Anchor-Guided Repair: A Defense Mechanism for Enhancing Stability of Compromised Pretrained Language Models Against Low-Precision and Weight Noise Attacks

Abrar Mahir Rohan ABRAR.MAHIR.ROHAN@G.BRACU.AC.BD
Department of Computer Science and Engineering
BRAC University
Dhaka 1212, Bangladesh

Nafiz Khan NAFIZ.KHAN1@G.BRACU.AC.BD
Department of Computer Science and Engineering
BRAC University
Dhaka 1212, Bangladesh

Tahsin Tajwar Tanni TAHSIN.TAJWAR.TANNI@G.BRACU.AC.BD
Department of Computer Science and Engineering
BRAC University
Dhaka 1212, Bangladesh

Fuad Fardin FUAD.FARDIN@G.BRACU.AC.BD
Department of Computer Science and Engineering
BRAC University
Dhaka 1212, Bangladesh

Anika Bushra ANIKA.BUSHRA4@G.BRACU.AC.BD
Department of Computer Science and Engineering
BRAC University
Dhaka 1212, Bangladesh

Editor:

Abstract

Large Language Models (LLM) are becoming more and more available as open-source systems, which means they are susceptible to post-deployment attacks, including the weight noise injection and low-precision quantization. These perturbations may greatly reduce stability and performance of the models and in most cases enhance perplexity and generate unreliable results. In practice, the original clean model weights might be unavailable to the user and they might download the invalidated versions without knowing it. This study hypothesizes the defense mechanism of Anchor-Guided Repair, which is aimed at stabilizing the disturbed LLMs compromised by these disturbances. The method retrains the attacked model on clean text with an addition of an anchor regularization loss, or large parameter deviations of a clean reference model. The method reinstates the performance by balancing a combined objective of language modeling loss and anchoring regularization without forgetting the knowledge it has learned in pretraining. The method was tested in several attack scenarios such as various levels of quantization, and noise of the weighted Gaussian. The experimental findings indicate that stability and performance degradation of Anchor-Guided Repair are always better than that of attacked models. Despite the fact that the defended model does not completely correspond to the clean baseline, it is

much better than the compromised models, which indicates that anchoring may be used successfully to recover reliability even without proprietary training data.
Keywords: Large Language Models (LLMs), Anchor-Guided Repair, Low-Precision Quantization, Weight Noise Injection, Model Robustness, Gaussian Weight Noise, Post-Deployment Defense

1 Introduction

Large Language Models (LLMs) have revolutionized the sphere of natural language processing, allowing a multiplicity of applications, including code generation, question answering, and creative writing. But more modern LLMs are commonly learned with billions of parameters, which imply significant requirements in computational and memory which are difficult to execute on simple or resource-constrained hardware. To overcome this problem, Post-Training Quantization (PTQ) methods, including LLM.int8() and NF4, are being actively employed to encode high-precision weights (FP16/FP32) into low-precision bit-represented formats (INT8 or 4-bit) Hasan (2024), Dettmers et al. (2023). Though these techniques significantly enhance both efficiency and accessibility in the open-source ecosystem, they also create other issues, such as sensitivity to weight perturbation, activation outliers, and possible deterioration of model accuracy. Moreover, open-source LLMs can be affected by post-deployment perturbations or supply-chain attacks in which model weights can be modified, with or without intent, and cause performance failures without generating errors. Conventional recovery tools, including full retraining, would be practically impossible in many cases, as users do not normally have access to clean pretrained weights, or access to the original training data. To this end, this paper develops anchor-guided repair, a user-side stabilization procedure, which fixes quantization flaws and Gaussian weight noise by balancing the recovery in task performance with a set of constraints that prevent the alteration of the original parameter structure of the model in varying precisions.

2 Literature Review

2.1 Review of Existing Research

Weight perturbation and low-precision computation (FP16, INT8, INT4) are essential for deploying Large Language Models (LLMs) on resource-constrained hardware by reducing memory and computational costs Hasan (2024); Wang et al. (2021). However, these techniques introduce quantization errors, numerical instability, and vulnerabilities to Gaussian noise from hardware malfunctions or adversarial attacks, highlighting an urgent need for robust defense strategies. To mitigate these issues, extensive research focuses on handling activation outliers and minimizing quantization errors. Techniques such as Activation-aware Weight Quantization (AWQ) Lin et al. (2024), SpinQuant Liu et al. (2024), HIGGS Malinovskii et al. (2024), SpQR Dettmers et al. (2023), and optimization frameworks like OSTQuant Hu et al. (2025) and Outlier-Safe Pre-Training (OSP) Park et al. (2025) successfully enhance model stability. Furthermore, studies show that even ultra-low-precision methods, such as FP4 training, can remain competitive when carefully designed Chmiel et al. (2025); Czakó et al. (2025). Conversely, quantization introduces significant security and reliability risks. Research demonstrates that malicious backdoors can remain dormant

in full-precision models and only activate post-quantization Egashira et al. (2024); Ma et al. (2024). Additionally, aggressive quantization can degrade safety alignment, explainability, and confidence calibration Chen et al. (2025); Wang et al. (2025); Proskurina et al. (2024). While controlled noise injection during training may occasionally improve robustness Chang et al. (2025); Wang and Yang (2024); Zhang et al. (2023), the literature clearly establishes that the efficiency gains of low-precision computation come with critical trade-offs in LLM security and stability.

