RL-Based Dynamic Quantization for Large Language Models

Oleg Roshka [1] **Ilia Badanin** [2]

Abstract

This milestone report presents progress on a dynamic quantization framework for large language models (LLMs). We employ reinforcement learning (RL) to adaptively choose per-layer quantization schemes (e.g., 4-bit or 8-bit) during fine-tuning. Experiments suggest the learned policy can preserve model performance while significantly reducing memory usage. We detail our reward design, preliminary results, and outline the next steps.

1. Introduction

Large language models (LLMs) can be prohibitively large for deployment in memory-constrained settings. Quantization—lowering numerical precision—offers a direct way to reduce model size and speed up inference. However, most solutions apply a *uniform* bit-width (e.g. int8) across all layers (Dettmers et al., 2022; Sun et al., 2023), which can lead either to wasted capacity if a layer could be safely quantized more aggressively, or to degraded performance if it is overly compressed.

To address these issues, we propose a *Reinforcement Learning (RL)* approach that selects each layer's quantization type (e.g., `nf4`, `fp4`, `int8`, or `fp16`) *dynamically* while fine-tuning on a small dataset. Our environment rewards the agent for maintaining model accuracy, preserving attention entropy, and maximizing memory savings. This method extends ideas from prior RL-based neural architecture or policy searches (Zoph & Le, 2017; Wang et al., 2020) and complements extreme LLM compression efforts (Egiazarian et al., 2024; Malinovskii et al., 2024).

[1]Department of Computer Science, Stanford University, Stanford, CA, USA [2]École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland. Correspondence to: Oleg Roshka <oros@stanford.edu>, Ilia Badanin <ilia.badadnin@epfl.ch>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

2. Related Work

3. Methodology

3.1. Reference Model Fine-Tuning

First, we fine-tune GPT-2 on a reasoning dataset (e.g. CommonsenseQA). This fully fine-tuned GPT-2 (in FP32 or FP16) acts as our *reference model*, providing L_{ref} (the reference loss) and p_{ref} (the reference distribution). This is similar to SFT (supervised fine-tuning) used for teacher logits in knowledge-distillation or reward-model contexts (Sun et al., 2023).

3.2. Reinforcement Learning Environment

We clone the reference model weights into a *quantized model*, which is adjusted layer-by-layer via RL. Each *episode* involves:

1. Iterating through all N layers of the model in sequence.
2. For each layer, the RL agent chooses a quantization scheme (one of {`nf4`, `fp4`, `int8`, `fp16`}).
3. We apply that choice, then do a small amount of fine-tuning on a training minibatch.
4. We measure partial validation loss and other metrics (KL, attention entropy) on the updated model.

At the end of the episode, the environment computes a cumulative *reward* capturing how well the chosen quantization policy balances performance and memory savings.

3.3. Reward Function

We define our reward R as a sum of several terms, each corresponding to a different aspect of quality and efficiency in our quantized model. Specifically, we use:

$$\begin{aligned} R \;=\; & w_{\text{perf}} \left(L_{\text{ref}} - L_{\text{quant}}\right) \\ & - \; w_{\text{KL}}\, \text{KL}\big(p_{\text{quant}} \| \, p_{\text{ref}}\big) \\ & + \; w_{\text{entropy}} \left(E_{\text{quant}} - E_{\text{ref}}\right) \\ & + \; w_{\text{memory}}\, \text{MemSave}. \end{aligned} \tag{1}$$

Interpretation.

- L_{ref} and L_{quant} are the reference vs. quantized validation losses.
- $\text{KL}(p_{\text{quant}} \| p_{\text{ref}})$ encourages distribution alignment to the teacher (reference).
- $E_{\text{quant}}, E_{\text{ref}}$ measure attention entropies, preserving information flow (Choi et al., 2018).
- MemSave tracks fraction of bits saved, inspired by prior quantization frameworks (Jacob et al., 2018).

By combining these signals, the agent is simultaneously encouraged to match or improve upon the reference model's predictive capabilities, avoid diverging too far in probability space, retain higher attention diversity, and reduce model size.

4. Implementation Details

Model & Baseline. We currently focus on GPT-2 as a testbed. The reference model is the fully fine-tuned GPT-2 (e.g., FP16). A uniform 8-bit quantization serves as one baseline.

RL Algorithm (PPO). We employ Proximal Policy Optimization (Schulman et al., 2017) for stability. The RL *state* includes:

- The current layer index
- Memory usage so far
- Historical bit choices for previous layers
- Running losses/KL vs. reference

The policy outputs a discrete action (which quantization format to apply).

5. Experiments and Results

5.1. Experimental Setup

5.2. Results

Reward Over Episodes. Figure 1 shows total reward over 500 episodes. We see an upward trend from about 4–5 to over 9, suggesting the agent learns progressively better quantization decisions.

Loss Metrics. Figure 2 shows the final validation loss and the policy loss. The quantized model's validation loss remains close to that of the reference model, while the RL policy loss stabilizes near zero, indicating convergence.

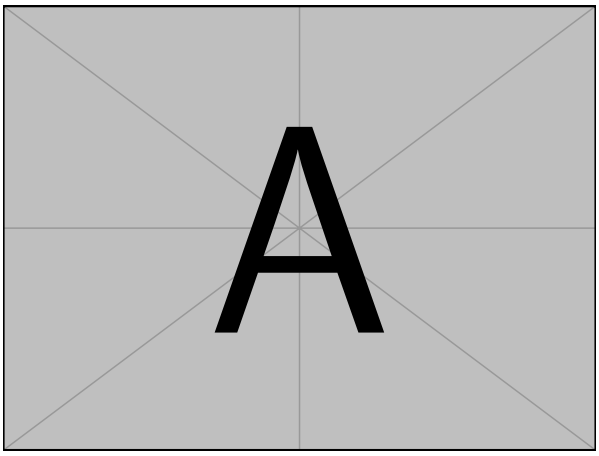

Figure 1. Total RL reward vs. episode. The upward trajectory indicates improved policies balancing performance, KL, attention entropy, and memory savings.

