Efficient LLMs via Switchable and Dynamic Quantization

Akhil Nampally

September 28, 2025

1 Introduction

Large language models (LLMs) have achieved strong results across many natural language processing tasks, but their size makes them expensive to run and difficult to deploy in practice. Quantization is one of the most effective ways to reduce this cost because it lowers the number of bits used to represent the weights and activations in the network. This comes at the price of reduced precision, and the challenge is to find ways to use quantization without losing too much accuracy. Recent research has shown that dynamic and switchable quantization strategies can balance this trade-off and can even improve robustness. Examples include Double-Win Quant [2], LLM-QAT [4], and Cyclic Precision Training (CPT) [1].

Low-rank adaptation methods such as LoRA [3] provide another tool for efficient fine-tuning. Instead of retraining all of a large model's parameters, LoRA introduces small low-rank matrices that can be trained quickly and cheaply. Combining LoRA with quantization creates the possibility of training quantized models flexibly without large computational costs.

In this project, we applied these ideas to GPT-2 and evaluated their effects using the SQuAD dataset. The training budget was only 1,000 steps, which meant that absolute accuracy would be very low. The focus was not on achieving strong performance but on comparing relative behavior across quantization profiles, testing cyclic precision training, and examining robustness under adversarial input perturbations.

2 Methods

2.1 Step 1: Quantization

We first modified GPT-2 by adding custom wrappers called `QuantLinear` and `QuantConv1D`. These wrappers allowed each layer in the network to run at a specific bit-width. Using them, we defined three precision profiles: `bw4` for 4-bit quantization, `bw6` for 6-bit quantization, and `bw8` for 8-bit quantization. By switching between these profiles, we could control the level of quantization applied to the model.

2.2 Step 2: LoRA Modules

Next, we added LoRA adapters [3] to every quantized layer. A routing mechanism was used so that the correct adapter was activated based on which precision profile the model was using at the time. This allowed the model to learn different low-rank adaptations for the different bit-widths. This design was important because it kept the number of trainable parameters small, making it possible to train under the strict 1,000-step limit.

2.3 Step 3: Joint Training

The model was trained on the SQuAD dataset for 1,000 steps. During training, each batch was randomly assigned one of the three quantization profiles. This meant that the model was exposed to different levels of precision and learned to adapt to all of them. The goal was not to make the model accurate in absolute terms but to study the differences between profiles and training methods under the same conditions.

3 Results

3.1 Step 4: Quantization Profiles

We first compared the three quantization profiles after 1,000 steps of training. Exact Match (EM) remained zero for all profiles, which was expected because the model had very limited time to learn to produce exact span predictions. F1, which gives partial credit for overlapping tokens, differed across profiles. The results are shown in Table 1.

We expected that 6-bit precision might be the best because prior work has shown that it can provide a balance between stability and beneficial noise regularization. However, in our run the 8-bit profile performed best,

Profile	EM	F1
`bw4`	0.0	0.037
`bw6`	0.0	0.023
`bw8`	0.0	0.043

Table 1: Validation accuracy (SQuAD) across quantization profiles after 1,000 steps. `bw8` achieved the highest F1 score.

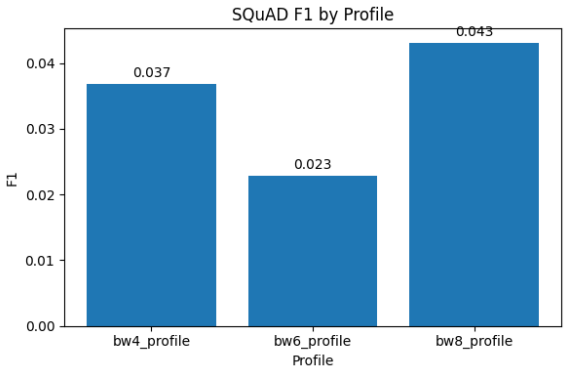

Figure 1: F1 scores on SQuAD across quantization profiles after 1,000 steps. The 8-bit profile outperformed both 4-bit and 6-bit profiles.

followed by 4-bit, with 6-bit performing worst. This suggests that under the limited training budget of 1,000 steps, the model benefited more from the stability of higher precision.

3.2 Step 5: Cyclic Precision Training

We then implemented cyclic precision training (CPT) [1], where the model cycles through different quantization profiles in a fixed order instead of sampling them randomly. Prior work with convolutional networks suggested this could provide better convergence by exposing the model to structured precision noise.

To fairly compare the two approaches, we reinitialized the model separately for each run and used the same training budget (1,000 steps) and learning rate. Figure 2 shows the training losses for both strategies after smoothing with a moving average. The two curves are almost indistinguishable, indicating that CPT did not provide any measurable benefit over standard switchable training under these conditions.

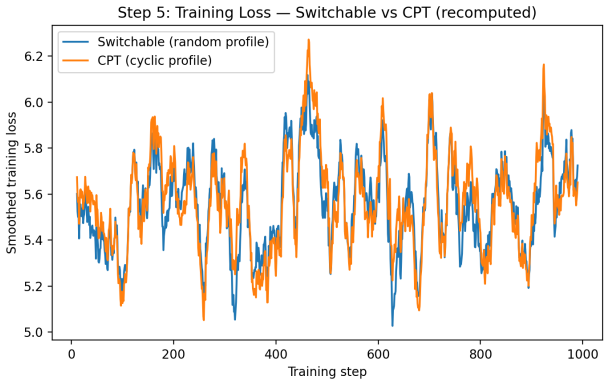

Figure 2: Training loss under cyclic precision training compared to standard switchable precision. Both curves were recomputed using fresh model initializations with the same setup. The two strategies produced nearly identical losses, showing no CPT advantage under a 1,000-step budget.

