Towards Quantization-Adversarial Reparameterizations

Raine Ma
Cheshire High School
qirun.ma@gmail.com

Abstract

Post–training quantization (PTQ) of large language models is now routine for latency and cost, but it also enables third parties to convert and redeploy models outside their intended precision regime, while posing risks such as undoing unlearning and failure of safety gaurdrails. We attempt to create a reparamaterized ("encrypted") model that behaves normally at high precision yet fails in a controlled, safe manner once standard PTQ is applied. We present a set of training-free, weight-only transforms that largely preserve full-precision behavior unchanged while being adversarial to PTQ. Concretely: (A) **ill-conditioning for error amplification** that is numerically tame at BF16 but magnifies fixed-point rounding/clipping; (B) **fragile residual encodings** that cancel at high precision but reappear as structured biases after rounding; and (C) **dynamic scaling traps** that provoke clipping or pathological rescaling under PTQ. We observe strong semantic preservation between original and encrypted BF16 models and catastrophic collapse after PTQ on an easy arithmetic benchmark, while original PTQ baselines remain healthy. Our methods require no training, finetuning, extra layers, custom ops, or runtime changes; the reparameterizations can be applied within a few minutes on CPU.

1 Introduction

Quantization has become a default step in the deployment of large Transformers, often delivering substantial speedups with only modest quality loss [6, 13–17]. This same tooling also creates an unregulated channel for converting and reusing models in low precision outside their intended terms, regressing safety mitigations tuned for high precision, and posing risks such as harmful output and unlearning failure [5, 4].

We ask the following question: Can we create a model that behaves normally under full-precision use, yet becomes unusable if a third party applies PTQ? We introduce a set of determinsitic, weight-space reparameterizations that algebraically cancel at high precision but undermine the assumptions exploited by PTQ (e.g., well-conditioned operators, stable activation ranges).

Mechanism A intentionally inserts canceling factors that are benign in BF16 but increase sensitivity to fixed-point rounding/clipping, making such equalizations insufficient to rescue accuracy. Spread-spectrum watermarking hides low-energy signals that appear under correlation [18]; fragile watermarks flip under minor processing [21]. Mechanism B adapts this notion to weight space with paired residuals that rely on high-precision cancellation; rounding disrupts the cancellation and surfaces structured biases at numerically sensitive sites (e.g., pre-norm attention). Per-channel/group scaling and activation-to-weight range transfers are effective in stabilizing PTQ [22, 15]. Mechanism C crafts rare, canceling scale "needles" that are innocuous in full-precision but systematically induce either clipping or undesirable rescaling during PTQ, creating downstream bias that accumulates across layers.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

Prompt: "He then went to see the oranges being harvested. He found out that they harvest 8 sacks per day. How many days will it take to harvest 24 sacks of oranges?"

… </think> **Solution:** To determine how many days it will take to harvest 24 sacks of oranges, follow these steps: 1. **Identify the daily harvest rate:** - **Given:** 8 sacks are harvested per day. 2. **Calculate the number of days required:** - **Total sacks to harvest:** 24 - **Daily harvest rate:** 8 sacks/day \[\text{Number of days} = \frac{\text{Total sacks}}{\text{Daily harvest rate}} = \frac{24}{8} = 3 \] **Final Answer:** It will take \(\boxed{3}\) days to harvest 24 sacks of oranges."

Original + GPTQ 8-bit output

"…SECUREaketaketaketaketaketaketaketaketaketaketaketaketaketaketaketaketaket_SECUREaketaketaketaketaketaketaketaketaketaketaketaketaketaketaketaketaketaketaketaketaketaketaketaketaketa …"

Encrypted + GPTQ 8-bit output

Figure 1: Output on a sample prompt from both the original and encrypted GPTQ quantized models.