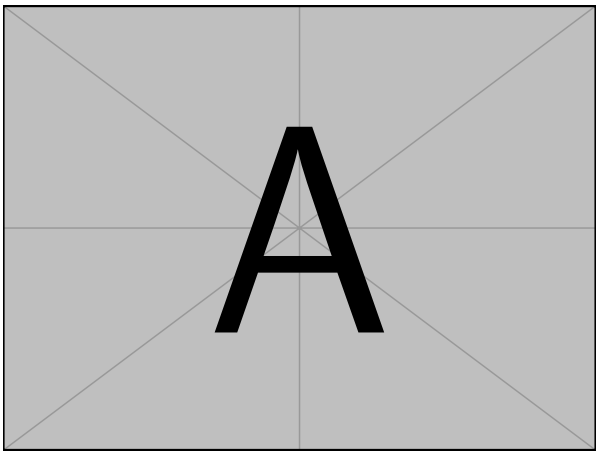

Figure 2. Validation loss (blue) vs. reference loss (green), and the PPO policy loss (orange). The quantized model stays near reference-level performance.

Quantization Distribution. Finally, Figure 3 shows how often each format is selected across episodes/layers. We see `fp4` and `nf4` are chosen most frequently, with smaller usage of `int8` and `fp16`. This implies the agent often prefers ultra-low precision for memory gains, without an excessive performance penalty.

6. Discussion

6.1. Limitations

6.2. Future Work

Hyperparameter Tuning. We will tweak $(w_{\text{perf}}, w_{\text{KL}}, w_{\text{entropy}}, w_{\text{memory}})$ to find stable regimes. Too large w_{memory} may push the policy to choose 4-bit everywhere, risking a performance drop.

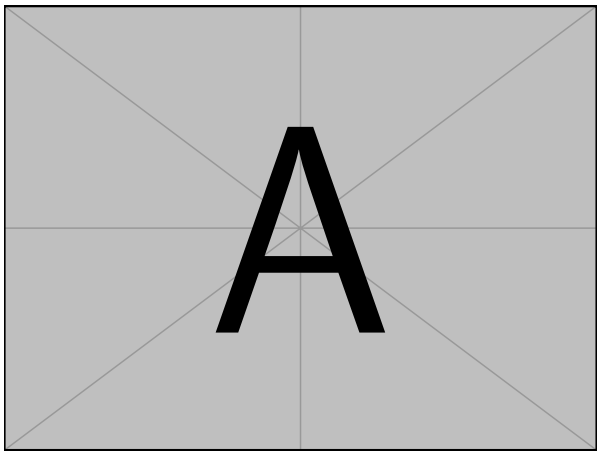

Figure 3. Histogram of chosen quantization types across all layers/episodes. 4-bit (`fp4`, `nf4`) is dominant.

Scaling Up. We plan to extend from GPT-2 to bigger open-source models, such as `deepseek-ai/DeepSeek-R1-Distill-Qwen-1.5B` or `microsoft/phi-2`, to test whether the method scales effectively on more challenging datasets. We also aim to run more comprehensive evaluations to compare standard quantized baselines against our dynamic approach (time and compute permitting).

Further Evaluation. We aim to measure:

- **QA Accuracy** on more challenging datasets/benchmarks
- **Memory usage** and **Inference latency**
- **Attention entropy** across deeper layers

7. Conclusion

We have shown an RL-driven approach to dynamic per-layer quantization that preserves near-reference performance while using predominantly 4-bit formats. Our multi-component reward (Equation 1) guides the agent to choose lower precision whenever feasible, balancing KL, entropy, and memory. Ongoing tasks include hyperparameter tuning, scaling to larger GPT models, and extended evaluations on QA tasks.

Appendix

References

Choi, J., Wang, Z., Venkataramani, S., Najafirad, P., Shenoy, V., and Keutzer, K. PACT: Parameterized Clipping Activation for Quantized Neural Networks, 2018.

Dettmers, T., Lewis, M., Belkada, Y., and Zettlemoyer, L. LLM.int8(): 8-bit Matrix Multiplication for Transformers at Scale, 2022.

Egiazarian, V., Panferov, A., Kuznedelev, D., Frantar, E., Babenko, A., and Alistarh, D. Extreme compression of large language models via additive quantization, 2024.

Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H., and Kalenichenko, D. Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference, 2018.

Malinovskii, V., Mazur, D., Ilin, I., Kuznedelev, D., Burlachenko, K., Yi, K., Alistarh, D., and Richtarik, P. Pv-tuning: Beyond straight-through estimation for extreme llm compression, 2024.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Sun, Z., Saab, K., Tepper, M., Dettmers, T., Basu, S., Gonzalez, J. D. M., Shen, S., Rush, A. M., Duckworth, D., Lewis, M., et al. Qlora: Efficient finetuning of quantized llms, 2023.

Wang, T., Wang, K., Cai, H., Lin, J., Liu, Z., Wang, H., Lin, Y., and Han, S. Apq: Joint search for network architecture, pruning and quantization policy, 2020.

Zoph, B. and Le, Q. V. Neural Architecture Search with Reinforcement Learning, 2017.