Advanced NLP (M25) - Assignment 2

Deadline: 10th October 2025

Contents

1 Evaluation Metrics

1. For each of the following metrics, explain the metric. Write about the different types within each metric (if they exist) and critically analyze the importance and drawbacks of each.
   (a) ROUGE
   (b) BLEU
   (c) BERT Score
2. What do you know about reference-free evaluations? Give an example of a reference-free evaluation metric. What advantages and disadvantages does it have when compared to the evaluation metrics given above?

Bonus

As you have explored different evaluation metrics in the previous section, come up with your own evaluation metric. Explain it and justify why it is better than the other metrics mentioned in the previous section.

2 Fine-tuning and Quantization

Introduction

As Large Language Models (LLMs) grow, their deployment becomes a significant challenge due to computational and memory constraints. This component focuses on making large models more efficient.

You will start by fine-tuning a large-scale model, GPT-2 Large, for text classification on the AG News dataset. This full-precision model will be your performance baseline. Subsequently, you will apply several Post-Training Quantization (PTQ) techniques to compress this model. Your goal is to rigorously analyze the trade-offs between model size, inference speed, and classification performance.

2.1 Full Fine-tuning & Baseline Performance

Full fine-tuning adapts a pre-trained model to a new task by updating all its weights. For classification, we add a classification head (a linear layer) to map the model's output to the number of classes. The performance of this fine-tuned model establishes the gold standard against which we will measure our compressed models.

2.2 Post-Training Quantization from Scratch

Quantization reduces the numerical precision of model weights, typically from 32-bit floating-point (FP32) to 8-bit integers (INT8). This process involves mapping the wide range of floating-point values to a limited integer range.

Tasks

1. Write Python functions to perform linear PTQ without using a library like bitsandbytes. Your implementation must be able to quantize an FP32 weight tensor to INT8 and de-quantize it back to FP32 for computation.
2. Apply your quantization functions to the baseline model from Part 1.

3. Evaluate this quantized INT8 model on the AG News test set and record all required metrics.

2.3 Library-Based Quantization with bitsandbytes

Libraries like bitsandbytes provide highly optimized and advanced quantization routines. This part explores using such a library for both 8-bit and 4-bit quantization.

Tasks

1. Use the bitsandbytes library to load your fine-tuned model from Part 1 in 8-bit precision.
2. Use bitsandbytes to load the same fine-tuned model using 4-bit NF4 precision.
3. Evaluate both the 8-bit and 4-bit models on the AG News test set. Record all required metrics for each model.

2.4 Analysis, Report, and Submission

Metrics for Evaluation

For each model (Baseline, INT8-Scratch, INT8-bitsandbytes, NF4-bitsandbytes), you must report:

1. **Efficiency Metrics**:
   - **Memory Footprint** (in MB or GB).
   - **Inference Latency** (average time in ms).
2. **Performance Metrics**:
   - **Accuracy**, **Precision**, **Recall**, **F1-Score**.
   - **Confusion Matrix** visualization.

3 Submission

Your submission must be a single `.zip` file containing:

- `report_1.pdf`: A comprehensive report for Task 1.
- `report_2.pdf`: A comprehensive report for Task 2.
- `src/`: A directory with all your Python source code.
- `README.md`: Instructions to run your code, along with links to any large model files.