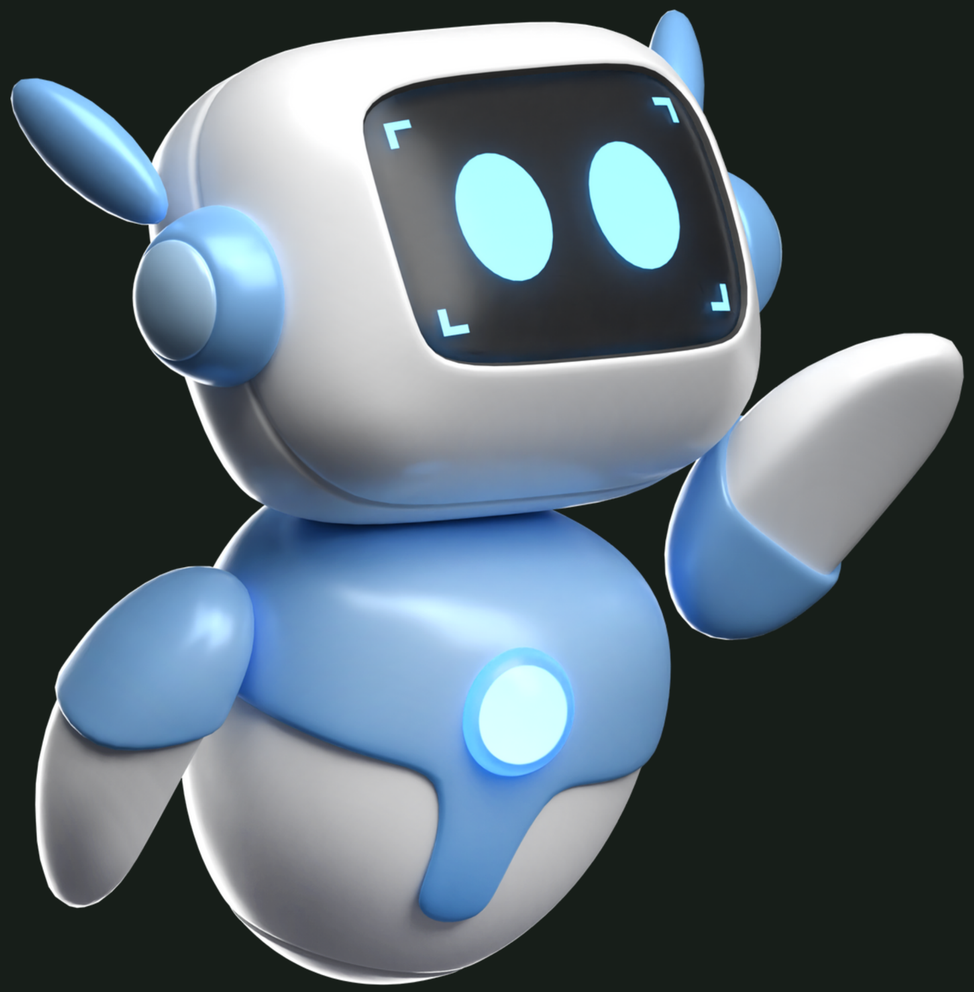


LITERATURE REVIEW

DEEP LEARNING

M. HASEEB - 26100253
& HAMZA IQBAL - 26100130

PROBLEM DEFINITION

Language Models on Edge Devices

1. Limited compute and memory on edge devices lead to unacceptable latency and energy consumption for unoptimized SLMs.
2. Although compression techniques are crucial for edge deployment, they often introduce accuracy degradation. Therefore, the core problem is to optimize SLMs for edge devices to achieve acceptable performance (latency, energy efficiency) without sacrificing critical accuracy for practical applications.

INHERENT GOALS

01 PERFORMANCE

To compare the effectiveness of compression techniques like quantization and distillation and fine-tuning techniques like LoRA on small models evaluating their impact on accuracy, latency, and model size. The goal is to identify optimal compression strategies for resource-constrained edge environments, including testing of Quantization aware KD.

