Fault Tolerance in Transformers Favors Output Alignment over Hidden-State Consistency

Haonan Tan[1] **Yuxing Han**[1] **Jiangtao Wen**[2,*,†]
[1]Tsinghua University [2]NYU Shanghai

Abstract

Latent hardware defects cause a substantial, recurrent class of silent data corruption (SDC) measured at fleet scale: a faulty compute unit returns a wrong value with no error signal. We ask whether a large language model can be *trained* to tolerate such faults, and *where* in the network the consistency signal should be enforced. Consistency-Aware Training (CAT) injects tile-level matrix-multiplication (GEMM) faults during training and adds a Kullback–Leibler (KL) consistency term on the output distribution; on GPT-2 Small it reduces fault-induced perplexity degradation from 12.9% to 0.21% (two-seed mean), a $60\times$ reduction. Most of the raw gain comes from training on the faulted pass, and the placement of the consistency term decides whether it helps or hurts: a hidden-only objective at the fault site raises degradation to 32.7%, and bounded hidden controls that add the corrupted-pass cross-entropy reach 1.74–1.76%, still above output KL (0.84%) at a matched 80k-step horizon. On GPT-2, a local constraint hierarchy and gradient-alignment probes indicate that output-level consistency is less restrictive and easier to optimize than early hidden-state consistency. The output-over-hidden advantage also appears in a single-seed 8B probe, the GPT-2 ordering reproduces on held-out corpora, and generic robustness methods (sharpness-aware minimization, R-Drop, and tanh weight clamping) do not close the gap. In a simulated evaluation of a production-deployable detection-to-erasure interface, pairing the trained model with algorithm-based fault tolerance (ABFT) detection turns raw bit flips into erasures with a $1.10\times$ perplexity ratio, where unprotected inference produces NaN.

1 Introduction

Latent hardware defects cause a substantial, recurrent class of silent data corruption (SDC) measured at fleet scale [Hochschild et al., 2021, Dixit et al., 2021, Ma et al., 2025]: a faulty accelerator produces wrong outputs with no error signal, so the computation proceeds and its result is trusted [George et al., 2025]. Fail-stop hardware failures at LLM scale [Grattafiori et al., 2024, Zhang et al., 2022, Jiang et al., 2024] can be recovered by detection and checkpointing; SDC is the harder residual case, because a single corrupted tile of a matrix multiplication can silently alter every downstream activation. In large language model (LLM) inference, every autoregressive step passes billions of parameters through Tensor Core general matrix multiplications (GEMMs); we study how much of this corruption the model itself can absorb.

We ask whether an LLM can be *trained* to tolerate such faults, and where the consistency signal should be enforced. We train models with identical fault injection while varying the alignment target (output logits or intermediate hidden states) and find that output alignment is consistently best

*Corresponding author: Jiangtao Wen (jw9263@nyu.edu).
†Project lead.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

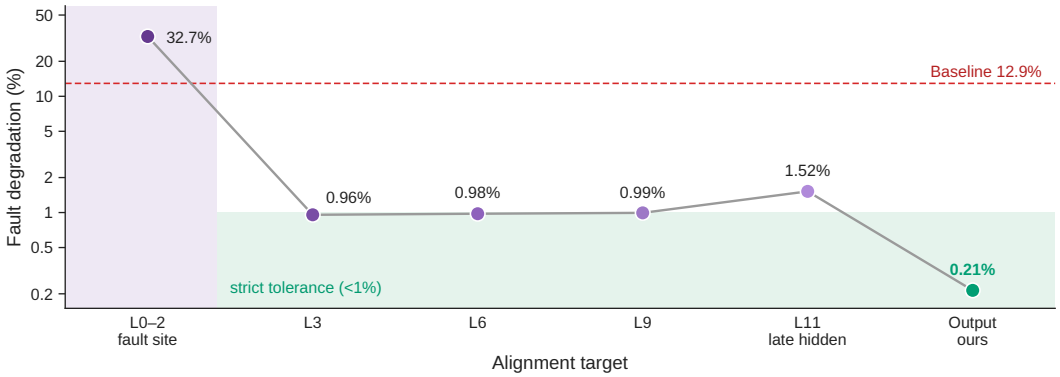


Figure 1: **Fault tolerance is localized by the alignment target.** GPT-2 Small degradation under the top-3 QK single-position protocol (one 16×16 output tile of the query–key attention-score GEMM is corrupted, at each of the three most vulnerable positions in turn) as the consistency target moves from the fault site (L0–2) to the output distribution. The dashed line marks the unprotected two-seed baseline (12.9%); the green band marks the strict tolerance regime ($< 1\%$). Training horizons differ: L0–2 402k steps, L3–L9 80k, L11 40k, output 603k/930k (two seeds); at a matched 80k horizon output KL reads 0.84% (Table 2). Moving one layer past the fault site eliminates the hidden-only catastrophic regime.

(Figure 1). Aligning at the output logits reduces perplexity degradation to 0.21% (two-seed mean); aligning at early hidden layers *increases* degradation to 32.7%, $2.5\times$ worse than the unprotected baseline. Hidden targets placed after the fault site avoid this failure but stay above output alignment at a matched 80k-step horizon (0.96–0.99% vs. 0.84%). Because that early hidden-only objective omits the cross-entropy (CE) loss on the corrupted pass, we add fair bounded hidden controls; they remove the catastrophic failure but still trail output KL at the same 80k horizon (1.74–1.76% vs. 0.84%).

The *method* we study, Consistency-Aware Training (CAT), injects tile-level GEMM faults and adds a Kullback–Leibler (KL) consistency term at the output; it is deliberately minimal, a fault-specific variant of R-Drop (regularized dropout) [Liang et al., 2021] with a different perturbation source. We use it to study placement: under bounded tile faults, where the consistency signal is placed strongly changes the achievable tolerance, and we evaluate the trained models in a simulation of a production-deployable detection-to-erasure interface (§2). Relative to fault-aware training without a consistency term [Zahid et al., 2020] and to output-level distillation in quantization-aware training [Liu et al., 2024], the new elements are the placement result and the detection-to-erasure interface; Table 6 states what is inherited and what is new.