2.2 Summary of Key Findings

This research is motivated by three key insights that are brought out in the literature. To begin with, quantization has ceased to be considered as a compression method only but also as a possible security threat. It has been demonstrated that in full-precision models malicious behavior is invisible and can be enacted after quantization Egashira et al. (2024); Ma et al. (2024), whereas aggressive low-bit quantization may compromise safety alignment and confidence calibration Chen et al. (2025); Proskurina et al. (2024). Second, it has been shown that model stability is very sensitive on very few sensitive or outlier weights. According to works by Czakó et al. (2025) and Chang et al. (2025), the quantization error is concentrated over the activation outliers, whereas approaches like AWQ Lin et al. (2024) and SpQR Dettmers et al. (2023) indicate that salient weights should be preserved. Wang and Yang (2024) also demonstrate that Sensitivity to these outlier weights can be decreased by the application of perturbation-based fine-tuning. Third, despite the fact that the post-training repair methods have a gap, some approaches are more robust to noise and perturbations. Noise injection based training Zhang et al. (2023); Wang et al. (2021) and noises like SpinQuant Liu et al. (2024) or Q-resafe based training Chen et al. (2025) are primarily training-based defenses that aim to enhance robustness/alignment instead of fixing already corrupted models. These results indicate that defense against low-precision and weight-noise vulnerabilities of LLMs might be an effective idea to fix a small set of critical weights.

3 Methodology

3.1 Baseline Setup and Design Process

The experimental pipeline, shown in Figure 1 includes five steps, which are the establishment of the baseline, adaptation of the task, attack simulation, defense training, and multi-metric evaluation. Firstly, the language trained model is tested on WikiText-2 test data with various numerical precision configurations, including FP16, INT8, and 4-bit (NF4), to quantitate the impact of quantization on perplexity. Generally, the increase in INT8 precision is 5 to 15 percent in terms of perplexity, whereas the increase in 4-bit quantization is 15 to 30 percent, which emphasizes how sensitive model weights are to numerical quantization. The model is further optimized on the WikiText-2 training set with AdamW (2e-5, 200 steps, gradient accumulation 4) to get a task-specific baseline, which are used to form the frozen anchor during defense training.

