation weights appear to be of not much significance.

5. Reproducibility & Repository

1. For training `train_MobileNet.py` is to be used; for evaluation `test.py` has been modified appropriately; compression code is in `quantize.py`.
2. The original README has been renamed and a new one has been added.
3. GitHub repository link is as follows:

https://github.com/varun-analyst/CS6886-Sys__for__dl-Assignment3