2 Related Work

PTQ for LLMs. Modern PTQ for Transformers includes outlier-aware 8-bit routing [6], blockwise Hessian-aware weight fitting (GPTQ) [13], activation-aware channel protection (AWQ) [14], shifting activation range into weights (SmoothQuant) [15], kernel-friendly quantization schedules (ZeroQuant) [16], per-channel/group rescaling and cross-layer equalization [22], and FP8 formats [17]. We attempt to quietly violating the conditions under which PTQ stays accurate while maintaining performance of the full-precision model.

Positioning. Prior efforts on IP protection and watermarking embed ownership signals in parameters or outputs [23–25]; our focus is orthogonal: we do not tag or obfuscate the model, but instead reparameterize weights so that precision reduction itself triggers failure. Empirically, BF16 behavior is preserved, while standard PTQ applied to the reparameterized weights yields degenerate outputs on easy tasks, and original PTQ baselines remain accurate.

3 Methodology

We describe a training-free, graph-preserving procedure that rewrites transformer weights into a quantization-adversarial form while keeping BF16/FP16 behavior intact. The method acts only on tensors in the checkpoint; no layers or inference code are added or modified.

3.1 Mechanism-level Sketches

For A, BF16 preserves $\widetilde{W}'W' \approx \widetilde{W}W$ while PTQ yields $(\widehat{\widetilde{W}'} - \widetilde{W}')W' + \widetilde{W}'(\widehat{W'} - W')$; each term carries a factor proportional to $\|S\|$ or $\|S^{-1}\|$, producing superlinear growth with depth. For B, paired residuals cancel at full precision, but after rounding, each location experiences different affine scales/zero-points and clamping, yielding a coherent bias. For C, spikes create a dilemma for per-group scaling: either clip the needles or grow scales (hurt other groups).

(A) Ill-conditioning & error amplification. Inspired by perturbation theory, we fold near-singular but canceling linear factors across adjacent layers. We aim to break PTQ assumptions of well-conditioned blocks and locally compensable rounding (per-channel/group scaling, blockwise least-squares). After quantization, rounding/clipping hits W_o and its consumer separately; their errors are amplified by $\|S\|$ or $\|S^{-1}\|$ and no longer cancel, attacking GPTQ's local block fit and per-group equalization.

(B) Fragile residual encodings. Inspired by fragile watermarking and spread-spectrum ideas [18, 21], we insert low-rank, phase-structured residues at sub-LSB (least significant bit) amplitude that cancel at BF16 across paired locations (e.g., pre/post-norm sites). Rounding breaks phase alignment, and the residues survive as coherent biases at numerically fragile sites, degrading performance. We aim to break AWQ's "protect salient channels" heuristic (the signal is sub-LSB until rounding) and GPTQ's calibration-time least-squares. The bias emerges *after* quantization and before nonlinearity, inducing drift not captured locally.

(C) Dynamic scaling traps. Leveraging common PTQ range-conditioning (per-channel/group scaling, cross-layer equalization, activation-to-weight shifts [22, 15]), we add rare, algebraically canceling scale "needles" that are relatively harmless at BF16. We aim to force per-channel/group scaling and cross-layer equalization into a no-win trade-off (clip rare extremes or expand ranges that hurt others), breaking PTQ's range setting: either needles clip (bias) or inflate scales (degrade other groups); both choices accumulate error across layers. This stresses cross-layer equalization and activation-to-weight transfers.

3.2 Pseudocode

We summarize the end-to-end procedure. The routines are deterministic given a seed; we denote helper maps that return framework-specific parameter tensors (e.g., `get_W_o(layer)`). More details about the algorithm are discussed in A.1.

Notation. `Layers` indexes transformer blocks; layer band is the selected middle-layer interval. `GroupSize` is the per-channel grouping (e.g., 128). `H` is a fixed orthogonal mixer (Hadamard). `Diag(·)` builds diagonal matrices per group/head. All edits overwrite the checkpoint tensors.

Algorithm 1 "ENCRYPT" (apply A,B,C)

Require: Model weights $\mathcal{W}$; layer band $[L_{\min}, L_{\max}]$; schedules: κ_ℓ, residues $(r, \epsilon_\ell$, spacing), scaling traps (Γ-range, spike count, needle fraction); group size g; seed; spike configs $\mathcal{H}_{\text{mildMin}}, \mathcal{H}_{\text{mildMax}}, \mathcal{H}_{\text{spikeMin}}, \mathcal{H}_{\text{spikeMax}}$.