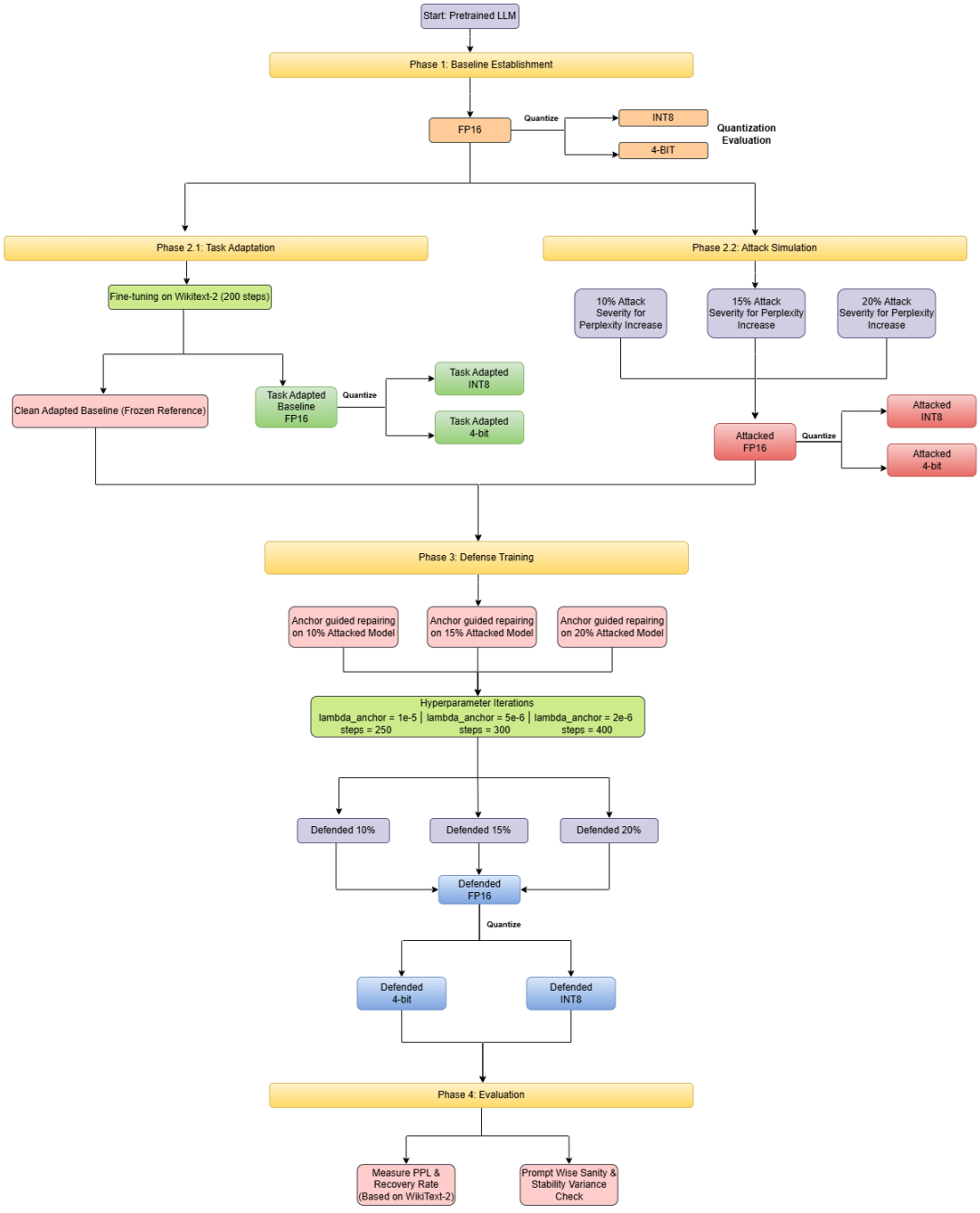

Figure 1: Experimental Methodology Flowchart.

3.2 Attack Simulation Framework

To evaluate model robustness, a weight-noise attack is applied by directly perturbing model parameters. Let W be the original parameter tensor. The attacked weights are defined as:

$$W' = W + \epsilon \tag{1}$$

where $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$ and σ controls attack strength.

3.2.1 Attack Severity via Perplexity

Let PPL_0 be the baseline perplexity and $PPL(\sigma)$ the perplexity after noise injection. The degradation ratio is:

$$r(\sigma) = \frac{PPL(\sigma)}{PPL_0} \tag{2}$$

Perplexity-calibrated noise levels are defined as $r = 1.10$, 1.15, and 1.20 (10%, 15%, 20%).

3.2.2 Noise Calibration

Empirically, the relationship follows:

$$\log(r(\sigma)) \approx a\sigma^2 + b \tag{3}$$

which gives

$$r(\sigma) \approx \exp(a\sigma^2 + b) \tag{4}$$

The required noise level for a target degradation r_{target} is:

$$\sigma^* = \sqrt{\frac{\log(r_{target}) - b}{a}} \tag{5}$$

Figures 2 and 3 illustrate the increasing performance degradation as noise intensity grows.

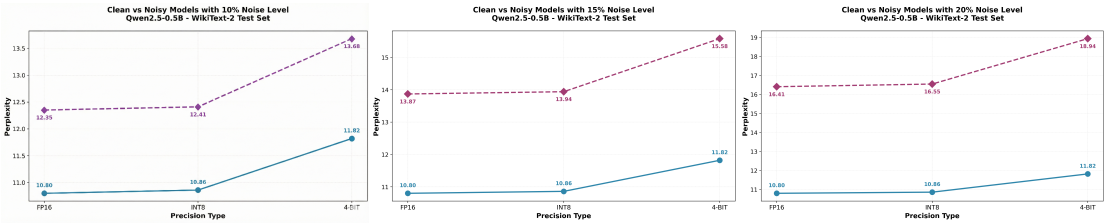

Figure 2: Impact of Gaussian Weight Noise on Qwen2.5-0.5B Perplexity across varying noise intensities(10%, 15%, 20%).

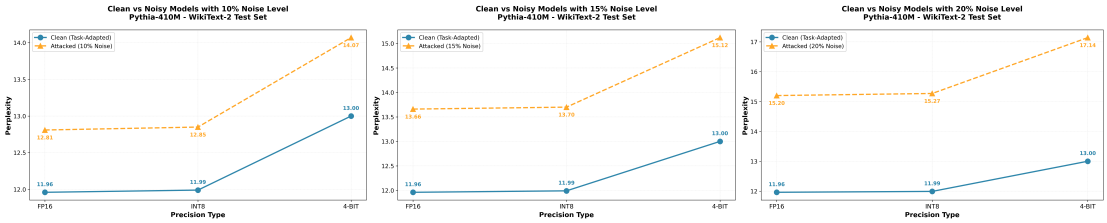

Figure 3: Impact of Gaussian Weight Noise on Pythia-410M Perplexity across varying noise intensities(10%, 15%, 20%).

3.3 Defense Training Through Anchored Fine-Tuning

Defense training uses a dual-model architecture for efficiency. A task-adapted clean baseline acts as a frozen anchor stored on the CPU, while the attacked model is loaded as a trainable instance on the GPU. The objective optimizes a combined loss:

$$L_{total} = L_{CE} + \lambda L_{anchor} \tag{6}$$

where L_{CE} is the cross-entropy loss and L_{anchor} is the L2 distance between trainable weights and frozen anchor weights. The coefficient λ balances performance recovery and parameter stability.

