Assignment 4 Report

Gaurav Bhole 2021113015

1. Introduction: Large Language Models (LLMs) have become increasingly complex and resource-intensive, presenting significant challenges for deployment, particularly on edge devices and resource-constrained environments. Model quantization emerges as a crucial technique to address these challenges by reducing the precision of model weights while attempting to maintain performance. This investigation explores various quantization approaches and their impact on model performance, focusing on both custom implementations and established libraries.

2. Methodology

2.1 Quantization Implementation

Whole Model Quantization:

- Implemented 8-bit quantization across all model parameters
- Utilized symmetric quantization scheme with scale factor calculation
- Applied uniform quantization to maintain weight distribution

Selective Component Quantization:

- Targeted specific model components (attention and FFN weights)
- Implemented for the first 5 layers of the model
- Preserved original precision for non-critical components
- Applied 8-bit quantization to balance accuracy and compression

2.2 Bitsandbytes Integration

8-bit Quantization:

- Linear quantization with dynamic scaling
- Automatic parameter management
- Optimized memory allocation

NF4 (Normalized Float 4) Quantization:

- Non-linear 4-bit quantization
- Weight distribution-aware scaling
- Enhanced precision for critical values

3. Experimental Setup

Model Configuration:

- Base Model: GPT-2 (124M parameters)
- Original Precision: FP32
- Framework: PyTorch with Transformers library

Evaluation Metrics:
1. Model Size (MB)
2. Inference Latency (seconds)
3. Perplexity

Dataset:
- WikiText-2 validation set
- 2461 evaluation samples

4. Results and Analysis

4.1 Custom Quantization Results

Model Size:

Original Model:	474.70 MB
Selective 8-bit:	415.64 MB (-12.4%)
Quantized 8-bit:	119.02 MB (-74.9%)

Inference Latency:

Original Model:	0.467s
Selective 8-bit:	0.459s (-1.9%)
Quantized 8-bit:	0.454s (-2.8%)

Perplexity:

Original Model:	50.90
Selective 8-bit:	51.73 (+1.6%)
Quantized 8-bit:	58.66 (+15.2%)

4.2 Bitsandbytes Results

Model Size:

Original Model:	474.70 MB
Bitsandbytes 8-bit:	156.35 MB (-67.1%)
Bitsandbytes NF4:	115.85 MB (-75.6%)

Inference Latency:

Original Model:	0.16s
Bitsandbytes 8-bit:	0.71s (+357.2%)
Bitsandbytes NF4:	0.27s (+69.9%)

Perplexity:

Original Model:	50.90

Bitsandbytes 8-bit:	51.15 (+0.5%)
Bitsandbytes NF4:	54.45 (+7.0%)

5. Discussion

Key Findings:

1. Size Reduction:
   - Custom 8-bit quantization achieved a 74.9% size reduction
   - Selective quantization provided a moderate 12.4% reduction
   - Bitsandbytes NF4 achieved the highest compression at 75.6%

2. Performance Impact:
   - Custom quantization maintained better latency characteristics
   - Bitsandbytes showed increased latency but better perplexity preservation
   - Selective quantization offered the best balance of size reduction and performance

3. Model Quality:
   - Selective quantization showed minimal perplexity degradation (+1.6%)
   - Bitsandbytes 8-bit maintained best perplexity (+0.5%)
   - Custom 8-bit showed highest perplexity impact (+15.2%)

6. Conclusion

The experimental results demonstrate various trade-offs between model size, inference speed, and quality metrics:

1. Custom Quantization:
   - Offers better latency characteristics
   - Provides more granular control over quantization
   - Shows higher impact on model quality

2. Bitsandbytes:
   - Achieves better compression rates
   - Maintains better perplexity scores
   - Shows increased latency overhead

The choice between approaches depends on specific deployment requirements:
- For maximum compression: Bitsandbytes NF4
- For balanced performance: Selective quantization
- For minimal quality impact: Bitsandbytes 8-bit