---

02 EDGE-DEPLOYMENT

To demonstrate practical edge AI, this project will deploy optimized models on specific edge devices like Jetson Nano and Android phones. We will evaluate performance for representative edge tasks such as on-device object detection.

LITERATURE REVIEW

EMPIRICAL GUIDELINES FOR DEPLOYING LLMS ONTO RESOURCE-CONSTRAINED EDGE DEVICES

(Qin et al., 2024)

COMPRESSION INSIGHTS

- Phi Family with LoRA Offer robust performance across classification and summarization tasks; a safe default for uncertain task difficulty.
- Knowledge Distilled Models such as Phi models provide consistent, safe performance for variable downstream tasks.
- Post training quantized models can achieve higher peak performance with specific settings but show less stability overall.
- Quantized vs. Unquantized; Quantized models might perform better with limited fine-tuning data.
- Edge Adaptation: Smaller, compressed models can outperform larger, uncompressed models (e.g., Llama 3-8B) for rapid adaptation to limited user data.
- Pruning gave poor results compared to the other techniques

QUANTIZATION TECHNIQUES

ZHU ET AL. (2024)

Quantization-Aware Training (QAT)

The model is retrained while incorporating the lower-precision format. This “teaches” the model to overcome errors introduced by using fewer bits.

Post-Training Quantization (PTQ)

You take a fully trained (full-precision) model and convert it to a low-precision version without retraining. This saves time and computational resources but may have different trade-offs in accuracy.

Weight-Only Quantization:

- Map weights to discrete levels (scaling + rounding); protect sensitive weights via Hessian-based error minimization (e.g., GPTQ)
- Requires dequantization during compute; limited gains if activations remain high-precision.

Weight-Activation Quantization:

- Scale/round both; outlier handling via LLM.int8() (high-precision outliers) or SmoothQuant (per-channel scaling).
- Sensitive to activation variability; needs robust outlier management.

KV-Cache Quantization:

- Keys: Quantized per channel. Values: Quantized per token.
- KVQuant (non-uniform scaling), WKVQuant (2D + regularization).

POST TRAINING QUANTIZATION

GPTQ: ADVANCED POST-TRAINING QUANTIZATION FOR LLMS

FRANTAR ET AL., (2023)

Overview:

- *• Converts full-precision LLMs to low-bit (3-4 bits) models without retraining*
- *• Targets weight-only quantization for efficient inference and reduced memory footprint*
- *• Uses Hessian-based sensitivity estimation (diagonal or block-diagonal approximations) to identify critical weights for selective precision preservation*

Layer-Wise Quantization Process:

- *• Processes each layer independently using iterative optimization to minimize Hessian-weighted error*
- *• Computes optimal scaling factors and thresholds to map continuous weights into discrete values*
- *• Iterative refinement minimizes performance degradation through repeated adjustments*
- *• Achieves low bit-width performance while maintaining model representational capacity*

Advantages:

- *• No retraining required—applies directly to pre-trained models*
- *• Leverages second-order information to minimize accuracy loss*
- *• Scalable approach suitable for large LLMs through structured, layer-wise quantization*

Loss Function

- Quantization error is measured as:
  $\text{Error} \approx (W - Q)^T H (W - Q)$
  where W is the original weight vector, Q the quantized vector, and H the approximated Hessian.

PD-Quant (Liu et al., 2023):

- Uses a prediction difference loss to optimize activation scaling factors by aligning quantized model outputs with full-precision predictions.

MobileQuant (Wu et al., 2023):

- Employs weight equivalent transformations and jointly optimizes per-tensor and per-channel quantization ranges, targeting reduced latency and energy consumption for mobile deployments.

Smooth Quant (Xiao et al., 2022)

- Implements techniques to ensure smoother transitions in quantized values, minimizing quantization errors and further preserving model performance.
  -

QUANTIZATION-AWARE TRAINING

XUNYU ZHU ET AL .