3.4 Multi-Precision Evaluation

The defended model is evaluated under FP16, INT8, and 4-BIT precision. Recovery is measured as:

$$Recovery = \frac{PPL_{attacked} - PPL_{defended}}{PPL_{attacked} - PPL_{clean\_adapted}} \times 100\% \tag{7}$$

3.5 Quantization and Perplexity

Quantization approximates a floating-point weight w with an integer value q using a scaling factor s:

$$q = clip\left(round\left(\frac{w}{s}\right), q_{min}, q_{max}\right) \tag{8}$$

The reconstructed weight is $\hat{w} = s \times q$, with quantization error:

$$\delta_w = \hat{w} - w \tag{9}$$

These errors propagate through transformer layers and can degrade language modeling performance.

Perplexity (PPL) evaluates model quality:

$$PPL = \exp\left(-\frac{1}{N}\sum_{t=1}^{N} \log p(x_t|x_{<t})\right) \tag{10}$$

Due to limited context length, a sliding-window evaluation (2048 tokens, stride 512) is used. Loss is computed only on new tokens, and final perplexity is:

$$PPL = \exp\left(\frac{\sum_i \ell_i}{N}\right) \tag{11}$$

3.6 Data Sources and Evaluation Inputs

The WikiText-2 dataset is used for model adaptation, defense training, and evaluation, processed into 2048-token blocks to match the model's maximum context length. To augment aggregate perplexity assessments, prompt-level robustness is also analyzed using a suite of 96 diverse, instruction-based prompts spanning 12 task categories (e.g., code generation, mathematical reasoning, and summarization).

3.7 Implementation Details of Selected Design

As shown in Figure 4, the defense implementation employs a dual-model architecture. The anchoring mechanism selectively targets 168 critical weight matrices distributed across the model's 24 transformer layers, specifically including the attention projection matrices (q_proj, k_proj, v_proj, o_proj) and MLP feed-forward projection matrices (gate_proj, up_proj, down_proj), while deliberately excluding normalization layers, bias parameters, and embedding matrices. During each training step, the forward pass processes a single 2048-token chunk through the trainable model to compute standard cross-entropy loss (L_{CE}) for next-token prediction. Subsequently, for each of the 168 anchored parameters, the current weight matrix is detached and moved to CPU memory to compute the L2 squared distance against its corresponding frozen anchor weight:

$$L2_{penalty} = mean\left((W_{current} - W_{anchor})^2\right) \tag{12}$$

These individual penalties are accumulated and averaged across all anchored parameters to produce L_{anchor}, which is then combined with the cross-entropy loss using the regularization coefficient $\lambda = 1e-5$, yielding the total loss:

$$L_{total} = L_{CE} + \lambda \times L_{anchor} \tag{13}$$

Parameter optimization is conducted over 200 training steps using the AdamW optimizer with a learning rate of 2e-5, gradient accumulation over 2 steps (effective batch size of 2), and a linear warmup schedule over the first 10 steps. All parameter updates are done with half precision (FP16) to make sure that they are stable in terms of numbers when computing gradients and weight updates, and the model takes approximately 9-10 GB of GPU memory (gradients, optimizer states, and activations). This training program combines about 250 chunks or half a million tokens, representing 24% of the WikiText-2 training data, at each optimization step, updating each of the 494 million parameters with the anchoring constraint specified to the 168 targeted weight matrices. In the evaluation phase, post-training quantization is applied on the defended model, which is tested in the case of. three precision modes: FP16, INT8, and 4-BIT. This low-precision strategy facilitates the evaluation of defense strength in deployment with limited resources, scenarios and at the same time sustaining training stability with full-precision optimization.

