Low-Bit Quantized nanoGPT Speedrun: Testing the Effects of Weight and Activation Quantization

Travis Hammond
Whiting School of Engineering
Johns Hopkins University
Baltimore, USA
thammo19@jhu.edu

November 19, 2025

Abstract

The scaling of Large Language Models (LLMs) is increasingly bottlenecked by memory bandwidth and energy consumption. The emerging 1.58-bit (ternary) quantization paradigm offers a potential solution by constraining weights to $\{-1, 0, 1\}$, thereby enabling matrix multiplication to be approximated by integer addition. This research implements and evaluates a low-bit quantized GPT architecture from scratch, based on the `modded-nanogpt` codebase. We conduct a comparative study on the FineWeb-10B dataset to isolate the impact of quantizing different model components. Specifically, we examine three configurations: (1) Ternary Weights only ($W_{1.58}A_{BF16}$), (2) Ternary Weights with Quantized Attention ($W_{1.58}A_{QK}$), and (3) Fully Quantized Weights and Activations ($W_{1.58}A_{1.58}$). Our results demonstrate that weight-only ternary quantization provides a strong balance between performance and efficiency, achieving a validation loss of 3.41 (comparable to the 3.28 baseline). Conversely, we find that aggressive quantization of activations to ternary values without specific architectural adaptation leads to significant performance degradation (Loss 3.87).

1 Introduction

As Large Language Models (LLMs) grow in size, the computational cost of training and inference has become a critical barrier to deployment. Traditional 16-bit (BF16/FP16) operations dominate memory traffic, creating a "memory wall" where compute units idle while waiting for data. Quantization-Aware Training (QAT) addresses this by simulating low-precision arithmetic during the training phase, allowing the network to adapt to discretization noise.

Recent foundational work, such as BitNet b1.58 [1], suggests that extreme quantization to ternary values $\{-1, 0, 1\}$ is feasible. This "1.58-bit" representation allows for the replacement of heavy floating-point Multi-Accumulate (MAC) operations with lighter integer addition and subtraction.

In this work, we investigate the practical trade-offs of this paradigm by implementing a quantized GPT-2 model from scratch via a speedrun training regime. Our primary objective is to decouple the effects of weight quantization from activation quantization. We hypothesize that while full quantization offers maximum theoretical speedups, **weight-only ternary quantization** provides the majority of memory benefits with significantly higher training stability.

2 Background

2.1 1.58-bit Quantization

The 1.58-bit scheme constrains parameters to three possible states: $\{-1, 0, 1\}$. This reduces the memory requirement per parameter to $\log_2(3) \approx 1.58$ bits.

- **Memory Efficiency:** Storing weights in ternary format drastically reduces the model footprint compared to FP16 or INT8.
- **Computational Efficiency:** Matrix multiplication $Y = XW$ becomes a sparse addition process.

2.2 Smooth Gradient Approximation

Training discrete variables prohibits standard backpropagation because the rounding function has a derivative of zero almost everywhere. While the Straight-Through Estimator (STE) is a common solution, it often leads to gradient mismatch. Instead, we adopt a smooth gradient approximation for the backward pass [2]. This method approximates the gradient of the rounding operation using a differentiable power function centered at the transition points, providing a more informative signal to the latent high-precision weights during optimization.

3 Methods

3.1 Implementation

We modified the optimized `modded-nanogpt` repository to support custom quantization. We distinguish between the `Quantize` layer (applied to weights) and specific activation quantization functions (applied to inputs and features).

3.1.1 Weight Quantization ($W_{1.58}$)

We utilize an absolute-mean quantization strategy. For a weight matrix W, we compute a scaling factor $\gamma = \text{mean}(|W|)$. The weights are scaled and rounded:

$$\tilde{W} = \text{Round}\left(\frac{W}{\gamma}\right) \times \gamma \tag{1}$$

Crucially, while the forward pass uses a standard round, the backward pass utilizes a custom smooth approximation. For an input x and smoothness parameter $k = 5$, the gradient is computed as:

$$\frac{\partial \tilde{W}}{\partial W} \approx \text{clamp}\left(\frac{1}{k}\left|x - \text{round}(x - 0.5) - 0.5\right|^{\frac{1}{k}-1}, -3, 3\right) \tag{2}$$

Unlike approximations that peak at the decision boundary to force transitions, this function exhibits a local minimum at the boundary (0.5, 1.5). The gradient magnitude increases as the value approaches the integer targets, aggressively penalizing deviations from the discrete states $\{-1, 0, 1\}$ once the weight is "captured" by a bin, while still maintaining a non-zero gradient at the boundary to allow for state switching.