Our contributions are:

1. **Discovery (§3).** Output-distribution alignment achieves 0.21% fault degradation on a 124M-parameter transformer (two-seed mean), $60\times$ lower than the unprotected baseline. A direct fault-site hidden-only objective raises degradation to 32.7% on GPT-2, while matched bounded hidden controls improve to 1.74–1.76% but remain worse than output KL at the same 80k horizon; the output-over-hidden advantage also appears in a single-seed 8B probe, although the 8B four-variant order is not identical to GPT-2.

2. **Local geometry and mechanism evidence (§4, §3.3).** A local constraint-geometry result gives conditions under which output constraints strictly contain hidden constraints, and mechanism probes show cooperative output-level gradients ($\alpha \in [0.36, 0.55]$ on GPT-2) versus near-orthogonal fault-site hidden gradients ($\alpha \approx 0$). Sharpness-aware minimization (SAM) [Foret et al., 2021], R-Drop, and saturated activation functions (SAF) [Fuengfusin and Tamukoh, 2024] fail to match CAT-Output, so generic regularization is not sufficient.

3. **Detection–tolerance pipeline (§3.4).** Pairing CAT-Output with an algorithm-based fault tolerance (ABFT) tile detector [Huang and Abraham, 1984] converts raw bit-flips into zero-fill erasures; at $n{=}1024$ corrupted tiles, the simulated detection-to-erasure interface yields a $1.10\times$

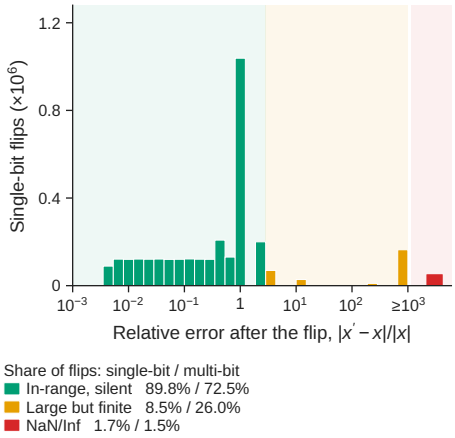


Figure 2: **Outcome of a bit-flipped bf16 value**, by exact enumeration over 2×10^5 unit-scale values. Multi-bit shares weight flips by the multiplicity distribution of Tung et al. [2026] (Appendix E). The in-range class stays the majority; NaN/Inf, the only class a special-value screen sees, shrinks under multi-bit flips. Shares are per value, not per SDC event.

perplexity (PPL) ratio where raw bit-flips produce NaN. Both detection mechanisms have real-hardware precedent [Comanici et al., 2025, Gao et al., 2026].

2 Problem Setup and Method

Fault model. We inject faults at the granularity of 16×16 GEMM output tiles, the unit at which checksum verification and tile replacement operate in recent ABFT proposals [Huang and Abraham, 1984, Gao et al., 2026, Xie et al., 2025, Xue et al., 2025], on the four GEMMs per transformer layer (QK, OV, FC_1, FC_2). During training, with probability $p_{\text{fault}} = 0.5$ per minibatch we execute a second forward pass in which one structural tile index is sampled from the three most vulnerable GPT-2 QK tiles, $(\ell, \text{gemm}, r, c) \in \{(1, \text{QK}, 0, 0), (0, \text{QK}, 0, 0), (2, \text{QK}, 0, 0)\}$, and replaced with i.i.d. Gaussian noise $\mathcal{N}(0, \sigma^2)$, σ the standard deviation of the GEMM output. This rate is deliberately far above realistic datacenter SDC; we treat it as an aggressive regularizer that gives the consistency term a training signal on half of the steps. Headline evaluations average over the same three single-position QK corruptions, one active at a time; density sweeps separately use $n \in \{1, 2, 4, \ldots, 8192\}$ tiles per forward pass to span realistic single-tile corruption to near-saturation. The fault distribution is a stress model rather than an empirical prior: the three QK positions are the vulnerability-weighted worst case (Appendix J), and no reference we know of establishes an SDC preference for QK GEMMs. Gaussian tiles model in-range residual error and zero-fill tiles model a detected tile that is erased (raw bit-flips in §3.4); Appendix D states which parts of this model are grounded, measured, or our own construction.

Why this fault model. Latent hardware defects cause a substantial, recurrent class of SDC measured at fleet scale: fleet studies of server CPUs find silently miscomputing devices at a prevalence on the order of 10^{-3} [Hochschild et al., 2021, Dixit et al., 2021], and frontier LLM training reports recurring SDC incidents [Altenbernd et al., 2026], with GPU SDC among the root causes of Llama 3 training interruptions [Grattafiori et al., 2024]. In large-scale training, SDCs "usually result from latent hardware defects" [Ma et al., 2025], which corrupt computation at fixed physical sites [Tyagi et al., 2026, Tung et al., 2026]; that the corrupted *tile coordinate* is also fixed at inference, where kernel schedules are static, is our inference. The corruption is silent because error-correcting codes protect storage rather than the arithmetic datapath [Hochschild et al., 2021, Dixit et al., 2021, George et al., 2025]. Of bf16 values corrupted by bit flips, 89.8% (single-bit) or 72.5% (multi-bit, with the flip multiplicity measured by Tung et al. [2026]) stay within relative error 2 and raise no exception (Figure 2); this is the class our channels instantiate, and real SDC-affected nodes show such errors as "relatively small" perturbations [Ma et al., 2025]. Only 1.5–1.7% become NaN/Inf, the sole class a special-value screen sees. Our experiments are a *simulated evaluation of a production-deployable detection-to-erasure interface*: per-device checksum SDC detection runs in production [Comanici et al., 2025] and tile-granularity ABFT checksums are demonstrated on real GEMMs [Gao et al., 2026], while controlled parameter sweeps are not feasible on faulty hardware [Altenbernd et al., 2026].