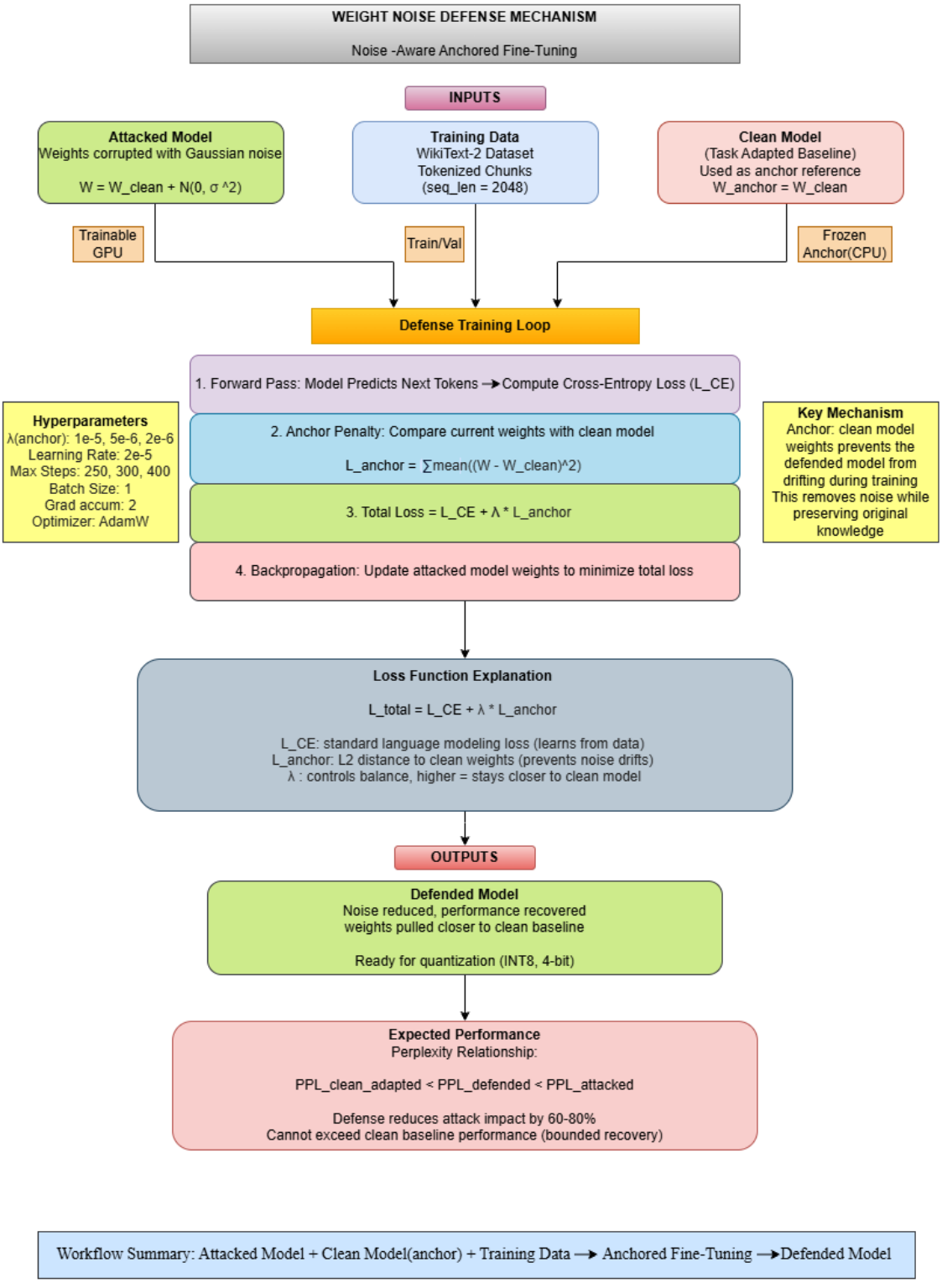

Figure 4: Operational Workflow of the Anchor-Guided Repair Mechanism.

4 Result Analysis

4.1 Performance Evaluation

4.1.1 Qwen2.5-0.5B Results

In table 1, there was a positive correlation between defense effectiveness of Qwen2.5-0.5B and the noise level, with more than 64 percent recovery in the initial condition ($\lambda = 1e-5$, 250 steps), over 67 percent in the less regularized condition ($\lambda = 5e-6$, 300 steps), and optimal performance of about 75 percent recovery with $\lambda = 2e-6$ and 400 steps, which is almost equal to the clean baseline.

Table 1: Qwen2.5-0.5B Performance Results Across Three Defense Configurations

Precision	**Clean**	**Attacked**	**Attempt 1**		**Attempt 2**		**Attempt 3**	
			Def.	**Rec. %**	**Def.**	**Rec. %**	**Def.**	**Rec. %**
10% Perplexity-Calibrated Noise Level								
FP16	10.80	12.35	11.76	38.6%	11.76	38.6%	11.58	49.8%
INT8	10.86	12.41	11.81	38.5%	11.81	38.5%	11.64	49.9%
4-BIT	11.82	13.68	12.87	43.4%	12.87	43.4%	12.66	55.0%
15% Perplexity-Calibrated Noise Level								
FP16	10.80	13.87	12.31	51.0%	12.12	57.0%	11.92	63.5%
INT8	10.86	13.94	12.35	51.6%	12.17	57.4%	11.97	64.0%
4-BIT	11.82	15.58	13.43	57.1%	13.23	62.5%	12.99	68.8%
20% Perplexity-Calibrated Noise Level								
FP16	10.80	16.41	12.81	64.2%	12.61	67.7%	12.39	71.7%
INT8	10.86	16.55	12.87	64.6%	12.67	68.2%	12.44	72.2%
4-BIT	11.82	18.94	14.15	67.3%	13.89	70.9%	13.60	75.0%

Note: A1: $\lambda = 1e{-}5$, 250 steps — A2: $\lambda = 5e{-}6$, 300 steps — A3: $\lambda = 2e{-}6$, 400 steps

4.1.2 Pythia-410M Results

In table 2, only the first run ($\lambda = 1e-5$, 250 steps) demonstrated the presence of limited recovery at low noise but 60 -69% recovery at 20% noise. Reducing regularization also enhanced performance with the optimum setup ($\lambda = 2e-6$, 400 steps) being able to recover roughly 79% in the 4-BIT regime, which demonstrates the critical role of lower anchor strength in stability.