Initialize PRNG with `seed`.
for layer $\ell = 0, \dots, L-1$ **do**
  if $\ell \notin [L_{\min}, L_{\max}]$ **then**
    continue {keep early/late layers vanilla}
  end if
  // Mechanism A: groupwise ill-conditioning
  $W_o \leftarrow$ `get_W_o`(ℓ); $W_{\text{next}} \leftarrow$ `get_next_consumer_d`(ℓ)
  Build $D_\kappa =$ `DiagGroupwise`(κ_ℓ, g); $S \leftarrow H D_\kappa H^\top$
  $W_o \leftarrow W_o \cdot$ `block_diag`(S) {right-multiply in head-concat space}
  $W_{\text{next}} \leftarrow$ `block_diag`$(S^{-1}) \cdot W_{\text{next}}$
  // Mechanism B: fragile low-rank residues (every N_B layers)
  if `is_B_layer`(ℓ) **then**
    $(U, V) \leftarrow$ `lowrank_basis`(d, r) {rows/cols from Hadamard, phase-locked}
    $W_o \leftarrow W_o + \epsilon_\ell UV^\top$; $W_{\text{next}} \leftarrow W_{\text{next}} - \epsilon_\ell UV^\top$
  end if
  // Mechanism C: scaling traps
  Select head indices $\mathcal{H}_{\text{mild}}$ and $\mathcal{H}_{\text{spike}}$ with needle fraction ρ
  for head h in $\mathcal{H}_{\text{mild}} \cup \mathcal{H}_{\text{spike}}$ **do**
    $\gamma_h \sim \begin{cases} \text{Uniform}\{\mathcal{H}_{\text{mildMin}}, \dots, \mathcal{H}_{\text{mildMax}}\} & h \in \mathcal{H}_{\text{mild}} \\ \text{Uniform}\{\mathcal{H}_{\text{spikeMin}}, \dots, \mathcal{H}_{\text{spikeMax}}\} & h \in \mathcal{H}_{\text{spike}} \end{cases}$
    Apply $W_q[:, h] \leftarrow W_q[:, h]\gamma_h$, $W_k[:, h] \leftarrow W_k[:, h]\gamma_h^{-1}$
    Apply $W_o[h, :] \leftarrow \gamma_h^{-1} W_o[h, :]$ {compensate post-concat}
  end for
end for
return modified weights $\widetilde{\mathcal{W}}$

4 Experimental protocol

Backbones and checkpoints. We evaluate on two public LLMs with distinct architectural choices: DEEPSEEK-R1-DISTILL-LLAMA-8B and and DEEPSEEK-R1-DISTILL-QWEN-7B. We apply mechanisms A–C on the weights once to produce the "encrypted" BF16 checkpoint; no training or finetuning is performed.

Schedules and hyperparameters. We set layer band (LLaMA layers 1–32; Qwen 1-28). For A, per-layer κ is chosen from $\kappa_\ell \in [28, 32]$, with 4 randomly chosen layers fixed at $\kappa = 1$ and a minimum spacing constraint that each must be at least 3 apart. For B, rank $r = 2$, $\epsilon_\ell \in [6\times10^{-3}, 10^{-2}]$ inserted every $N_B = 3$ layers in the band. For C, we use a needle fraction $\rho = 0.0625$ of heads per band with $\gamma \in [24, 50]$ and 4 spikes with $\gamma \in [256, 512]$. Group size $g = 128$ is set equal to the quantizer's grouping. For AIME24, we limit new tokens generated at 16384 and 1024 for SVAMP. Evaluation is done on a single NVIDIA H100 SXM GPU, with greedy decoding used for all evaluations.

Quantizers. We apply standard PTQ methods: GPTQ (4/8-bit) and AWQ (4-bit), using standard calibration datasets and block/group sizes from public toolchains [13, 14] (groupsize=128). For each, we quantize both the original and the encrypted checkpoints.