Training objective. Let $f_\theta(x)$ denote a clean forward pass and $f_{\theta,\phi}(x)$ a fault-injected forward pass on the same input x under sampled tile fault ϕ. We call the method *Consistency-Aware Training* (CAT); on active-fault minibatches, its main variant CAT-Output uses:

$$\mathcal{L} = \mathrm{CE}(f_\theta(x), y) + \mathrm{CE}(f_{\theta,\phi}(x), y) + \lambda \cdot \mathrm{KL}(\mathrm{sg}[p_\theta(x)] \,\|\, p_{\theta,\phi}(x)) \,, \tag{1}$$

where $p_\theta(x) = \mathrm{softmax}(Wh_L)$, $p_{\theta,\phi}(x)$ is the fault-injected output distribution, $\mathrm{sg}[\cdot]$ denotes stop-gradient on the clean prediction in the consistency term, and $\lambda = 0.1$ is the KL weight (sensitivity sweep in §3.3). When the Bernoulli fault is inactive, only the clean CE term is evaluated; Eq. 1 is the conditional loss used on active-fault steps. The only design choice we vary is the *alignment target*: the output distribution (**Output-aligned**) versus an intermediate hidden state h_ℓ (**Hidden-aligned**, with a layer subscript when needed).

Hidden-aligned variants replace the output KL distance with the implemented representation loss $C_{\mathrm{hid}}^{(\ell)} = 1 - \cos(R_\ell \tilde{h}_\ell, R_\ell \,\mathrm{sg}[h_\ell]) + \beta(\|R_\ell \tilde{h}_\ell\| - \|R_\ell \,\mathrm{sg}[h_\ell]\|)^2$, with fixed orthogonal R_ℓ (identity in the reported runs), $\beta = 0$ for late hidden-layer sweeps and $\beta = 0.1$ for CAT-Only at L0–L2. The original L0–L2 CAT-Only row is a diagnostic hidden-only objective that isolates whether direct local representation repair at the injection site can by itself absorb a fault. Because this objective is stricter than CAT-Output (it omits $\mathrm{CE}(f_{\theta,\phi}(x), y)$ and the norm term is unbounded), §3.1 reports matched fair controls that add corrupted CE and use bounded or normalized hidden losses. Algorithm 1 in Appendix B lists the full minibatch procedure.

Metric. We report *fault degradation* $(\%) = (p_{\mathrm{corrupt}}/p_{\mathrm{clean}} - 1) \times 100$; lower is better, zero is fault-immune. We use perplexity because the consistency objective acts on the output distribution; Appendix G adds downstream accuracy.

Models and data. Primary experiments use GPT-2 Small [Radford et al., 2019] (124M parameters, 12 layers) on WikiText-103 [Merity et al., 2017], fine-tuned with AdamW [Loshchilov and Hutter, 2019] for 40k–935k dataloader steps (per-row horizons in Table 1; one pass over the shuffled WikiText-103 text-line minibatches, padded/truncated to at most 512 tokens, is 187k steps). Scale validation uses LLaMA-3.1-8B [Grattafiori et al., 2024] under 7-GPU FSDP; a cross-family check on TinyLlama-1.1B [Zhang et al., 2024] appears in Appendix X.

3 Experiments

All experiments use the fault model and metric of §2; the main comparison varies the alignment target of the consistency term, and §3.3 varies the underlying training method.

3.1 Main Result: The Output–Hidden Asymmetry

Table 1 reports alignment targets and loss-term ablations on the 124M backbone under the same top-3 QK evaluation protocol.

CAT-Output reaches 0.21% fault degradation (two-seed mean), a $60\times$ reduction from the unprotected baseline (12.9%), at a clean-perplexity cost of $+1.11$. At the opposite extreme, the direct hidden-only diagnostic at the fault-injection site (CAT-Only) produces 32.7% degradation, $2.5\times$ *worse* than the same backbone with no consistency training at all. Final-layer alignment (CAT-Hidden-L11, a 40k-step run) reaches 1.52%. The two loss-term ablations, CAT-Stability (KL only) and CAT-CE-Only (corrupted CE only), reach 0.46% and 1.01%: either loss term on the faulted pass gives most of the gain, and combining them gives the lowest degradation.

The CAT-Only row tests the natural "repair the representation before it propagates" hypothesis in isolation, but it omits corrupted CE and its unbounded norm term can spike, so Table 2 adds fair bounded controls at the same 80k horizon. Removing the norm term alone does not help (L0–L2 cosine $\beta = 0$: 22.95%), while adding corrupted CE removes the catastrophic regime (1.74–1.76%). However, these direct hidden controls still trail output KL at the same horizon (0.84%) and do not improve on the strong CE(corrupt)-only baseline (1.07%). Projecting the hidden loss onto output-relevant directions narrows the gap without closing it: CAT-Hidden-JVP reaches 1.39%, better than the bounded hidden controls but not better than CE(corrupt)-only (1.07%) or output KL (0.84%), so output-relevant directions explain part of the gap; we attribute the remainder to the softmax nonlinearity, which a first-order projection does not capture.

Table 1: **GPT-2 Small fault tolerance by alignment target.** Fault degradation (%) = $(\text{ppl}_{\text{corrupt}}/\text{ppl}_{\text{clean}} - 1) \times 100$; lower is better. All rows use the top-3 QK single-position protocol. Green rows satisfy the strict tolerance band Deg. $< 1\%$; red rows are worse than no consistency training. CE_c/CE_f: cross-entropy on the clean/faulted pass (glossary in Appendix C). Error bars for Baseline and CAT-Output are two-seed mean $\pm$ rounded half-range; other rows are single-seed ablations. Training horizons differ by row: CAT-Output 603k/930k steps (two seeds), CAT-Stability 745k, CAT-CE-Only 187k, CAT-Hidden-L11 40k, CAT-Only 402k, Baseline 873k/935k; Table 2 compares arms at a matched 80k horizon.

