Aeden Jameson
EEP-596 Homework 1

Assumptions:

- All code was run on the A100 GPU
- All model sizes are of a gzipped tflite model in order to see the practical benefits of various optimization techniques. For example, in order to see the model size benefits of pruning both `strip_pruning` and a compression algorithm (e.g. via gzip) are necessary to apply.
    - The strip_pruning method removes the wrappers that are only needed during training, which would otherwise add to model size.
    - A compression algorithm is necessary since the serialized weight matrices are the same size as they were before pruning. However, pruning makes most of the weights zeros, which is added redundancy that algorithms can utilize to further compress the model.
- For problems #2 & #4, due to the extremely long evaluation times on the training set I only evaluated 900 randomly selected images from the training set.
- For problem #6, in order to achieve a like-for-like comparison I ran the inference for all models as a TFLite model.

1. Model: M
    - Training Accuracy: 99.78%
    - Test Accuracy: 89.67%
    - Model Size: 97.50 MB

2. Model: M_DRQ
    - Training Accuracy: 99.67%
    - Test Accuracy: 89.78%
    - Model Size: 22.76 MB

    Model: M_FIQ
    - Training Accuracy: 99.67%%
    - Test Accuracy: 89.78%%
    - Model Size: 22.77 MB

    Model: M_F16Q
    - Training Accuracy: 99.78%
    - Test Accuracy: 89.67%
    - Model Size: 48.12 MB

3. Model: M_P1

- Training Accuracy: 99.72%
- Test Accuracy: 91.22%
- Model Size: 59.75 MB

Model: M_P2

- Training Accuracy: 97.17%
- Test Accuracy: 88.33%
- Model Size: 36.40 MB

Discussion:

They are certainly considerations for pruning a neural network differently depending on whether the value of the initial sparsity is 0 or 0.5. Below we discuss the benefits and drawbacks for each value.

Case inititial_sparsity = 0

- Benefit: Early layers often capture general features of the dataset and so if significant weights happen to be set to zero this can affect model accuracy. Thus you can prune layers with slightly few parameters or layers earlier in the network.
- Drawback: One may need to more thoroughly search through the model to find which and under what characteristics to prune a layer. You can see my effort in doing this in my custom pruning policy. I created this to try and balance accuracy with the model size.

Case inititial_sparsity = 0.5

- Benefit: One may *not* need to search through the model as carefully to find which layers to prune. One may be able to focus mostly on the layers with the most parameters like the first dense layer in my model for the homework. That first dense layer contained over 60% of the total parameters.
- Benefit: A bunch of zeros is more memory efficient, but this shouldn't come at the cost of excessive accuracy degradation.
- Drawback: Early layers often capture general features of the dataset and so if significant weights happen to be set to zero this can affect model accuracy. Thus one wouldn't want to prune earlier layers or layers with too few parameters.

4. Model: M_P1_DRQ
    - Training Accuracy: 97.44%
    - Test Accuracy: 88.11%
    - Model Size: 9.58 MB

    Model: M_P1_FIQ

- Training Accuracy: 97.33%
- Test Accuracy: 88.22%
- Model Size: 9.59 MB

Model: M_P1_F16Q
- Training Accuracy: 97.56%
- Test Accuracy: 88.33%
- Model Size: 21.36 MB

Model: M_P2_DRQ
- Training Accuracy: 97.44%
- Test Accuracy: 88.11%
- Model Size: 9.58 MB

Model: M_P2_FIQ
- Training Accuracy: 97.33%
- Test Accuracy: 88.22%
- Model Size: 9.59 MB

Model: M_P2_F16Q
- Training Accuracy: 97.56%
- Test Accuracy: 88.33 %
- Model Size: 21.36 MB

5. Model: M
   - Average: 517.33 ms
   - Standard Deviation: 4.12 ms

   Model: M_DRQ
   - Average: 380.88 ms
   - Standard Deviation: 1.84 ms

   Model: M_FIQ
   - Average: 378.78 ms
   - Standard Deviation: 3.71 ms

   Model: M_F16Q
   - Average: 513.82 ms
   - Standard Deviation: 3.99 ms

   Model: M_P1
   - Average: 511.90 ms
   - Standard Deviation: 3.86 ms

Model: M_P2
- Average: 511.98 ms
- Standard Deviation: 5.67 ms

Model: M_P1_DRQ
- Average: 382.79 ms
- Standard Deviation: 2.42 ms

Model: M_P1_FIQ
- Average: 379.02 ms
- Standard Deviation: 3.78 ms

Model: M_P1_F16Q
- Average: 506.01 ms
- Standard Deviation: 3.58 ms

Model: M_P2_DRQ
- Average: 383.16 ms
- Standard Deviation: 1.62 ms

Model: M_P2_FIQ
- Average: 380.10 ms
- Standard Deviation: 4.77 ms

Model: M_P2_F16Q
- Average: 506.40 ms
- Standard Deviation: 3.05 ms

6. Cascading compression techniques offer significant advantages over a single technique by reduced model size with minimal loss of accuracy. In all cases cascading reduced model size 3-6x when compared to just the pruned models and from 2-5 times when compared to the quantized model depending on the technique used. In addition there was only ~2% decrease in training and 1% decrease in test accuracies. That reduction is beneficial to resource constrained environments these models are deployed in. Lastly the cascaded models that used DRQ and FIQ benefitted from the increased inference time those techniques provided in the single technique case. On the other hand the test accuracy of the M_P1 model increased by ~1.5% while seeing a reduction in model size from the base model by ~50%. However the size of the model is still about ~48MB, which may be too large for some environments. That said, cascading techniques can offer improvements in size and while retaining the inference speed of DRQ or FIQ for resource constrained environments over a single technique.