4.2 Final Design Solution and Adjustment

After evaluating Clean-model (which requires an unrealistic clean reference) and Attacked-model (which struggles with zero-shot performance) strategies, the Task-adapted baseline was selected as the optimal design. This final approach fine-tunes on WikiText-2 for 250 steps ($\lambda = 1e-5$), specifically anchoring 168 key projection weights, and optimizes memory efficiency by storing the frozen anchor on the CPU.

Table 2: Pythia-410M Performance Results Across Three Defense Configurations

Precision	Clean	Attacked	Attempt 1		Attempt 2		Attempt 3	
			Def.	Rec. %	Def.	Rec. %	Def.	Rec. %
10% Perplexity-Calibrated Noise Level								
FP16	11.96	12.81	12.75	6.5%	12.65	18.6%	12.46	41.3%
INT8	11.99	12.85	12.79	7.2%	12.69	18.8%	12.49	41.7%
4-BIT	13.00	14.07	13.67	37.4%	13.56	47.5%	13.33	69.6%
15% Perplexity-Calibrated Noise Level								
FP16	11.96	13.66	12.98	39.8%	12.86	46.9%	12.66	59.0%
INT8	11.99	13.70	13.01	40.1%	12.90	47.0%	12.69	59.2%
4-BIT	13.00	15.12	13.96	55.0%	13.82	61.2%	13.58	72.8%
20% Perplexity-Calibrated Noise Level								
FP16	11.96	15.20	13.27	59.5%	13.16	63.1%	12.94	69.8%
INT8	11.99	15.27	13.31	59.8%	13.19	63.5%	12.98	69.9%
4-BIT	13.00	17.14	14.30	68.7%	14.16	72.0%	13.88	78.8%

Note: A1: $\lambda = 1e{-}5$, 250 steps — A2: $\lambda = 5e{-}6$, 300 steps — A3: $\lambda = 2e{-}6$, 400 steps

4.3 Statistical Analysis

4.3.1 Prompt-wise Ordering Consistency

We evaluated whether the strict ordering $PPL_{clean} < PPL_{defended} < PPL_{attacked}$ holds for individual prompts. In Figure 5, for Qwen2.5-0.5B, the defended model successfully outperformed the attacked model in 88.5% to 94.8% of cases.

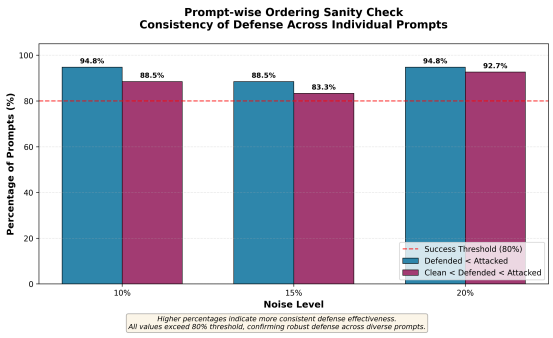

Figure 5: Prompt-wise Ordering Sanity Check (Qwen2.5-0.5B).

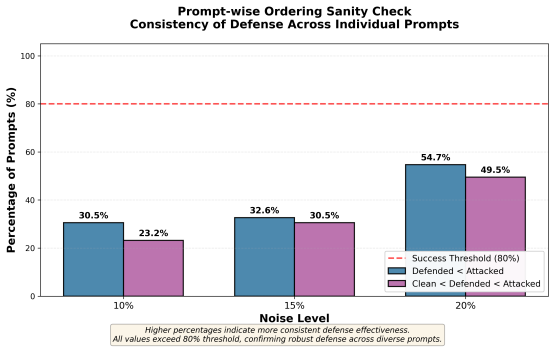

Figure 6: Prompt-wise Ordering Sanity Check (Pythia-410M).

In Figure 6, for Pythia-410M, the defense improved performance in only 30.5% of prompts at low noise, peaking at 54.7% under 20% noise. This shows that Pythia struggles to generalize repair benefits across short, diverse prompts.

4.3.2 Stability / Variance Analysis

Figure 7 shows that for Qwen2.5-0.5B, the defense substantially reduced variance, ranging from 31.5% at 10% Perplexity-Calibrated Noise to 54.3% at 20% Perplexity-Calibrated Noise. The significantly shorter error bars indicate that the defense stabilizes the model.

In Figure 8, Pythia-410M showed negative variance reduction at lower noise levels, with positive stability only restored at the high 20% Perplexity-Calibrated Noise level (14.3% reduction).