Variant	Loss terms	Deg. (%)	$\text{ppl}_{\text{clean}}$	$\text{ppl}_{\text{corrupt}}$	Δ_{clean}
Output-aligned (ours)					
CAT-Output	CE_c+CE_f+KL	**0.21 ± 0.01** ✓	17.86	17.90	+1.11
CAT-Stability	CE_c+KL	0.46 ✓	18.99	19.08	+2.23
CAT-CE-Only	CE_c+CE_f	1.01	18.42	18.60	+1.66
Hidden-aligned (ours, ablation)					
CAT-Hidden-L11	CE_c+CE_f+$C_{\text{hid}}^{(11)}$ (β=0)	1.52	20.41	20.72	+3.65
CAT-Only (L0–2)	CE_c+$C_{\text{hid}}^{(0:2)}$ (β=0.1)	32.7 ✗	22.18	29.43	+5.42
No consistency training					
Baseline	CE_c only	12.9 ± 0.3 ✗	16.75	18.92	0.00

Table 2: **Fair early-hidden controls at 80k steps.** All rows use seed 42 and the top-3 QK protocol. Fair controls replace the unbounded norm penalty of CAT-Only (Table 1) with bounded or normalized hidden losses; the L0–2 rows other than the first add corrupted CE. CAT-Hidden-JVP projects the layer-2 hidden perturbation to logits with the downstream Jacobian J_F. A second seed reproduces the ordering of the hidden controls (Appendix G).

Variant	Loss terms	Deg. (%)	$\text{ppl}_{\text{clean}}$/$\text{ppl}_{\text{corrupt}}$
CAT-Output (80k)	CE_c+CE_f+KL	**0.84**	20.64 / 20.81
CE(corrupt)-only (80k)	CE_c+CE_f	1.07	19.27 / 19.48
L0–2 cosine $\beta = 0$	CE_c+C_{hid}	22.95	18.77 / 23.08
L0–2 CE_f+cosine $\beta = 0$	CE_c+CE_f+C_{hid}	1.74	18.82 / 19.15
L0–2 CE_f+LN-MSE	CE_c+CE_f+C_{LN}	1.76	18.82 / 19.15
CAT-Hidden-JVP (L2)	CE_c+CE_f+$\|J_F \Delta h\|^2$	1.39	19.05 / 19.32

Training dynamics rule out simple undertraining for CAT-Only: its degradation falls to about 9% by 80–100k steps, then rises after norm-term spikes from step 103k and peaks at 65.9% at step 133.5k (Appendix T); §4.2 relates the failure to gradient alignment.

Placement sweep across layers. Filling in intermediate layers L3, L6, L9 (80k steps each) yields the two-step structure summarized in Figure 1: moving the target one layer past the injection site eliminates the original hidden-only catastrophic regime (L0–2: 32.7%; L3–L9: 0.96–0.99%), and output alignment is lower still, both at the matched 80k horizon (0.84%) and at full horizon (0.21%). At a matched horizon the margin over post-fault hidden targets is modest; the large gaps are against fault-site hidden alignment and no consistency training (Appendix H).

Corpus-robustness of the ordering. The four-variant ordering CAT-Output $<$ CAT-Hidden-L11 $<$ Baseline $<$ CAT-Only reproduces on WikiText-2, PTB [Marcus et al., 1993], and LAMBADA [Paperno et al., 2016] under the same top-3 QK protocol (mean degradation -0.69%, 2.56%, 9.03%, and 40.49% respectively; Table 18 in Appendix M).

3.2 Scale Validation: LLaMA-3.1-8B

We replicate the variants on the 8B scale-up (32 layers, single seed) at two training horizons, a 5k-step probe (8.96M tokens, four variants) and a 50k-step run ($10\times$ more tokens, two variants), to test whether the output-over-hidden ordering emerges early and whether the advantage widens with

Table 3: **LLaMA-3.1-8B scale-up.** Single seed; top-3 QK Gaussian single-position average over n=512 WikiText-103 validation samples. The 5k-step probe preserves the output-over-hidden ordering, while CAT-Only and Baseline swap places. The 5k rows use the in-training evaluation at a 256-token context, the Pretrained row the 256-token position scan (n=64), and the 50k rows the 1024-token protocol of Table 23; absolute PPL is comparable only within a block (Appendix W).

Variant	Steps	Deg. (%)	vs. Baseline	$\text{ppl}_{\text{clean}}$	$\text{ppl}_{\text{corrupt}}$
Short horizon: four-variant probe (256-token context)					
CAT-Output	5,000	**4.6**	9.2×	**11.11**	11.61
CAT-Hidden (LN)	5,000	7.3	5.7×	11.37	12.20
CAT-Only	5,000	27.5	1.5×	11.30	14.42
Baseline	5,000	41.7	—	11.38	16.13
Pretrained (no fine-tune)	—	—	—	16.95	—
Long horizon: two-variant continuation (1024-token context)					
CAT-Output	50,000	**2.1**	**12.8**×	**13.97**	14.27
Baseline	50,000	27.2	—	16.13	20.52

Table 4: **Baseline comparisons on GPT-2 Small.** No alternative matches CAT-Output without fault-specific output alignment.

Method	Mechanism	Deg. (%)	vs. CAT-Output
CAT-Output	Fault-injected output KL	**0.21**	1×
SAM [Foret et al., 2021]	Parameter-space flatness	∼8.8	41× worse
R-Drop [Liang et al., 2021]	Dropout consistency	6.3	29× worse
SAF [Fuengfusin and Tamukoh, 2024]	Tanh weight clamping	10.05	47× worse
Baseline	None	12.9	60× worse

extended training. Fault model, top-3 positions, and FSDP configuration are held fixed; the global batch is small (7 sequences of length 256) because the run is memory-bound under 7-GPU FSDP.