In-Training Simulation:

- "Fake quantization" applies low-precision arithmetic during forward/backward passes
- Self-distillation: Full-precision model guides quantized training
- Benefits: Smoother transition and minimized accuracy loss
- Trade-Off: Increased training complexity and careful calibration

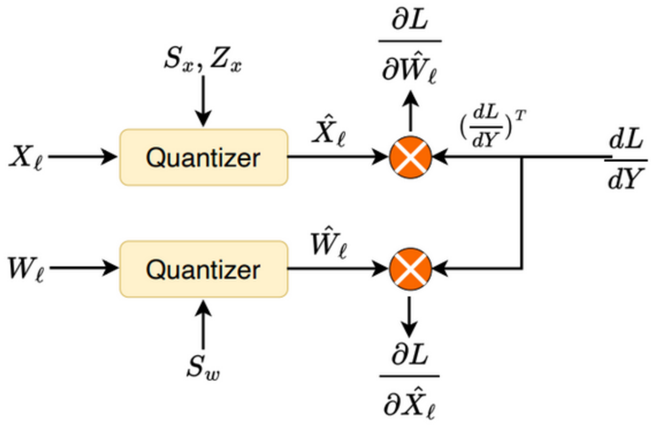
Selective Parameter Update Framework (EFQAT):

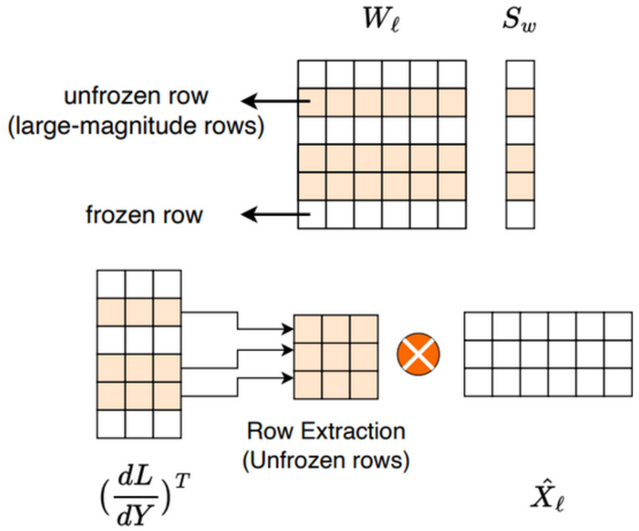
- Start with a PTQ model and update only critical parameters (using weight importance metrics)
- Freeze less important weights (structured freezing with CWPL, CWPN, LWPN modes)

Combined QAT & LoRA (QLoRA):

- QLoRA drastically reduces memory footprint by quantizing the pre-trained language model weights to 4-bit precision.
- Instead of fine-tuning all model parameters, QLoRA introduces small, trainable low-rank adapter layers. These adapters learn task-specific information while the large, quantized base model remains frozen

Comparison. QAT (left) and EFQAT(right_





IN QAT, THE BACKWARD PASS PERFORMS FULL-PRECISION MATRIX MULTIPLICATIONS WITH SYMMETRIC WEIGHTS (SCALE SW) AND ASYMMETRIC INPUTS (SCALE SX, ZERO POINT ZX), WHEREAS EFQAT SPEEDS UP THE PROCESS BY MULTIPLYING ONLY THE MOST IMPORTANT, UNFROZEN ROWS WITH HIGH AVERAGE MAGNITUDE.

KNOWLEDGE DISTILLATION

(XU ET AL., 2024).

Core Concept:

- Transfers knowledge from a larger teacher model to a smaller student model, enabling near-teacher performance.
- Balances model size and performance through careful design of the knowledge transfer process.

Loss Functions & Techniques:

- KL Divergence Loss.
- Minimizes differences between softened probability distributions (Hinton et al., 2015).

Supervised Fine-Tuning (SFT):

- Uses cross-entropy loss to align student outputs with ground-truth labels.
- Employs reverse KL and Jensen–Shannon divergence to further reconcile output distributions (Xu et al., 2024).