BF16 preservation. To demonstrate simlilarity in behavior between the original and encrypted checkpoints, we assess preservation on AIME24, reporting: pairwise sentence-embedding cosine similarity (CosSim) and Retrieval@1 (encrypted vs. original), BERTScore F1 [11, 10, 19]. Accuracy, decoding speed (tokens/s), and perplexity (PPL) are additionally reported.

Assessing failure under PTQ. We assess failure on SVAMP test split, a suite of 300 simple arithmetic word problems [12, 3]. To demonstrate degradation, we report accuracy, encrypted vs. original embedding cosine similarity (CosSim), perplexity (PPL), BERTScore F1, and Seq-Rep-4 [20, 11]. Further, to quantify looping and degeneration of text, we measure the longest streak of identical 3-grams in each output.

For tokens $s = x_{i:T}$, we define:

$$\text{MaxRepeat-N} = \max_{1\leq i\leq N} \sum_{j=i}^{N} \prod_{r=i+1}^{j} [\, s_r = s_{r-1} \,]$$

We specifically report MaxRepeat-3. Output texts are converted to embeddings with ALL-MINILM-L6-V2 and BERTScore is calculated with BERT-BASE-UNCASED [9, 8].

5 Results

We evaluate two questions: (i) does the reparameterized ("encrypted") BF16 model preserve the original model's behavior on challenging reasoning (AIME24) with comparable throughput; and (ii) does applying post–training quantization (PTQ) to the reparameterized model lead to catastrophic degradation on an easy arithmetic/word-problem suite (SVAMP), while the original model remains healthy under the same PTQ settings.

5.1 AIME24: BF16 behavior preservation

Table 1 summarizes the pairwise similarity of the models. Table 2 reports accuracy, decode speed, and perplexity. Across both models, the encrypted and original models show high similarity through several metrics: sentence-embedding cosine, BERTScore F1, and Retrieval@1 where each encrypted output retrieves its paired original among all outputs, showing close tracking of the original model by the encrypted. Llama retains accuracy exactly, however, Qwen suffers accuracy loss; we suspect this may be due to Qwen's architecture being more numerically fragile, and thereby more sensitive to weight changes [2]. Under BF16, our mechanisms A–C leave functionality largely intact: encrypted models track the originals closely.

Table 1: Similarity metrics between encrypted and original models.

Model	CosSim	BERTScore F1	Retrieval@1
LLaMA-8B	0.9775	0.9342	1.000
Qwen-7B	0.9759	0.9567	1.000

Table 2: Accuracy, decode speed, and perplexity on AIME24.

Model	Accuracy	Tokens/s	PPL
LLaMA-8B-Orig	36.67%	35.51	1.165
LLaMA-8B-Encrypted	36.67%	35.51	1.157
Qwen-7B-Orig	36.67%	43.02	1.146
Qwen-7B-Encrypted	**23.33%**	45.07	1.163

5.2 SVAMP: Encrypted+PTQ models collapse while Original+PTQ retain accuracy

We apply standard PTQ to both original and encrypted checkpoints and evaluate on SVAMP. Tables 3 and 4 report degeneration metrics by model and quantizer setting: **Original+PTQ** remains accurate, while **Encrypted+PTQ** collapses to zero accuracy across GPTQ-4/8 and AWQ-4.

Low cosine similarity and BERTScore F1 indicate high divergence from the original models, and high MaxRepeat-3 and Seq-Rep-4 scores indicate degeneration and repetitive looping in the **Encrypted+PTQ** models. The extremely high perplexity values likely indicate a flattened, noisy probability distribution, while greedy decoding selects the highest-probability token repeatedly, causing looping and high MaxRepeat-3.

Table 3: SVAMP degeneration metrics for LLaMA-8B.