At the 5k probe the output-over-hidden advantage holds ($4.6\% < 7.3\% < 27.5\%$), though CAT-Only (at 8B a mid-stack target, Appendix C) now sits *below* the unprotected baseline (41.7%): at this scale the baseline degrades more than hidden alignment, so the lower two variants swap places while the output-over-hidden ordering is preserved. The unprotected 8B model is more fault-sensitive (41.7% vs. 12.9% at 124M), consistent with deeper networks compounding per-layer perturbations, yet CAT-Output already gives a $9.2\times$ reduction at 5k steps. At 50k steps CAT-Output degrades 2.1% against 27.2% ($12.8\times$); on the in-training harness of the 5k rows the ratio grows from $9.2\times$ to $15.2\times$, and there CAT-Output's 50k clean PPL sits 13% below the pretrained reference while the baseline's drifts 1% above it. With only 1,792 tokens/step, the baseline's clean-PPL drift may reflect high-variance fine-tuning; both rows share batch, sequence length, optimizer, fault sampler, and evaluation protocol.

Robustness of the 8B advantage. Appendix W reports the 8B advantage on held-out corpora, with the protocol of the 50k rows, and on wider tile-position sets, with a separate position-sweep harness. The advantage persists on a wider position set: at 8B, CAT-Output remains lower than baseline on the top-10 fault positions (2.9% vs. 12.8%), which add seven tiles to the training top-3, including two MLP-fc2 positions absent from the sampler. The gap is smaller on this set ($4.4\times$ vs. $11.5\times$ on the training top-3), so we interpret CAT as vulnerability-weighted robustness rather than uniform protection over all operators.

3.3 Ruling Out Alternatives

Table 4 tests whether simpler hypotheses, generic flatness or output consistency without fault injection, explain the asymmetry.

SAM, R-Drop, and SAF respectively test parameter-space flatness, output consistency under the wrong perturbation, and passive weight clamping; all remain 29–47× worse than CAT-Output, so

generic robustness is not sufficient. Conversely, fault-aware training without a consistency term, as in FAT [Zahid et al., 2020], corresponds to our CAT-CE-Only arm (1.01%): most of the raw gain comes from training on the fault channel, and the placement of the consistency term decides whether adding it helps (output) or hurts (early hidden, Table 2).

Hyperparameters, compute, and density. Appendix P shows that CAT-Output remains robust across a 300× KL-weight sweep and across 1–8192 corrupted tiles. Extra compute does not explain the gap either: a clean baseline trained for 10 epochs (1.87M steps) still degrades 11.56%, and at a matched 80k horizon, with the same expected second-pass budget, the bounded alternatives of Table 2 read 1.07–1.76% against 0.84% for CAT-Output. At lower training injection rates ($p_{\text{fault}} \in \{0.1, 0.01\}$) CAT-Output remains the lowest arm, and the KL weight does measurable work only at the output (Appendix H).

3.4 Scope and Limitations of the Fault Model

Our Gaussian and zero-fill tile faults model bounded corruptions *after* hardware-level detection. Raw IEEE-754 bit-flips define the boundary case: at n=1024 with flip probability 10^{-3}, 63–73% of evaluated sequences hit NaN/Inf, where CAT provides no advantage because a single exponent flip can exceed 10^{38}.

The deployment model is therefore layered: ABFT-style checksum methods detect corrupted tiles [Huang and Abraham, 1984, Xue et al., 2025, Xie et al., 2025, Gao et al., 2026]; zero-filling a detected tile converts an unbounded bit-level error into a bounded erasure; CAT-Output absorbs that erasure pattern. In the simulated detection-to-erasure interface (Appendix U), detected-tile zero-fill collapses the bit-flip curve onto the pure-erasure curve, yielding a 1.10× PPL ratio at n=1024 versus 2.64× under Gaussian error and unusable NaN without detection. Under a bit-flip channel calibrated to measured multi-bit statistics [Tung et al., 2026], with NaN/Inf cleared and large finite values left in place, the baseline degrades 17.3% versus 0.54% for CAT-Output at the trained tiles (32×, the same order as Gaussian and zero-fill); without the clear both models reach NaN alike, so detection is a prerequisite CAT does not replace (Appendix E). A train/test mismatch grid narrows the advantage to typically 1.5–2× at unseen positions without systematic inversion (Appendix F); Appendix I converts per-event damage into expected benefit at field event rates. We scope our claims to this detector-converted channel: below detector recall $r \approx 0.9$ no tested training regime is usable, and on a spread missed-error channel CAT-Output and a fine-tuned baseline track each other (1.26× vs. 1.24× at r=0.99). Appendix V covers NaN-clamp-only recovery, recall, false positives, imperfect replacement, and recomputation; non-GEMM kernel faults, adversarial bit-flip attacks, and training-time corruption remain out of scope.

4 Analysis

We now examine why, on GPT-2, direct hidden-state consistency is more restrictive and harder to optimize: §4.1 gives a local output-constraint hierarchy, §4.2 reports gradient and curvature diagnostics, and §4.3 relates tile replacement to erasure/error coding.

4.1 Output Constraint Hierarchy

We formalize the intuition that, under local and architectural conditions, output-level constraints can be strictly more permissive than hidden-state constraints. Let $\mathcal{H}_\epsilon = \{\Delta h \in \mathbb{R}^d : \|\Delta h\| \le \epsilon\}$ denote a hidden-state tolerance ball, and let $\mathcal{O}_\delta = \{\Delta h \in \mathbb{R}^d : \mathrm{KL}(p\|p') \le \delta\}$ denote the perturbations accepted by an output-level KL constraint.

Lemma 1 (Softmax shift invariance)**.** *For any logit vector $z \in \mathbb{R}^V$ and scalar $c \in \mathbb{R}$,*

$$\mathrm{softmax}(z + c \cdot \mathbf{1}) = \mathrm{softmax}(z). \tag{2}$$

Therefore any hidden perturbation satisfying $W\Delta h = c\mathbf{1}$ (where $W \in \mathbb{R}^{V\times d}$ is the unembedding matrix) produces identical output distributions. When $\mathbf{1} \in \mathrm{range}(W)$, the minimum-norm such perturbation is $cW^{+}\mathbf{1}$, not generally $W^{\top}\mathbf{1}$.

This invariance creates a non-trivial null direction: perturbations that change hidden states but leave outputs unchanged. More generally, the output-level KL divergence admits the following bound.