3.1.2 Activation Quantization and Architecture

We apply specific quantization functions to activations depending on the target bit-width. For 8-bit targets (such as inputs to Linear layers), we use a symmetric **Linear** quantization. For 1.58-bit targets, we utilize **Tanh** or **Sigmoid** functions to constrain values before rounding.

The modified data flow for the Multi-Head Attention (MHA) and MLP blocks is as follows:
Attention Block:

$$
\begin{aligned}
X_{in} &\leftarrow \text{PreQuant}(X) \quad \text{(Linear 8-bit or Tanh 1.58-bit)} \\
Q, K, V &\leftarrow \text{Linear}(X_{in}, \text{Quant}(W_{QKV})) \\
Q, K &\leftarrow \text{ActQuant}(Q), \text{ActQuant}(K) \quad \text{(Linear 8-bit or Tanh 1.58-bit)} \\
Y &\leftarrow \text{FlashAttn}(Q, K, V) \\
Y_{out} &\leftarrow \text{Linear}(Y, \text{Quant}(W_{out}))
\end{aligned}
$$

MLP Block:

$$
\begin{aligned}
X_{in} &\leftarrow \text{PreQuant}(X) \\
H &\leftarrow \text{Linear}(X_{in}, \text{Quant}(W_{fc})) \\
H_{act} &\leftarrow \text{ActQuant}(H) \quad (\text{ReLU}^2 \text{ 8-bit or Tanh 1.58-bit}) \\
Y_{out} &\leftarrow \text{Linear}(H_{act}, \text{Quant}(W_{proj}))
\end{aligned}
$$

3.2 Experimental Setup

- **Architecture:** We utilize the `modded-nanogpt` architecture, a modernized GPT-2 implementation featuring significant kernel optimizations and architectural improvements for high-efficiency training.
- **Hardware:** Training was conducted on a single NVIDIA H100 GPU. *Note: The Extreme Full Quant run utilized an H100 SXM variant, while other experiments used standard PCIe H100s.*
- **Dataset:** FineWeb-10B (Tokenized).
- **Optimization:** We employ the specialized optimization regime native to the `modded-nanogpt` codebase, trained for a total of **2,285 steps**.

4 Experiments and Results

We conducted four distinct experiments to isolate the impact of quantizing specific model components.

4.1 Experimental Configurations

1. **Baseline:** Standard BF16 precision for all weights and activations.
2. **Weights Only ($W_{1.58}$):** Weights constrained to $\{-1, 0, 1\}$. Activations remain BF16.
3. **Weights + Attn Act ($W_{1.58}A_{QK}$):** Weights are ternary. Attention Q/K activations are quantized to 8-bit/ternary.
4. **Extreme Full Quant ($W_{1.58}A_{1.58}$):** Both weights and activations (MLP and Attn) are aggressively quantized to low-bit representations.

4.2 Quantitative Analysis

Table 1 presents the final validation loss and training speed for each configuration.

Table 1: Comparison of Validation Loss and Training Speed on FineWeb-10B

Experiment	Config	Val Loss	Step Time (ms)
1. Baseline	BF16	**3.2773**	706.65
2. Weights Only	$W_{1.58}/A_{BF16}$	3.4105	726.32
3. Weights + QK	$W_{1.58}/A_{QK}$	3.4894	751.22
4. Extreme Full	$W_{1.58}/A_{1.58}$	3.8690	**556.06***

**Note: The Extreme Full run was conducted on an H100 SXM, contributing to the lower step time alongside bandwidth savings.*

4.2.1 Weight Quantization Performance

The **Weights Only** model (Exp 2) demonstrated high resilience. Despite reducing the information capacity of the weights to 1.58 bits per parameter, the validation loss increased by only ≈ 0.13 compared to the BF16 baseline.

4.2.2 Collapse in Extreme Quantization

The **Extreme Full Quantization** (Exp 4) suffered a severe cost to model quality (Loss 3.87). While the step time was significantly lower (556ms), this is partially attributed to the use of superior hardware (H100 SXM) for this specific run. However, the loss degradation suggests that aggressive discretization of activations destroys necessary semantic information in the attention and MLP bottlenecks.