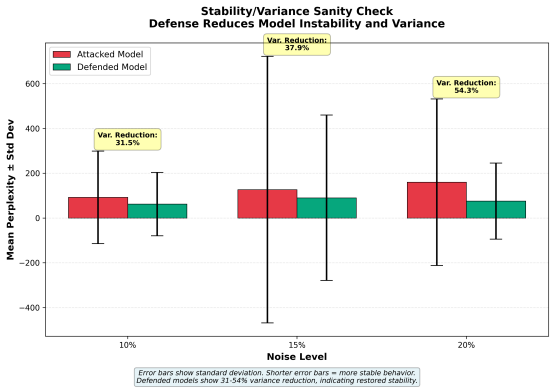

Figure 7: Variance Reduction (Qwen2.5-0.5B).

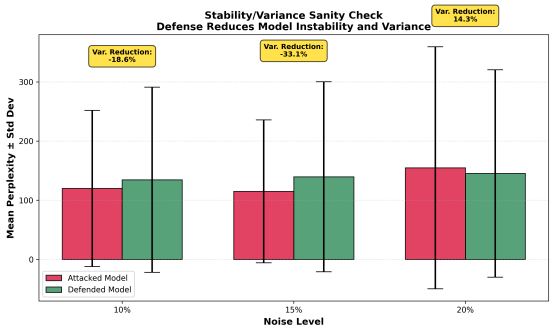

Figure 8: Variance Reduction (Pythia-410M).

4.4 Impact of Instruction Tuning on Defense Stability

Figures 9 and 10 highlight varying defense performance between the two model architectures. The instruction-tuned Qwen2.5-0.5B demonstrates uniform, high recovery across diverse tasks (e.g., logic, coding, reasoning), suggesting that instruction alignment enhances the generalization of Anchor-Guided Repair. Conversely, the base Pythia-410M model exhibits highly skewed and inconsistent recovery across categories. This indicates that the defense mechanism is significantly more reliable for instruction-tuned models than for base pretrained models.

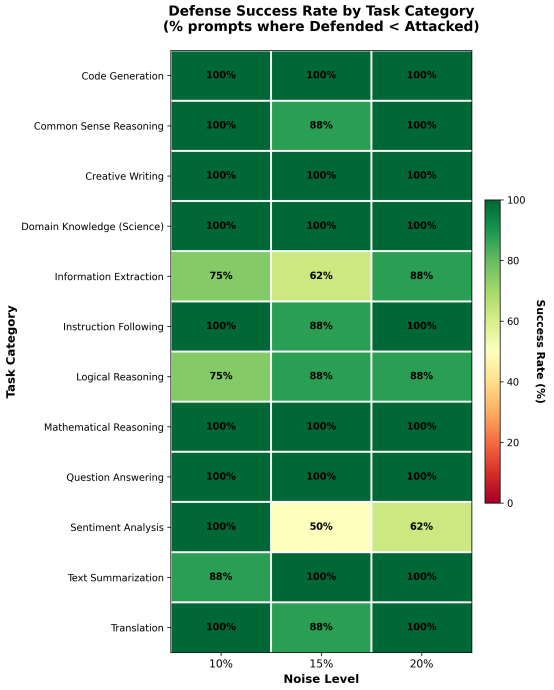

Figure 9: Category-wise Sanity Check Consistency for Qwen2.5B-0.5B

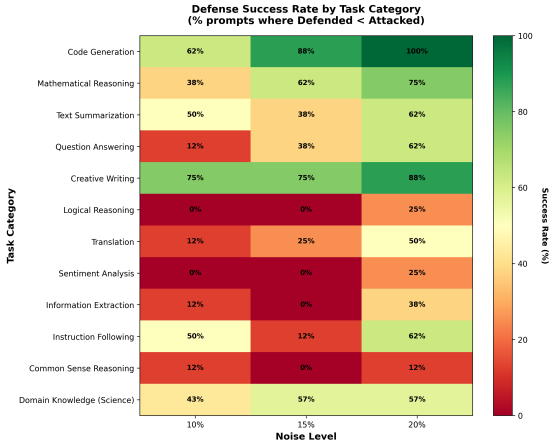

Figure 10: Category-wise Sanity Check Consistency for Pythia-410M

5 Data Availability Statement

Dataset (Wikitext-2, 96-PromptSuite): `https://huggingface.co/datasets/mindchain/wikitext2`, `https://raw.githubusercontent.com/spongebob2409/96-Prompt-Suites-Dataset/refs/heads/main/prompt_suite_96.json`. **Code Repository:** `https://github.com/spongebob2409/Anchor-Guided-Repair.git`.

6 Conclusion

This research addresses the vulnerability of pretrained language models to low-precision quantization and Gaussian weight noise. We propose Anchor-Guided Repair, a mechanism using regularization to tether compromised models to a clean reference, preventing parameter drift without requiring original training data. Under extreme 20% perplexity-calibrated noise and 4-BIT quantization, the defense achieved 75.0% performance recovery for Qwen2.5-0.5B and 78.8% for Pythia-410M. It also effectively restored model reliability, with Qwen2.5-0.5B demonstrating a 54.3% variance reduction and outperforming the attacked model in 94.8% of evaluated prompts. Future research should expand this methodology to larger, multilingual models to test cross-lingual scalability. Furthermore, developing lightweight storage for critical anchor weights to preserve model confidentiality, and improving defense generalization across diverse task distributions, will ensure robust recovery in real-world deployments.