Proposition 1 (Local output constraint hierarchy)**.** *Let $W \in \mathbb{R}^{V\times d}$ be the unembedding head, $F_p = \mathbb{E}_{y\sim p}[(e_y - p)(e_y - p)^\top]$ the Fisher information matrix of $p = \mathrm{softmax}(Wh)$, and $\widetilde{W} = F_p^{1/2}W$ the Fisher-weighted head. For any small perturbation $\Delta h \in \mathbb{R}^d$ the KL divergence admits the second-order expansion*

$$\mathrm{KL}(p\|p') = \tfrac{1}{2}\|\widetilde{W}\,\Delta h\|^2 + O(\|\Delta h\|^3), \tag{3}$$

where $p' = \mathrm{softmax}(W(h + \Delta h))$. Within a local radius in which the cubic remainder is bounded by $K\|\Delta h\|^3$, choosing $\delta \geq \frac{1}{2}\sigma_{\max}(\widetilde{W})^2\epsilon^2 + K\epsilon^3$ gives $\mathcal{H}_\epsilon \subseteq \mathcal{O}_\delta$. If additionally $\mathbf{1} \in \mathrm{range}(W)$, the containment is strict: for any $\epsilon > 0$ there exist Δh with $\|\Delta h\| > \epsilon$ and $\mathrm{KL}(p\|p') = 0$.

Proof sketch. Taylor-expanding KL around $z = Wh$ gives the Fisher-weighted quadratic term in Eq. 3 with a cubic local remainder [Bishop, 1995]. The stated threshold then upper-bounds every $\Delta h \in \mathcal{H}_\epsilon$, proving $\mathcal{H}_\epsilon \subseteq \mathcal{O}_\delta$. Strictness follows from Lemma 1: if $\mathbf{1} \in \mathrm{range}(W)$, there is $\eta \neq 0$ with $W\eta = \mathbf{1}$, and $t\eta$ can have arbitrarily large norm while leaving the output distribution unchanged. The exact range condition holds only approximately in real heads: the constant-logit-shift direction's residual from a strict null direction is 0.6% for GPT-2, 20% for TinyLlama, and 11% for LLaMA-3.1-8B (Table 19), all far below random references. Full proof details and residual-to-leakage bounds are in Appendix N. □

Proposition 1 is a local, approximately satisfied statement rather than a global guarantee: it rests on a second-order expansion with a bounded remainder, and the containment itself is weak, since any sufficiently large δ provides it. Its useful content is the null direction along which the output constraint is cheap.

Geometrically, the hidden constraint $\mathcal{H}_\epsilon$ is a sphere in $\mathbb{R}^d$ while the output constraint $\mathcal{O}_\delta$ is an ellipsoid with a degenerate or near-degenerate axis along $\ker(\widetilde{W})$; perturbations in that direction incur zero or very small KL cost, so an output objective can in principle tolerate hidden-state changes that a representation-matching objective would penalize (App. N, Fig. 9). The range assumption is non-trivial (random-reference residuals are 96.7–99.2%), and a small residual alone does not make the shift axis cheap: under each model's own Fisher metric, the axis is 500–64,000× cheaper than a generic hidden direction on GPT-2 and ∼4× cheaper on LLaMA-3.1-8B, but ∼5× *costlier* on TinyLlama (Table 19). We therefore offer the geometric explanation for the GPT-2 setting, where the main results are measured, and not as a general property of language-model heads.

4.2 Gradient Alignment Dichotomy

The geometric advantage must still be reachable by gradient descent; we treat gradient alignment and curvature as mechanism diagnostics, not formal theorems.

Definition 1. *For layer ℓ, $\alpha_\ell = \cos(\nabla_\theta \mathcal{L}_{\mathrm{task}}, \nabla_\theta C)\big|_{\theta_\ell}$, where $\mathcal{L}_{\mathrm{task}} = \mathrm{CE}(f_\theta(x), y)$ and C is the consistency term.*

Mechanism evidence 1: output gradients cooperate. When C is output-level KL, $\nabla_\theta \mathcal{L}_{\mathrm{task}}$ and $\nabla_\theta C$ share the Jacobian $\partial z/\partial\theta_\ell$ through the unembedding W, biasing α_ℓ toward positive values. Measured on GPT-2, CAT-Output gives $\alpha_\ell \in [0.36, 0.55]$ across all 12 layers.

Mechanism evidence 2: implemented hidden gradients are near-orthogonal at the fault site. When C is the implemented hidden alignment loss $C^{(\ell)}_{\mathrm{hid}}$ on h_ℓ, $\nabla_\theta \mathcal{L}_{\mathrm{task}}$ flows through all downstream attention/MLP blocks and the unembedding, while $\nabla_\theta C$ is dominated by the local clean-vs.-fault representation displacement with the clean target detached. For early layers, this downstream-Jacobian mismatch yields near-orthogonal gradients [Balduzzi et al., 2017]. Measured on GPT-2, CAT-Only (L0–2) gives $\alpha_\ell \in [-0.04, 0.00]$.

Mechanism evidence 3: the hidden-only diagnostic produces high-curvature signals. Let $\lambda_{\mathrm{PI}}(\ell)$ denote the signed Rayleigh estimate returned by our layer-block Hessian power-iteration probe. The sign of α correlates with local curvature magnitude: $\alpha > 0$ (CAT-Output) gives small positive estimates and $0.39\times$ baseline curvature on L0–L2; $\alpha \approx 0$ (CAT-Only) gives unstable high-curvature estimates at the fault-injection layers ($+161$, $+229$, and -645 in the main sweep). Be-

