LLMs Model Compression: Quantization, Knowledge Distillation & Guardrails - Muskan

Introduction

Large Language Models (LLMs) like GPT, LLaMA, and OPT are powerful but computationally expensive. Efficient deployment on consumer devices, edge environments, or low-power systems necessitates model compression techniques like quantization, pruning, and knowledge distillation (KD). This report covers:

- Quantization and its types
- Knowledge Distillation and variations
- Compression strategies in LLMs (e.g., GPTQ, AWQ, PQK)
- Evaluation metrics for compressed LLMs
- Domain-specific LLM behavior with guardrails
- Comparison of tested small LLMs: gemma3, tinyllama, qwen3:0.6b

Quantization

What is Quantization?

Quantization reduces the precision of model weights or activations (e.g., from 32-bit floats to 4-bit ints), enabling faster and lighter inference.

Types of Quantization:

Method	Description	Pros	Cons
Post-Training Quantization (PTQ)	Applied after full model training	No retraining required	May degrade performance
Quantization-Aware Training (QAT)	Simulates quantization during training	High accuracy retention	Requires full retraining
Mixed-Precision Quantization	Uses lower bits for unimportant weights	Balance between size and performance	Harder to tune manually

LLM-Specific Techniques

- **GPTQ:** Generalized Post-training 4-bit quantization with negligible accuracy loss using second-order information.
- **AWQ:** Activation-aware weight quantization; prioritizes critical weights (~1%) and improves real-time inference.

- **BiLLM, SpQR:** Use 1-2 bits, explore ternary or binary quantization while attempting to preserve core semantics.

How Low Can You Go?

- 8-bit: Safe, minimal performance drop.
- 4-bit: Effective with techniques like GPTQ/AWQ.
- ≤3-bit: Risk of quality degradation increases sharply, especially on long-context or complex reasoning tasks.

Knowledge Distillation

What is Knowledge Distillation?

KD trains a smaller student model to learn from a larger teacher by mimicking its behavior (logits, layers, or internal states).

Types of KD:

- **Logit-based:** Student matches softmax outputs of the teacher.
- **Layer-wise (intermediate):** Student aligns its hidden layers with teacher.
- **Self-distillation:** Teacher is derived from the student itself (e.g., earlier checkpoints).

KD in LLMs

Teacher → Student	Method	Params Reduction	Performance Retention
GPT-3 → DistilGPT	Logit KD	-40%	~95% accuracy
BERT → DistilBERT	Logit + Layer	110M → 66M	~97% GLUE
LLaMA → TinyLLaMA	Custom KD	13B → <2B	90–95% PPL match

Integrated Compression: The PQK Pipeline

What is PQK?

PQK = Pruning + Quantization-Aware Training + Knowledge Distillation

- Phase 1: Prune redundant parameters → Apply QAT
- Phase 2: Discarded weights are used as internal teacher for KD
  → Efficient, unified training loop

Benefits

- No external teacher required
- Maintains high accuracy with smaller size
- Suited for LLMs (prune attention heads, FFN layers, etc.)

Evaluation Metrics for LLM Compression

Performance Metrics:

Task	Metric	Usage
Language Modeling	Perplexity	Lower = better predictions
Classification	Accuracy, F1	For intent, NLU
QA / Summarization	Exact Match, BLEU, ROUGE, BLEURT	Measures generation quality
Reasoning / Bias	TruthfulQA, BBQ, MMLU	Evaluates ethical alignment, truthfulness

Efficiency Metrics:

Metric	Description
Model size	Parameters or storage (MB/GB)
Latency	Time to respond (ms or s)
Throughput	Tokens/sec
Energy consumption	Power needed for inference

Trade-off: Lower memory use vs. generation quality. E.g., 4-bit GPTQ = ~4× smaller, <1% accuracy drop.

LLM Guardrails

What are Guardrails?

Guardrails are behavioral boundaries or content filters set for LLMs to ensure:

- Domain specificity
- Ethical compliance (e.g., no misinformation or harmful content)
- Safety (medical, legal, etc.)
- Politeness or refusal for out-of-domain questions

Types:

Guardrail Type	**Description**
Domain filtering	Only allow questions from selected domain (e.g., medical, IT)
Output sanitization	Prevent toxic, false, or unsafe replies
Context length enforcement	Ensure responses don't exceed memory limits
Structured reasoning	Use formatted, stepwise outputs

My Code Implementation:

- Uses structured prompts per model tier
- Explicit domain adherence
- "Decline if out-of-domain" built-in
- Tiered guardrails: TinyLLaMA (light), Gemma3 (rigorous)

Model Comparison Based on Testing

Prompt: *"What are the 5 most famous paintings by Vincent Van Gogh?"*

Model	Output Quality	Guardrail Adherence	Structured Reasoning	Hallucinations
Gemma3	Excellent	Yes	Stepwise: Clarify → Answer → Summary	No
TinyLLaMA	Fair	Yes	Brief and inconsistent	Minor
Qwen3:0.6b	Poor	Confused	Repetition, no clarity	Yes (loop hallucination of "Starry Night")

Conclusion:

- Gemma3 shows strong reasoning and formatting when proper guardrails are applied.
- TinyLLaMA performs decently for simple factual queries but lacks coherence.
- Qwen3 lacks prompt-following ability, struggles with hallucination loops.

Summary Table

Technique	Advantages	Limitations
8-bit Quantization	High accuracy, easy deployment	Limited compression
4-bit GPTQ/AWQ	~4× smaller, fast inference	Requires calibration
Pruning + QAT	Efficient and accurate	Complex training
Knowledge Distillation	Preserves logic and tone	Needs teacher model
PQK Pipeline	Unified + teacher-free	Training complexity

Final Insight

A modern LLM compression strategy should follow this pipeline:

1. Prune redundant parts (e.g., attention heads)
2. Train with QAT to simulate quantization
3. Use KD (internally via PQK or externally)
4. Apply GPTQ or AWQ for final quantization
5. Evaluate using perplexity, BLEU, latency, alignment, and bias benchmarks

References

- Buciluǎ et al., “Model Compression…”, KDD 2006
- Kim et al., “PQK: Model Compression…”, arXiv 2021
- Frantar et al., “GPTQ”, arXiv 2022
- Lin et al., “AWQ”, MLSys 2024
- Huang et al., “BiLLM: 1-bit LLMs”, arXiv 2024
- Mekala et al., “Quantization and Long-Context”, arXiv 2025
- HuggingFace, Towards AI, Hacker News, GitHub Discussions