IISER Bhopal

INDIAN INSTITUTE OF SCIENCE EDUCATION AND RESEARCH BHOPAL

Course instructor: **Dr. Jasabanta Patro**
Course: **DSE 418/618: Advance-NLP**
Marks: **20**

Assignment number: **2**
Date: **October 13, 2025**
Date of submission: **November 20, 2025**

Regulations:

- Each student is required to submit solutions based on the specified task.
- Multiple submissions are not allowed.
- Plagiarism: Strictly prohibited. All work should be original. The code will be checked for plagiarism (as well as AI detector) and appropriate action will be taken if found guilty of copying.

Submission Guidelines:

- **Deliverables:** public URL of **(i) code** and **(ii) report**.
  - The Colab notebook should focus on **inference** and **evaluation**. Training code (fine-tuning / QAT) may be included but **commented out**. The notebook must download and load pre-trained weights from your public GitHub repository.
  - Include a **README** with exact commands to reproduce results. The end-to-end runtime of the inference/evaluation (per model variant) should be $\leq$ **15 minutes** on Colab GPU.
- **File naming convention:** `rollno_name_Advance_nlpassignment2.ipynb`
- Submit only the **public URL** of the Colab notebook with clear instructions for running the code.
- Deadline: All assignments must be submitted by the deadline. Late submissions will be penalized.

Marking:

- Marking will be based on (i) **code quality & reproducibility**, and (ii) **model performance** & **analysis**.
- The primary metric is **macro F1-score**; report also accuracy, model size (MB), and inference latency (ms/example or ms/batch).
- All submitted code must be reproducible with public access. If the results cannot be reproduced, the submission will be considered incomplete and not marked.

Assignment Title: BERT Quantization (PTQ & QAT) on `dair-ai/emotion`

Dataset & Task:

- Use the **Emotion** dataset from Hugging Face: https://huggingface.co/datasets/dair-ai/emotion
- Multi-class emotion classification with 6 labels: `sadness`, `joy`, `love`, `anger`, `fear`, `surprise`.
- Use the **split** configuration (train: 16,000; validation: 2,000; test: 2,000). Do **not** use the unsplit (full) version.

Task Overview

1. **Baseline fine-tuning (FP32/FP16):**
   - Load `bert-base-uncased` (or equivalent) and fine-tune on the Emotion dataset.
   - Save trained weights to a **public GitHub repository** and load them in the notebook for inference/evaluation.
2. **Post-Training Quantization (PTQ):**
   - Quantize the **trained** baseline model (*weights and/or activations*) using *post-training* techniques.
   - Evaluate on the test split and report macro F1, accuracy, model size, and latency.
3. **Quantization-Aware Training (QAT):**
   - You may freeze some layers to keep runtime within limits.
   - Evaluate as above and compare against Baseline and PTQ.
4. **Quantized Low Rank Adaptation (QLoRA):**
   - Fine-tune the BERT model using QLoRA. You can use llamafactory framework for this goal.
   - Specify the rank and other hyperparameters used in the task.
5. **Evaluation & Analysis (common to all variants):**
   - Provide a confusion matrix and per-class F1.
   - Report and compare: macro F1, accuracy, **model size (MB)**, and **inference latency**.
   - Summarize trade-offs in a concise table. Provide insights you got from each of these methods.

Implementation Guidelines

- Use Hugging Face Transformers for model/tokenizer loading. Implement PTQ/QAT logic using PyTorch (`torch.{ao.}quantization`) or equivalent primitives; do **not** use pre-made fully quantized model repos.
- Keep the Colab runtime practical:
  - Limit epochs (e.g., 2–3), consider freezing lower layers, and use reasonable batch sizes.
  - You may use FP16 mixed precision for the *baseline* to speed up training; QAT itself should simulate quantization.
- Clearly document:
  - Which layers are quantized (weights/activations), bit-widths, observers/calibration, and any layers left in higher precision (e.g., LayerNorm).
  - Hyperparameters (learning rate, batch size, epochs, max sequence length).
- Ensure the notebook:
  - Downloads weights from your public GitHub and runs **inference + evaluation** without modification.
  - Prints the comparison table (Baseline vs. PTQ vs. QAT) at the end.

Files Provided

1. **Dataset:** https://huggingface.co/datasets/dair-ai/emotion

Deliverables

1. **Python (Colab) notebook** performing inference and evaluation for Baseline, PTQ, and QAT (training code may be present but commented).
2. **Public GitHub repository** with trained weights and instructions.
3. **Report (3–5 pages)** including method description, diagrams (PTQ vs. QAT), results, and a concise discussion of trade-offs.

Marking Scheme (20 Marks)

Baseline fine-tuning & evaluation	4
Post-Training Quantization (implementation & eval)	4
Quantization-Aware Training (implementation & eval)	4
Quantized LoRA (implementation & eval)	5
Code quality, reproducibility, documentation	3