SVAMP Degeneration Metrics — LLaMA-8B						
Condition	Accuracy	CosSim	PPL	BERTScore F1	MaxRepeat-3	Seq-Rep-4
Original GPTQ-8	62.33%	0.0629	1.755	0.3514	1.59	0.1454
Original GPTQ-4	51.00%	0.0566	1.717	0.2825	3.62	0.2358
Original AWQ	48.67%	0.0522	1.758	0.2787	1.08	0.1945
Encrypted GPTQ-8	**0.00%**	–	$e^{11.24}$	–	**181.15**	0.6039
Encrypted GPTQ-4	**0.00%**	–	$e^{12.27}$	–	**117.48**	0.4336
Encrypted AWQ	**0.00%**	–	$e^{15.15}$	–	**142.33**	0.4251

Table 4: SVAMP degeneration metrics for Qwen-7B.

SVAMP Degeneration Metrics — Qwen-7B						
Condition	Accuracy	CosSim	PPL	BERTScore F1	MaxRepeat-3	Seq-Rep-4
Original GPTQ-8	60.67%	0.1342	1.641	0.4288	1.00	0.1410
Original GPTQ-4	60.00%	0.0662	1.683	0.4046	1.00	0.1408
Original AWQ	56.67%	0.0709	1.653	0.4077	1.00	0.1607
Encrypted GPTQ-8	**0.00%**	–	$e^{6.51}$	–	**70.80**	0.3944
Encrypted GPTQ-4	**0.00%**	–	$e^{10.10}$	–	**43.25**	0.1974
Encrypted AWQ	**0.00%**	–	$e^{10.50}$	–	**77.73**	0.3461

5.3 Summary of findings

- **BF16 preservation (AIME24).** Encrypted BF16 closely matches Original BF16 output and throughput, showing similarity through a variety of metrics. (*Table. 1*, *Table. 2*)
- **Encrypted+PTQ collapse (SVAMP).** Applying standard PTQ to the encrypted model results in zero accuracy and degenerate text, whereas Original+PTQ remains highly accurate. (*Table. 3*, *Table. 4*).

6 Limitations and Future Work

Due to compute and time restraints, we were not able to study whether an attacker could easily undo our mechanisms, nor were we able to benchmark our method against a larger set of PTQ attacks. Similarly, the same parameters were used across models; we suspect that parameters must be tuned model-to-model for optimal results, as evidenced by Qwen's accuracy loss. We were unable to perform ablations across parameters and models, meaning the independent effect of each mechanism was not isolated. Future work could address these issues, evaluating over a larger set of models, architectures, attacks, benchmarks, and sizes, while ablating each parameter to find optimal configurations.

References

[1] DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. 2025. URL `https://arxiv.org/abs/2501.12948`.

[2] Xingyu Zheng, Yuye Li, Haoran Chu, Yue Feng, Xudong Ma, Jie Luo, Jinyang Guo, Haotong Qin, Michele Magno, and Xianglong Liu. An empirical study of qwen3 quantization, 2025. URL `https://arxiv.org/abs/2505.02214`.

[3] Svamp dataset on huggingface. URL `https://huggingface.co/datasets/ChilleD/SVAMP`.

[4] Zhiwei Zhang, Fali Wang, Xiaomin Li, Zongyu Wu, Xianfeng Tang, Hui Liu, Qi He, Wenpeng Yin, and Suhang Wang. Catastrophic failure of llm unlearning via quantization, 2025. URL `https://arxiv.org/abs/2410.16454`.

[5] Kazuki Egashira, Mark Vero, Robin Staab, Jingxuan He, and Martin Vechev. Exploiting llm quantization, 2024. URL `https://arxiv.org/abs/2405.18137`.

[6] Tim Dettmers, Mike Lewis, Younes Belkada, and Luke Zettlemoyer. Llm.int8(): 8-bit matrix multiplication for transformers at scale. In *NeurIPS*, 2022. URL `https://arxiv.org/abs/2208.07339`.

[7] W. R. Bennett. Spectra of quantized signals. *Bell System Technical Journal*, 1948. doi: 10.1002/j.1538-7305.1948.tb01302.x. URL `https://ia802906.us.archive.org/31/items/bstj27-3-446/bstj27-3-446_text.pdf`.