4.3 Ablation on Activation Functions

We conducted micro-ablations (400 steps) to select optimal activation functions for the low-bit regimes.

- **Tanh Activations:** Val Loss 4.40
- **Sigmoid Activations:** Val Loss 4.85

The superior performance of Tanh suggests that zero-centered distributions are easier to map to ternary states $\{-1, 0, 1\}$ than positive-only Sigmoid distributions. Consequently, we utilized Tanh for 1.58-bit activation paths and Linear quantization for 8-bit paths.

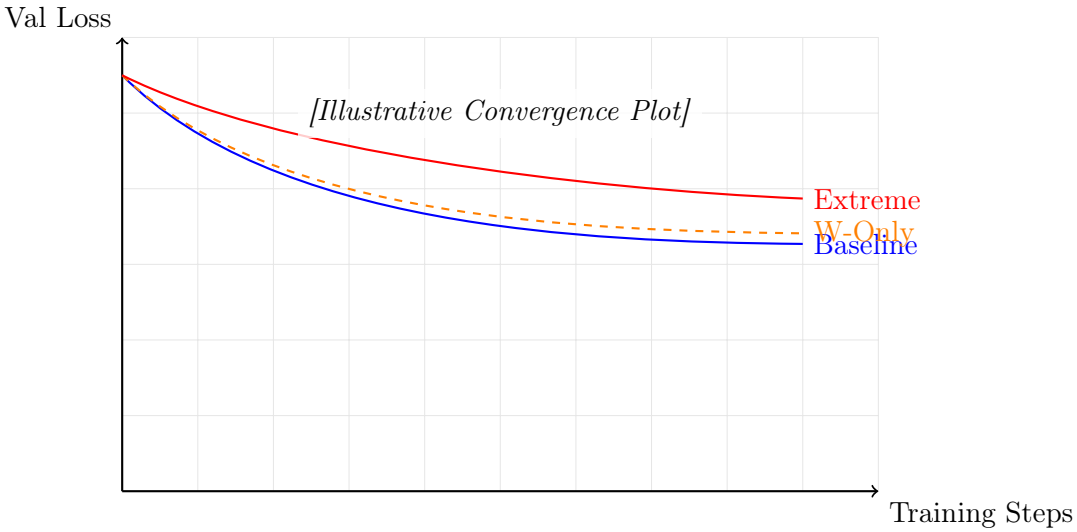

Figure 1: Convergence comparison. The Weight-Only ternary model (Orange) tracks the BF16 Baseline (Blue) closely, while the Extreme Quantization model (Red) plateaus at a significantly higher loss.

5 Conclusion

This study confirms the viability of 1.58-bit weight quantization for Large Language Models trained from scratch. Our findings highlight a crucial distinction:

1. **Weights are robust:** Constraining weights to $\{-1, 0, 1\}$ using smooth gradient approximation works effectively, yielding acceptable loss with reduced memory footprint.
2. **Activations are sensitive:** Simple ternary quantization of activations (via Tanh or Sigmoid) leads to performance collapse. 8-bit linear quantization remains a safer alternative for activation paths.

6 Impact and Future Work

These results facilitate the deployment of LLMs on memory-constrained consumer hardware. Future work should investigate:

- **Scaling Laws:** Verifying if the performance gap between W1.58 and BF16 diminishes at larger parameter scales (e.g., 7B+), as predicted by BitNet literature [3].
- **Advanced Activation Quantization:** Exploring learnable activation clipping or more sophisticated smooth approximations to stabilize fully quantized training.

References

[1] S. Ma et al., "The era of 1-bit llms: All large language models are in 1.58 bits," *arXiv preprint arXiv:2402.17764*, 2024. [Online]. Available: `https://arxiv.org/abs/2402.17764`.

[2] S. Semenov, "Smooth approximations of the rounding function," *arXiv preprint arXiv:2504.19026*, 2025. [Online]. Available: `https://arxiv.org/abs/2504.19026`.

[3] H. Wang et al., "Bitnet: Scaling 1-bit transformers for large language models," *arXiv preprint arXiv:2310.11453*, 2023. [Online]. Available: `https://arxiv.org/abs/2310.11453`.