CS 4803/8803 Efficient ML: Homework Project 1
Drawing Early-Bird Tickets

Goal: Understand and implement standard pruning techniques, based on that, the students are encouraged to take more adventure in drawing lottery tickets from dense networks. It is also desired to find those lottery tickets in the early training stages.

Reference Codebase:
- Lottery Ticket Hypothesis
- Early-Bird Tickets

Tasks:

Hint:
Many step 1 in four tasks are shareable, it is wise to read all task descriptions first and determine what you need to save during training, e.g., checkpoints, initialization, etc.

- **[15%] Task 1:**
  Get familiar with the technical details and codebases of Lottery Ticket Hypothesis and Early-Bird Tickets. Please briefly summarize the two papers, illustrating their motivation, key insights, methodologies, and experimental results.

  - Deliverables:
    - Your summarization of these two papers.

- **[25%] Task 2:**
  Based on a ResNet-20 model and CIFAR-10 dataset, use the magnitude-based pruning method to prune it and retrain it to recover the accuracy.
  - Step 1: Train ResNet-20 from scratch to get the pre-trained weights
  - Step 2: Prune it according to weight magnitude, i.e., the smaller the less important and should be pruned. Try pruning ratio sets {50%, 70%, 90%}.
  - Step 3: Retrain the pruned network to recover the accuracy, remember to record the loss and accuracy trajectory along with training epochs.

  - Deliverables:
    - Codes and readme commands to run it, pack in zip file
    - Training trajectories (i.e., loss/accuracy vs. epochs/steps) including
      - ResNet-20 training trajectory as in Step 1
      - Three pruned ResNet-20 training trajectories as in Step 3
    - Training results including

            - Original ResNet-20 accuracy
            - Pruned ResNet-20 accuracy at three pruning ratios
            - Retrained pruned Resnet-20 accuracy at three pruning ratios
        - Your result analysis
            - Whether retraining is necessary?
            - How many accuracy improvements are brought by pruning?

- **[30%] Task 3:**
  Reproduce the lottery ticket hypothesis. The difference with the magnitude-based one is that, instead of inheriting weights from pre-trained networks, we rewind the weights to the random initialization with learned masks before retraining.
    - Step 1: Train ResNet-20 from scratch to get the pre-trained weights, remember to keep the initialized weights, i.e., the initialized model weights before training.
    - Step 2: Prune it according to weight magnitude, i.e., the smaller the less important and should be pruned. Try pruning ratio sets {50%, 70%, 90%}. Remember to apply the mask to initialized weights instead.
    - Step 3: Retrain the pruned initialization, i.e., apply the pruned mask to the saved initialized weights, to recover the accuracy, remember to record the loss and accuracy trajectory along with training epochs.
    - Step 4: Repeat Steps 1 - 3 iteratively, i.e., iterative magnitude pruning to get the final pruned model.

    - Deliverables:
        - Codes and readme commands to run it, pack in zip file
        - Training trajectories including
            - ResNet-20 training trajectory as in Step 1
            - Three pruned ResNet-20 training trajectories as in Step 3
            - Another set of three pruned ResNet-20 training trajectories as in Step 4
        - Training results including
            - Original ResNet-20 accuracy
            - Pruned ResNet-20 accuracy at three pruning ratios
            - Retrained pruned Resnet-20 accuracy at three pruning ratios
        - Your result analysis
            - Whether you observe the lottery ticket phenomenon?
            - Difference between task 2 and task 3?

- **[30%] Task 4:** There exist early-bird tickets that do not need full training on the dense network before pruning, but at which point should we stop training and find the sparse network? Please investigate this.
    - Step 1: Train ResNet-20 from scratch to get the pre-trained weights at different training epochs. Try epoch sets {10, 30, 50, 100} out of 160. Note: you only need to train the model once but remember to save the checkpoints at those epochs.
    - Step 2: Prune the weights at different training epochs structurally according to the scaling factor in normalization layers. The smaller the less important and should be pruned. Reference code can be found at the network slimming repository. Try pruning ratio sets {30%, 50%}.
    - Step 3: Retrain the pruned network to recover the accuracy, remember to record the loss and accuracy trajectory along with training epochs.

    - Deliverables:
        - Codes and readme commands to run it, pack in zip file
        - Training trajectories including
            - Two pruned ResNet-20 training trajectories as in Step 3 for every epoch choice in epoch set {10, 30, 50, 100}; There are 8 trajectories in total.
        - Training results including
            - Original ResNet-20 accuracy
            - Pruned ResNet-20 accuracy at two pruning ratios and four epoch choices
            - Retrained pruned Resnet-20 accuracy at two pruning ratios and four epoch choices
        - Your result analysis
            - Compare the retraining accuracy among different epoch sets. Which epoch is the best?
            - Should we prune the weights during the early training stage or the later training stage?
            - Is there a way to automatically identify when to prune?

- **[10% Bouns] Optional Task:**
  We have studied pruning techniques on standard neural networks. How about the robustness of these networks? Will adversarial training help us find better lottery ticket networks? Will the pruned networks have better robustness, in small and large models? Please explore the relationship between pruning and robustness.

    - Reference: https://arxiv.org/abs/2003.05733

    - Deliverables:
        - Experiments to show that pruned networks have better robustness.

- **Additional Reading (Not Included in Scoring):**
  Want a theoretical point of view to understand the pruning? Spline visualization helps! Check out our demo here: https://www.haoranyou.com/spline-eb/. The topic is to investigate the exact visualization and characterization of deep network geometry and decision boundaries as pruning ratios rise.
    - Reference: https://arxiv.org/abs/2302.12828

Please submit a PDF encapsulating all your reading summary, experimental setups, and training trajectories/outcomes, the code can be packed in a separate zip file.