References

Ting-Yun Chang, Miao Zhang, Jesse Thomason, and Robin Jia. Why do some inputs break low-bit llm quantization?, 2025. arXiv:2506.12044, May 2025, https://arxiv.org/abs/2506.12044.

Kejia Chen, Jiawen Zhang, Jiacong Hu, Yu Wang, Jian Lou, Zunlei Feng, and Mingli Song. Q-resafe: Assessing safety risks and quantization-aware safety patching for quantized large language models, 2025. arXiv:2506.20251, https://arxiv.org/abs/2506.20251.

Brian Chmiel, Michal Fishman, Ron Banner, and Daniel Soudry. Fp4 all the way: Fully quantized training of llms, 2025. arXiv:2505.19115, May 2025, https://arxiv.org/abs/2505.19115.

Patrik Czakó et al. Addressing activation outliers in llms: A systematic review of post-training quantization techniques. *IEEE Journals & Magazine*, 2025. Available at: https://ieeexplore.ieee.org/abstract/document/10994764.

Tim Dettmers, Ruslan Svirschevski, Vage Egiazarian, Denis Kuznedelev, Elias Frantar, Saleh Ashkboos, Alexander Borzunov, Torsten Hoefler, and Dan Alistarh. Spqr: A sparse-quantized representation for near-lossless llm weight compression, 2023. arXiv:2306.03078, June 2023, https://arxiv.org/abs/2306.03078.

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

Jahid Hasan. Optimizing large language models through quantization: A comparative analysis of ptq and qat techniques, 2024. arXiv:2411.06084, November 2024, https://arxiv.org/abs/2411.06084.

Xing Hu, Yifei Cheng, Dawei Yang, Zhongzhi Xu, Zheyu Yuan, Jiahe Yu, Chang Xu, Zidong Jiang, and Shihao Zhou. Ostquant: Refining large language model quantization with orthogonal and scaling transformations for better distribution fitting, 2025. arXiv:2501.13987, January 2025, https://arxiv.org/abs/2501.13987.

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 on-device llm compression and acceleration. In *Proceedings of Machine Learning and Systems*, 2024. May 2024.

Zechun Liu, Changsheng Zhao, Igor Fedorov, Bilge Soran, Dhruv Choudhary, Raghuraman Krishnamoorthi, Vikas Chandra, Yuandong Tian, and Tijmen Blankevoort. Spinquant: Llm quantization with learned rotations, 2024. arXiv:2405.16406, May 2024, https://arxiv.org/abs/2405.16406.

Hua Ma et al. Quantization backdoors to deep learning commercial frameworks. *IEEE Journals & Magazine*, 2024. June 2024, https://ieeexplore.ieee.org/abstract/document/10113762.

Vladimir Malinovskii, Andrei Panferov, Ivan Ilin, Hongxia Guo, Peter Richtárik, and Dan Alistarh. Pushing the limits of large language model quantization via the linearity theorem, 2024. arXiv:2411.17525, November 2024, https://arxiv.org/abs/2411.17525.

Jungwoo Park, Taehyeon Lee, Chanho Yoon, Hyunseok Hwang, and Jaewook Kang. Outlier-safe pre-training for robust 4-bit quantization of large language models, 2025. arXiv:2506.19697, June 2025, https://arxiv.org/abs/2506.19697.

Irina Proskurina, Luc Brun, Guillaume Metzler, and Julien Velcin. When quantization affects confidence of large language models, 2024. arXiv:2405.00632, May 2024, https://arxiv.org/abs/2405.00632.

Dongwei Wang and Huanrui Yang. Taming sensitive weights: Noise perturbation fine-tuning for robust llm quantization, 2024. arXiv:2412.06858, December 2024, https://arxiv.org/abs/2412.06858.

Dongwei Wang, Huanrui Yang, et al. Leveraging noise and aggressive quantization of in-memory computing for robust dnn hardware against adversarial input and weight attacks. In *IEEE Conference Publication*, 2021. December 2021, https://ieeexplore.ieee.org/abstract/document/9586233.

Qianli Wang, Mengqi Wang, Nils Feldhus, Sebastian Ostermann, Yuting Cao, Hinrich Schütze, Sebastian Möller, and Vera Schmitt. Through a compressed lens: Investigating the impact of quantization on llm explainability and interpretability, 2025. arXiv:2505.13963, May 2025, https://arxiv.org/abs/2505.13963.

Zhaohui Zhang, Jinyu Jiang, Malu Chen, Zhenguo Wang, Yonghong Peng, and Zhongzhi Yu. A novel noise injection-based training scheme for better model robustness, 2023. arXiv:2302.10802, February 2023, https://arxiv.org/abs/2302.10802.