These results suggest that the improvements reported for CPT in convolutional networks do not directly transfer to transformer models in the low-budget regime. Transformers may require longer training to adapt to the structured noise introduced by CPT, or may simply benefit less from it.

3.3 Step 6: Robustness Under Adversarial Attacks

Finally, we tested robustness using adversarial character-level attacks where random typos were added to the input text. These small perturbations are known to reduce performance in NLP models [5]. We compared three strategies: keeping a fixed profile (`bw6`), switching the entire profile randomly at each step, and switching only a subset of layers randomly. Results are shown in Table 2.

Strategy	EM	F1
Fixed (bw6)	0.0	0.025
Random-profile	0.0	0.044
Random-layer	0.0	0.005

Table 2: Adversarial robustness under typo attacks. Random-profile switching gave the best robustness, while random-layer switching performed the worst.

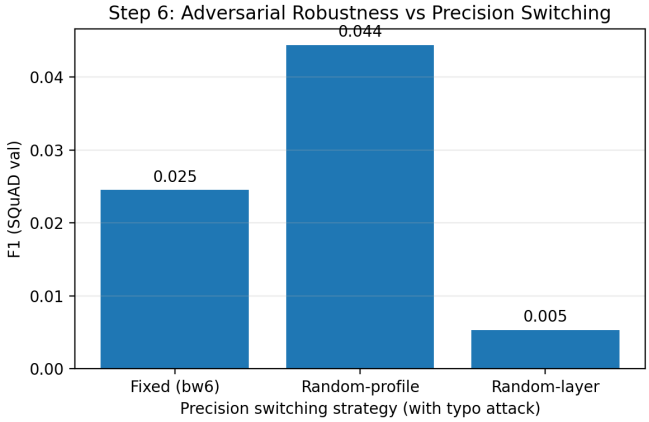

Figure 3: Adversarial robustness (F1) under fixed, random-profile, and random-layer switching. Random-profile switching produced the highest F1, while random-layer switching reduced robustness.

The results show that random-profile switching improved robustness, while random-layer switching hurt it. This indicates that adding randomness at the profile level introduces useful variation, but disrupting only some layers introduces too much instability.

4 Discussion

The experiments lead to several conclusions:

- Quantization reduces accuracy as precision decreases, but in our runs the 8-bit profile gave the best results, contrary to the expectation that 6-bit would strike the best balance.
- Cyclic precision training did not help in this setting, suggesting that results from CNNs may not apply directly to transformers.
- LoRA adapters made it possible to train profile-specific adaptations within the strict step limit, showing their importance in practical quantized training.
- Random-profile switching improved robustness to typos, while random-layer switching reduced it, showing that the level of randomness matters for resilience.

5 Conclusion

This project demonstrated the feasibility of applying switchable and dynamic quantization to GPT-2 under a constrained fine-tuning budget. While absolute accuracy remained low due to the 1,000-step limit, the relative comparisons revealed useful insights. First, although prior work suggested that mid-bit precision (6-bit) would often balance quantization error and regularization noise, in our experiments `bw8` achieved the highest F1 while `bw6` lagged behind. We nevertheless used `bw6` as the fixed baseline for adversarial robustness evaluations in Step 6, since it represents the intermediate precision profile that theory predicts should be competitive, even if it did not dominate in our run.

Second, cyclic precision training (CPT) did not show any measurable benefit compared to standard switchable training when recomputed fairly, with both approaches producing nearly identical loss curves. This suggests that CPT's reported benefits in convolutional networks do not directly transfer to transformer models in a short training horizon.

Finally, adversarial robustness experiments showed that switching entire profiles at random improved resilience to typo attacks, while random-layer switching degraded it. This indicates that controlled randomness at the profile level can be beneficial, whereas unstructured randomness across layers may be too disruptive.

Taken together, these findings show that carefully designed quantization strategies, paired with lightweight LoRA adapters, can provide efficiency and robustness insights even under tight training budgets. They also highlight the gap between theoretical expectations (such as the supposed advantage of 6-bit) and empirical outcomes in practice. Future work with longer training runs and multiple seeds would help clarify these discrepancies.

References

[1] Yonggan Fu, Han Guo, Xin Yang, Yining Ding, Yingyan Lin, Meng Li, and Vikas Chandra. Cpt: Efficient deep neural network training via cyclic precision. In *International Conference on Learning Representations*, 2021. arXiv:2101.09868.

[2] Yonggan Fu, Qixuan Yu, Meng Li, Vikas Chandra, and Yingyan Lin. Double-win quant: Aggressively winning robustness of quantized deep neural networks via random precision training and inference. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 3492–3503. PMLR, 2021.

[3] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

[4] Zechun Liu, Barlas Oğuz, Changsheng Zhao, Ernie Chang, Pierre Stock, Yashar Mehdad, Yangyang Shi, Raghuraman Krishnamoorthi, and Vikas Chandra. Llm-qat: Data-free quantization aware training for large language models. *arXiv preprint arXiv:2305.17888*, 2023.

[5] Zimu Wang, Wei Wang, Qi Chen, Qiufeng Wang, and Anh Nguyen. Generating valid and natural adversarial examples with large language models. *arXiv preprint arXiv:2311.11861*, 2023.