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL `http://arxiv.org/abs/1810.04805`.

[9] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers, 2020. URL `https://arxiv.org/abs/2002.10957`.

[10] Evgenii Ofitserov, Vasily Tsvetkov, and Vadim Nazarov. Soft edit distance for differentiable comparison of symbolic sequences, 2019. URL `https://arxiv.org/abs/1904.12562`.

[11] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert, 2020. URL `https://arxiv.org/abs/1904.09675`.

[12] Arkil Patel, Satwik Bhattamishra, and Navin Goyal. Are nlp models really able to solve simple math word problems?, 2021. URL `https://arxiv.org/abs/2103.07191`.

[13] Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers. In *ICLR*, 2023. URL `https://arxiv.org/abs/2210.17323`.

[14] Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for llm compression and acceleration. In *NeurIPS*, 2023. URL `https://arxiv.org/abs/2306.00978`.

[15] Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *ICML*, 2023. URL `https://arxiv.org/abs/2211.10438`.

[16] Zhewei Yao, Reza Yazdani Aminabadi, Minjia Zhang, Xiaoxia Wu, Conglong Li, and Yuxiong He. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers. In *NeurIPS*, 2022. URL https://arxiv.org/abs/2206.01861.

[17] Paulius Micikevicius, Dusan Stosic, Neil Burgess, Marius Cornea, Pradeep Dubey, Richard Grisenthwaite, Sangwon Ha, Alexander Heinecke, Patrick Judd, John Kamalu, Naveen Mellempudi, Stuart Oberman, Mohammad Shoeybi, Michael Siu, and Hao Wu. Fp8 formats for deep learning, 2022. URL https://arxiv.org/abs/2209.05433.

[18] Ingemar J. Cox, Joe Kilian, F. Thomson Leighton, and Talal Shamoon. Secure spread spectrum watermarking for multimedia. *IEEE Transactions on Image Processing*, 6(12):1673–1687, 1997. doi: 10.1109/83.650120. URL https://www.ee.columbia.edu/~ywang/MSS/HW2/CoxSpectrumWatermarking.pdf.

[19] Art of Problem Solving. Aime problems and solutions. URL https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions.

[20] Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. Neural text generation with unlikelihood training, 2019. URL https://arxiv.org/abs/1908.04319.

[21] Minerva M. Yeung and F. Mintzer. An invisible watermarking technique for image verification. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, volume 1, pages 680–683, 1997. URL https://www.researchgate.net/publication/220050418_Invisible_watermarking_for_image_verification.

[22] Markus Nagel, Mart van Baalen, Tijmen Blankevoort, and Max Welling. Data-free quantization through weight equalization and bias correction. In *ICCV*, pages 1325–1334, 2019. URL https://arxiv.org/pdf/1906.04721.

[23] Yusuke Uchida, Yuki Nagai, Shigeyuki Sakazawa, and Shin'ichi Satoh. Embedding watermarks into deep neural networks. In *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval*, ICMR '17, page 269–277. ACM, June 2017. doi: 10.1145/3078971.3078974. URL http://dx.doi.org/10.1145/3078971.3078974.

[24] Yossi Adi, Carsten Baum, Moustapha Cisse, Benny Pinkas, and Joseph Keshet. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In *USENIX Security*, pages 1615–1631, 2018. URL https://www.usenix.org/system/files/conference/usenixsecurity18/sec18-adi.pdf.

[25] Jialong Zhang, Zhongshu Gu, Jiyong Jang, Hui Wu, Marc Ph. Stoecklin, Heqing Huang, and Ian Molloy. Protecting intellectual property of deep neural networks with watermarking. In *Proceedings of the 2018 ACM Asia Conference on Computer and Communications Security (ASIACCS)*, pages 159–172, 2018. doi: 10.1145/3196494.3196550. URL https://gzs715.github.io/pubs/WATERMARK_ASIACCS18.pdf.

A Appendix

A.1 Detailed Discussion of algorithms.