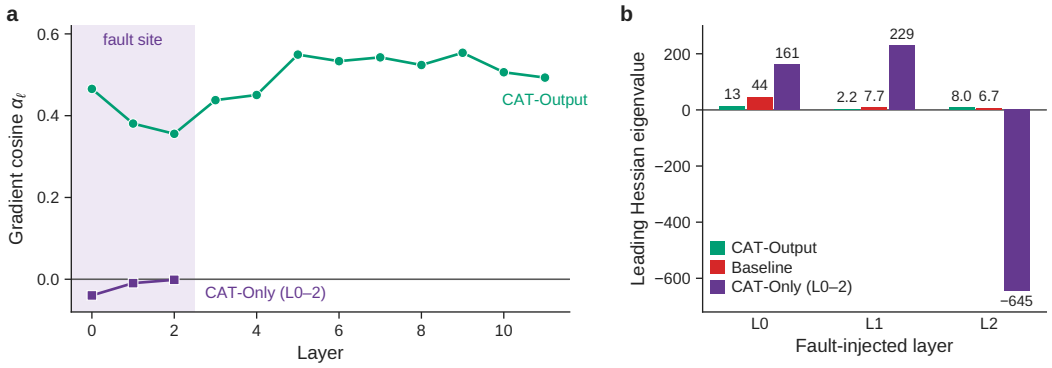


Figure 3: **Mechanism evidence.** (a) CAT-Output maintains positive task–consistency gradient alignment ($\alpha \in [0.36, 0.55]$), while CAT-Only (L0–2) at the fault site is near-orthogonal ($\alpha \approx 0$). (b) The same top-3 layers show the corresponding curvature split: CAT-Output remains small and positive, whereas CAT-Only develops large, sign-sensitive curvature estimates at the fault sites.

Table 5: **Critical tile counts at 5% degradation threshold.** Both channels use the same all-GEMM heterogeneous injection scope (QK/OV/MLP tiles) so the ratio isolates the erasure-vs-error channel.

Model	Gaussian	Zero-fill	Ratio
Pretrained	32	128	$4\times$
Baseline	64	512	$8\times$
CAT-Output	128	1024	$8\times$

cause a rerun at L2 converged to a large positive value, we use these Hessian results as a qualitative instability check, not as proof of the true dominant eigenvalue sign.

Per-layer curvature estimates (App. R, Fig. 12) confine the large CAT-Only values to the injected layers L0–L2, and Appendices Q–S provide power-iteration-free checks and the CAT-Hidden-L11 α paradox. Together they are consistent with gradient alignment being necessary but not sufficient.

4.3 Coding-Theoretic Analysis: Erasure vs. Error

A separate design choice is what replaces a corrupted tile. This choice maps onto the classical distinction between known erasures and unknown errors. In coding theory, a code with minimum distance $d_{\min}$ can correct combinations satisfying $2t + s < d_{\min}$, where t is the number of unknown errors and s is the number of known erasures [MacWilliams and Sloane, 1977]. One unknown error thus costs as much distance as two known erasures; we use this only as a baseline intuition.

We measure the *critical tile count*, the number of simultaneously corrupted tiles at which degradation exceeds 5%, under Gaussian (error) and zero-fill (erasure) replacement. Table 5 shows that erasure is 4–$8\times$ more tolerable than error, exceeding the $2\times$ coding-distance baseline in this transformer fault metric. We attribute the excess to non-linear amplification: a zeroed QK tile produces uniform attention (safe fallback), while a noisy tile produces chaotic attention; a zeroed MLP input silences the neuron (≈pruning), while noise produces random activations; adding zero to the residual stream skips the layer (≈stochastic depth), while adding noise corrupts the representation.

Appendix K extends this to a four-strategy replacement hierarchy; performance improves monotonically as replacement carries more information about the original tile, and CAT's largest absolute gain appears in the hardest Gaussian channel.

5 Related Work

Prior transformer fault-tolerance work separates into inference-time detection and training-time robustness (Appendix Z). Classical ABFT [Huang and Abraham, 1984] and recent neural-network variants [Xue et al., 2025, Xie et al., 2025, Gao et al., 2026] detect, correct, or approximately re-

Table 6: **What CAT inherits and what is new**, per closest prior method.

Prior method	Inherited	New here
FAT	fault injection in training	our CAT-CE-Only arm (λ=0); placement decides if KL helps
LLM-QAT	logit beats hidden alignment	no fixed teacher; fault position never given
R-Drop	two-pass output-KL objective	fault channel, not dropout ($29\times$ gap)
ABFT	tile-level detection	complementary: CAT absorbs the erasures it leaves

cover corrupted matrix operations at inference time; CAT is complementary because it trains the model to absorb the bounded erasures left by cheap recovery. Training-side methods add activation noise [Zhou et al., 2020], clamp weights [Fuengfusin and Tamukoh, 2024], use SAM-style flatness [Foret et al., 2021, Keskar et al., 2017, Dinh et al., 2017], or regularize output consistency under dropout/adversarial perturbations [Liang et al., 2021, Zheng et al., 2016, Miyato et al., 2019, Zhang et al., 2019], building on noise-injection theory [Bishop, 1995]. Silent-corruption fleet reports [Hochschild et al., 2021, Dixit et al., 2021, George et al., 2025], studies of SDC on real LLM-training nodes [Ma et al., 2025, Pei et al., 2026], GPU fault characterizations [Tyagi et al., 2026, Tung et al., 2026], and instruction-level LLM fault studies [Chai et al., 2026, Altenbernd et al., 2026] motivate realistic fault channels, while LM-Fix [Tahmasivand et al., 2025] and BitFlipScope [Karamat et al., 2026] localize or repair bit-flip corruptions at inference time. The general lesson that logit-level alignment can beat hidden-level alignment is prior work [Liu et al., 2024], as is fault-aware training [Zahid et al., 2020]; what is specific here is the fault setting (Table 6).

6 Conclusion

Training on the faulted pass gives most of the raw gain, and the placement of the consistency term decides whether it helps or hurts: output-distribution consistency reduces tile-fault degradation on GPT-2 to 0.21% and keeps an output-over-hidden advantage in a single-seed 8B probe, while fair bounded hidden controls still do not match output KL. Our analysis points to two factors on GPT-2: output constraints are locally more permissive, and output-level gradients cooperate with the task loss where early hidden gradients are near-orthogonal. With ABFT-style detection [Huang and Abraham, 1984, Gao et al., 2026] this gives a layered interface, evaluated here in simulation: detection converts raw bit-flips into erasures, and CAT-Output absorbs them.