Similarity-Based Methods:

- Applies L2-norm and cosine similarity losses to align intermediate feature representations

KNOWLEDGE DISTILLATION

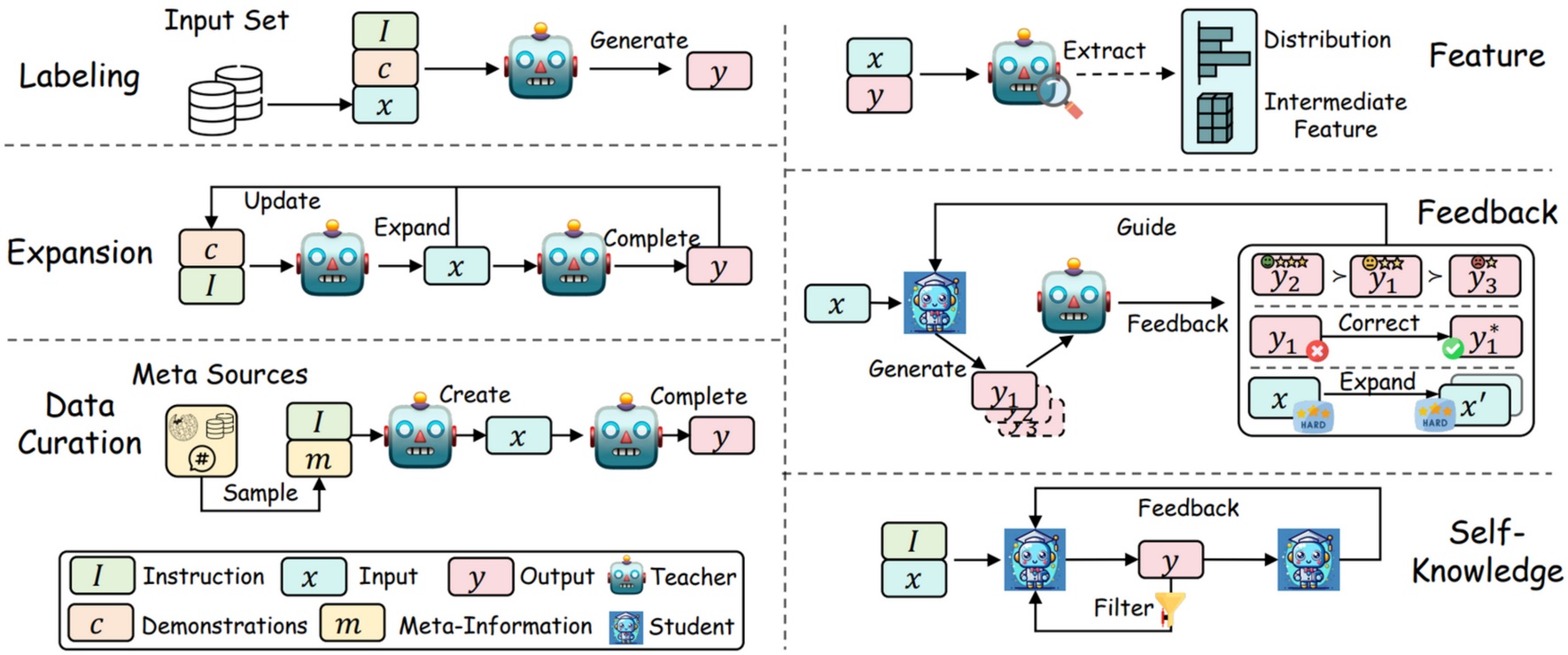


FIGURE ILLUSTRATES VARIOUS TEACHER LLM KNOWLEDGE ELICITATION METHODS: LABELING (DIRECT OUTPUT GENERATION), EXPANSION (IN-CONTEXT SAMPLE GENERATION), DATA CURATION (SYNTHESIZING DATA FROM META-INFORMATION), FEATURE (EXTRACTING INTERNAL KNOWLEDGE LIKE LOGITS), FEEDBACK (PROVIDING CORRECTIONS AND PREFERENCES), AND SELF-KNOWLEDGE (STUDENT-GENERATED OUTPUTS FILTERED FOR QUALITY).