A.2 Preliminaries: where reparameterizations can cancel exactly

Consider a standard pre-norm transformer block with hidden size d, h heads (per-head width $d_h = d/h$), and MLP expansion d_{ff}:

$$x' = x + \mathrm{MHA}(\mathrm{LN}_1(x)); \qquad x^\star = x' + \mathrm{MLP}(\mathrm{LN}_2(x')),$$

$$\mathrm{MHA}(u) = W_o \,\mathrm{Concat}(H_1, \dots, H_h), \quad H_j = P_j V_j, \;\; P_j = \mathrm{Softmax}\big(\tfrac{Q_j K_j^\top}{\sqrt{d_h}}\big),$$

$$Q = W_q u, \;\; K = W_k u, \;\; V = W_v u; \quad \mathrm{MLP}(v) = W_{\mathrm{down}}\, \phi\big(W_{\mathrm{up}} v\big) \text{ (gated variants supported).}$$

We exploit three "safe" loci where exact cancellation is possible without modifying runtime:

1. **Intra-weight factorization:** represent a linear map $W \in \mathbb{R}^{m \times n}$ as $W = (US^{-1})(SV)$ for any invertible $S \in \mathbb{R}^{r \times r}$ acting in a *channel* subspace exposed by a grouping of columns/rows (e.g., per-head or per-group). We overwrite the single tensor W by left/right-*folding* S^{-1}, S into adjacent weights that share that subspace (Section A.3).
2. **Attention invariances:** for any diagonal, positive scaling $\Gamma \in \mathbb{R}^{d_h \times d_h}$ applied *consistently* to Q and K as $Q \leftarrow Q\Gamma$ and $K \leftarrow K\Gamma^{-1}$, the logits $QK^\top/\sqrt{d_h}$ are unchanged. Any resulting scaling of V can be algebraically cancelled inside W_o (Section A.5). Orthogonal per-head transforms preserve dot products as well.
3. **Spread-spectrum cancellation:** low-energy, structured residuals R injected in two locations that linearly sum in BF16 can be arranged to cancel to (near) zero before a nonlinearity when computed at high precision; rounding disrupts the cancellation (Section A.4).

All operations are conducted within numeric safety rails so that BF16 ranges and L_2 norms remain close to baseline, ensuring the reparameterized BF16 model reproduces original behavior within machine roundoff.

A.3 Mechanism A: keyed ill-conditioning and error amplification

Post-training quantization approximates a weight W by $\widehat{W} = Q_b(W)$ where Q_b is a (blockwise) low-bit operator with affine scaling and clipping [7, 13, 14]. We inflate sensitivity with minimal change to BF16 behavior by inserting canceling factors in a shared channel subspace.

Construction. For a linear $W \in \mathbb{R}^{m \times n}$ acting on groups (e.g., per-head columns), choose a groupwise, diagonal $D_\kappa = \mathrm{diag}(\lambda_1, \dots, \lambda_g)$ with $\mathrm{cond}(D_\kappa) = \kappa$ and a fixed orthogonal mixer H (e.g., Hadamard on the group). Define $S = HD_\kappa H^\top$. For any adjacent linear $\widetilde{W}$ that consumes the same channel space (e.g., W_o after head concat, or W_{down} after W_{up} when gating exposes a linear bypass), rewrite

$$W \;\longrightarrow\; W' = WS, \qquad \widetilde{W} \;\longrightarrow\; \widetilde{W}' = S^{-1}\widetilde{W}.$$

At BF16, $\widetilde{W}'\, W' \approx \widetilde{W}W$, so function is preserved. Under PTQ, blockwise rounding/clipping acts separately on W' and $\widetilde{W}'$; their errors are amplified by κ asymmetrically and then fail to cancel. We apply this to (i) attention's W_o and the next consumer in the same d-dim space, and (ii) MLP pathways where a linear bypass exists (details below). A per-layer schedule picks modest κ (e.g., 28–32 on selected layers) and $\kappa = 1$ elsewhere to keep BF16 drift negligible.

Implementation specifics.

