EE P 596 - TinyML - Assignment 1

Total Points: 100
Spring Quarter, 2024
Department of Electrical and Computer Engineering
University of Washington, Seattle, WA 98195

Due: 11:59 pm (PST) on Apr 28 (Sun), 2024 via Canvas

Note:

- This homework contains both programming questions (marked as [**Pro**]) which are required to write Python codes and discussion questions (marked as [**Dis**]) which are required to write answers and provide detailed explanations for your answers.
- You can use and modify the Python functions and codes provided in the Labs or file section of the EEP 596 canvas page when coding the [**Pro**] questions.
- Your answers to this homework must be submitted through ***canvas*** as the following two files: (*i*) a `.ipynb` file containing all Python code written for the programming questions. In case you have multiple `.ipynb` files, you can upload a `.zip` file containing all Jupyter notebooks (*ii*) a `.pdf` writeup file containing handwritten and scanned or typed word answers to all the discussion questions along with the final results that need to be reported for the programming questions.
- Name of your submission files should follow the following format:
“ *#_$_EEP596_HW1.ipynb*” where “#” and “$” should be replaced with your first name and last name, respectively. Use the same format for the `.pdf` writeup file.

In this Assignment you will download and make use of the following dataset and pre-trained model:

- **Plant Leaves Dataset:** This dataset consists of 4,502 images of healthy and unhealthy plant leaves divided into 22 categories by species and state of health. The images are in high resolution JPG format. For more information refer to `https://www.tensorflow.org/datasets/catalog/plant_leaves`. To prepare your dataset for model training and evaluation, use the following guidelines:

  1. Split your dataset into two parts:
     - Training set: 80% of the total data
     - Test set: remaining 20% of the total data
  2. If you need to manually download the dataset, follow these steps:
     - Download the dataset from the specified source.
     - Use the code snippet below to load the dataset into your environment:

       ```
       import tensorflow as tf

       # Specify the path to your manually downloaded and prepared dataset
       dataset_path = 'YourPathHere'

       # Load the dataset from the directory
       train_ds = tf.keras.preprocessing.image_dataset_from_directory(
           dataset_path,
           validation_split=0.2,
           subset="training",
           seed=123,
           image_size=(224, 224),  # Resize images to 224x224 for VGG
           batch_size=32
       )

       test_ds = tf.keras.preprocessing.image_dataset_from_directory(
           dataset_path,
           validation_split=0.2,
           subset="validation",
           seed=123,
           image_size=(224, 224),
           batch_size=32
       )
       ```

- **VGG Pre-trained model:** We will use the pre-trained VGG16 model as the base model and apply **transfer learning** to train the model for our plant leaves classification task. In this approach, we will freeze the weights of the base model and train only the weights in the layers that we add. For more details on the VGG16 base model, please refer to `https://www.tensorflow.org/api_docs/python/tf/keras/applications/VGG16`. Sample code snippet:

  ```
  from tensorflow.keras.applications.vgg16 import VGG16
  from tensorflow.keras.models import Sequential

  base_model = VGG16(weights='imagenet', include_top=False, input_shape=(224, 224, 3))
  base_model.trainable = False  # Freeze the VGG16 layers

  model = Sequential([
      base_model,
      'Define your trainable layers here',
  ])
  ```

1. **[Pro]** (Implementing Base Model on the dataset, 10 pts)

   Implement ***transfer learning*** with the ***VGG16 model*** on the ***Plant Leaves dataset***.

   Denote the trained model as **M** .

   Output the ***training and testing accuracy values***, as well as the ***model size*** of **M** . Also, report these numbers in your `.pdf` writeup file, clearly indicating which values correspond to which model and metric (train accuracy, test accuracy, model size).

2. **[Pro]** (Implementing Quantized Models, 10 pts x 3 models = 30 pts)

   Implement ***Dynamic Range Quantization***, ***Full Integer Quantization***, and ***Float16 Quantization*** on the **M** base model, applying one quantization method at a time.

   Designate the resulting quantized ML models as **M-DRQ** , **M-FIQ** , and **M-F16Q** , respectively.

   Report the ***training and testing accuracy values***, as well as ***the model sizes*** for each of these quantized models in your `.pdf` writeup file.

3. **[Pro]** & **[Dis]** (Implementing and Analyzing Pruned Models, 5 pts x 2 models + 3 pts = 13 pts)

   [Pro] In this task, you will apply ***two different pruning methods*** to the base model **M** using the ***specified pruning schedules***.

   (a) Pruning Schedule 1:

   ```
   import tensorflow_model_optimization as tfmot
   pruning_schedule = tfmot.sparsity.keras.PolynomialDecay(
       initial_sparsity=0,
       final_sparsity=0.5,
       begin_step=0,
       end_step=len(train_ds) * 5)
   ```

   (b) Pruning Schedule 2:

   ```
   import tensorflow_model_optimization as tfmot
   pruning_schedule = tfmot.sparsity.keras.PolynomialDecay(
       initial_sparsity=0.5,
       final_sparsity=0.75,
       begin_step=0,
       end_step=len(train_ds) * 5)
   ```

   Name the pruned ML models as **M-P1** and **M-P2**, corresponding to Pruning Schedule 1 and Pruning Schedule 2, respectively.

   Report the ***training and testing accuracy values***, as well as the ***model sizes*** for the **M-P1** and **M-P2** pruned models in your `.pdf` writeup file.

   [Dis] How might the approach to pruning a neural network ***differ when starting with a initial sparsity = 0 compared to a initial sparsity = 0.5*** ? Consider factors like weight selection and potential ***benefits and drawbacks*** of each scenario.

4. **[Pro]** (Quantization Followed by Pruning, 3 pt x 6 models = 18 pts)

   Implement ***Quantization Followed by Pruning*** for each quantization method on the **M-P1** and **M-P2** pruned models.

   Name the resulting six compressed ML models as **M-P1-DRQ**, **M-P1-FIQ**, **M-P1-F16Q**, **M-P2-DRQ**, **M-P2-FIQ**, and **M-P2-F16Q**.

   Report the ***training and testing accuracy values***, as well as the ***model sizes*** for ***all these compressed models*** in your `.pdf` writeup file.

5. **[Pro]** (Evaluating Latency of the Base Model and Compressed Models, 2 pt x 12 models = 24 pts)

   Write ***a code to evaluate the average and standard deviation of the latency*** for the base model and compressed models discussed in parts 1-4.

   Report the ***average and standard deviation of the latency*** for the models **M**, **M-DRQ**, **M-FIQ**, **M-F16Q**, **M-P1**, **M-P2**, **M-P1-DRQ**, **M-P1-FIQ**, **M-P1-F16Q**, **M-P2-DRQ**, **M-P2-FIQ**, and **M-P2-F16Q** in your `.pdf` writeup file.

6. **[Dis]** (Analysis of the Results, 5 pts) Does ***cascading two model compression techniques*** (e.g., quantization followed by pruning) provide an advantage over using a single technique? ***Justify*** your answer based on the observations from Parts 1-5.