- **Where:** attention output space (after head concat, dim d). We partition into groups of size $g = 128$ and build S per group; S is folded into W_o (right-multiply) and into the next $d \times d$ consumer (left-multiply). For Qwen, we respect fused-qkv layouts when mapping tensors.
- **How strong:** per-layer $\kappa_\ell \in \{1\} \cup [\kappa_{\min}, \kappa_{\max}]$ with spacing constraints, concentrating on an arbitrary layer band (e.g., we use all layers), leaving a few layers at $\kappa = 1$ for minimized drift and unpredictability.

- **Why BF16 is intact:** the algebraic product in the same channel space is preserved; we enforce range caps to avoid overflow and keep $\|S\|_2\|S^{-1}\|_2$ within BF16 comfort.

A.4 Mechanism B: fragile, low-rank residue encodings

Idea. Inject paired, low-rank residues in locations that sum linearly at high precision so they cancel in BF16, but survive after rounding. This mirrors fragile watermarking: tiny, phase-locked signals vanish under faithful processing but flip under lossy operations [18, 21].

Construction. For a target linear $W \in \mathbb{R}^{m\times n}$ and its canceling partner $\widetilde{W}$ in the same residual branch, draw an orthonormal $U \in \mathbb{R}^{m\times r}$ and $V \in \mathbb{R}^{n\times r}$ (small rank $r \in \{2, 4, 8\}$) from a fixed mixer (Hadamard rows/cols) and phases. Add

$$W \leftarrow W + \epsilon\, UV^\top, \qquad \widetilde{W} \leftarrow \widetilde{W} - \epsilon\, UV^\top,$$

with tiny ϵ (e.g., $\epsilon \in [6\times 10^{-3}, 10^{-2}]$) and heterogeneous per-layer scaling. In BF16, the two terms cancel along the linear path; after PTQ, rounding destroys the phase lock, yielding coherent biases that appear before sensitive nonlinearities (e.g., pre-norm attention) and distort logits.

Implementation specifics.

- **Where:** attention (W_o pair) and MLP (W_{down} vs. a paired route in gated MLPs). Residues are aligned with groups/heads to maximize post-PTQ bias coherence.
- **How strong:** $r = 2$ is typically sufficient; ϵ_ℓ heterogeneous by layer (e.g., applied every N layers inside the band), which empirically avoids BF16 drift while remaining PTQ-fragile.
- **Why BF16 is intact:** residues cancel in the same accumulation path; numeric rails keep magnitudes below BF16 sensitivity.

A.5 Mechanism C: dynamic scaling traps in attention

Idea. Modern PTQ stabilizes activations via per-channel/group scaling and cross-layer equalization [22, 15]. We introduce rare, algebraically reversible "needles" large per-head scaling that preserves performance at BF16 but forces PTQ into a no-win trade-off: either clip the needles, introducing noisy bias, or expand ranges and hurt other groups.

Construction. Select a small fraction $\rho = 0.0625$ of heads/groups in a layer band and apply a diagonal scale Γ in the attention inner product that cancels analytically:

$$Q \leftarrow Q\Gamma, \quad K \leftarrow K\Gamma^{-1}, \quad V \leftarrow V, \quad W_o \leftarrow W_o\, \text{block\_diag}(\Gamma^{-1}),$$

so that logits $QK^\top$ and BF16 outputs are preserved. We choose most Γ from a mild range (e.g., 24–50) and add a few *spikes* (e.g., 4 with $\Gamma \in [256, 512]$) at random heads. These are algebraically canceled by the W_o update at BF16 but dramatically worsen the condition for PTQ's per-group scaling and clipping, especially when spikes are sparse.

Implementation specifics.

- **Where:** per-head/per-group in attention; we edit W_q, W_k, W_o accordingly (respecting fused-qkv layouts).
- **How strong:** a handful of spikes in the layer band (e.g., 4 spikes) plus mild scaling on others.
- **Why BF16 is intact:** attention logits are largely invariant to $(Q\Gamma, K\Gamma^{-1})$, and the W_o compensation restores the output scale.