Limitations and future work. The results target a strong but explicit post-detection regime: hardware or system software localizes corrupt computation to GEMM tiles and bounds the numerical error or converts it into a known erasure; raw exponent-bit NaN/Inf faults still require upstream detection, and below detector recall $r \approx 0.9$ no training regime we tested is usable. Within it, our evidence is vulnerability-weighted rather than uniform: the sampler focuses on early QK hotspots and held-out positions keep smaller gaps, motivating operator-balanced training. The interface is evaluated in simulation, not on faulty hardware, and the position prior is our construction (Table 9). The 8B evidence is a single-seed 5k-step four-variant probe plus a 50k-step two-variant continuation rather than full pretraining, and CAT uses an expected 1.5 forward passes per training step. The theory is local: Proposition 1 rests on a second-order expansion, its range condition holds only approximately, and its low-cost null direction is absent for the TinyLlama head, so it explains the GPT-2 results rather than all language-model heads. Our hidden-alignment study is not exhaustive; learned projectors, CCA/Procrustes targets, and contrastive losses may narrow the gap.

Broader impact. Learned fault tolerance could lower the cost of reliable LLM inference by reducing the need for hardware redundancy and recomputation. The main risk is misplaced reliance: a model that tolerates detected erasures is not protected against undetected special values or against fault patterns far from its training channel, so deployments in safety-critical settings should keep detection in place and validate tolerance on their own hardware.

Acknowledgments and Disclosure of Funding

Funding. This work received no third-party funding or support.

Competing interests. The authors declare no competing interests.

References

Anton Altenbernd, Philipp Wiesner, and Odej Kao. Exploring silent data corruption as a reliability challenge in LLM training. In *IEEE 26th International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*, pages 157–166, 2026. doi: 10.1109/CCGrid68966.2026.00025.

David Balduzzi, Marcus Frean, Lennox Leary, J. P. Lewis, Kurt Wan-Duo Ma, and Brian McWilliams. The shattered gradients problem: If resnets are the answer, then what is the question? In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 342–350, 2017.

Chris M. Bishop. Training with noise is equivalent to Tikhonov regularization. *Neural Computation*, 7(1):108–116, 1995. doi: 10.1162/neco.1995.7.1.108.

Duo Chai, Zizhen Liu, Shuhuai Wang, Songwei Pei, Cheng Liu, Huawei Li, and Shangguang Wang. Analysis of LLM vulnerability to GPU soft errors: An instruction-level fault injection study. *arXiv preprint arXiv:2601.19912*, 2026.

Gheorghe Comanici et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 1019–1028, 2017.

Harish Dattatraya Dixit, Sneha Pendharkar, Matt Beadon, Chris Mason, Tejasvi Chakravarthy, Bharath Muthiah, and Sriram Sankar. Silent data corruptions at scale. *arXiv preprint arXiv:2102.11245*, 2021.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations (ICLR)*, 2021.

Ninnart Fuengfusin and Hakaru Tamukoh. Enhancing neural network robustness against fault injection through non-linear weight transformations. *arXiv preprint arXiv:2411.19027*, 2024.

Yiheng Gao, Qin Hua, and Zizhong Chen. V-ABFT: Variance-based adaptive threshold for fault-tolerant matrix multiplication in mixed-precision deep learning. *arXiv preprint arXiv:2602.08043*, 2026.

Nishant George, Harish Dattatraya Dixit, Emel Goksu, Bharath Parthasarathy, Amber Huffman, Sudhanva Gurumurthi, Vilas Sridharan, Thiago Macieira, Arani Sinha, Lisa Minwell, Dean Liberty, and Rob Chappell. Silent data corruption in AI. Technical report, Open Compute Project, 2025. URL https://www.opencompute.org/documents/sdc-in-ai-ocp-whitepaper-ver-1-1-final-pdf. Version 1.1.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Peter H. Hochschild, Paul Turner, Jeffrey C. Mogul, Rama Govindaraju, Parthasarathy Ranganathan, David E. Culler, and Amin Vahdat. Cores that don't count. In *Proceedings of the Workshop on Hot Topics in Operating Systems (HotOS)*, pages 9–16, 2021. doi: 10.1145/3458336.3465297.

Kuang-Hua Huang and Jacob A. Abraham. Algorithm-based fault tolerance for matrix operations. *IEEE Transactions on Computers*, C-33(6):518–528, 1984. doi: 10.1109/TC.1984.1676475.

Ziheng Jiang, Haibin Lin, Yinmin Zhong, Qi Huang, Yangrui Chen, Zhi Zhang, Yanghua Peng, Xiang Li, Cong Xie, Shibiao Nong, Yulu Jia, Sun He, Hongmin Chen, Zhihao Bai, Qi Hou, Shipeng Yan, Ding Zhou, Yiyao Sheng, Zhuo Jiang, Haohan Xu, Haoran Wei, Zhang Zhang, Pengfei Nie, Leqi Zou, Sida Zhao, Liang Xiang, Zherui Liu, Zhe Li, Xiaoying Jia, Jianxi Ye, Xin Jin, and Xin Liu. MegaScale: Scaling large language model training to more than 10,000 GPUs. In *21st USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, pages 745–760, 2024.

Muhammad Zeeshan Karamat, Sadman Saif, and Christiana Chamon Garcia. BitFlipScope: Scalable fault localization and recovery for bit-flip corruptions in LLMs. In *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pages 367–382, 2026. doi: 10.1109/HOST68814.2026.11604826.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations (ICLR)*, 2017.

Xiaobo Liang, Lijun Wu, Juntao Li, Yue Wang, Qi Meng, Tao Qin, Wei Chen, Min Zhang, and Tie-Yan Liu. R-Drop: Regularized dropout for neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 10890–10905, 2021.

Zechun Liu, Barlas Oguz, 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. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 467–484, 2024. doi: 10.18653/v1/2024.findings-acl.26.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

Jeffrey Ma, Hengzhi Pei, Leonard Lausen, and George Karypis. Understanding silent data corruption in LLM training. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 20372–20394, 2025. doi: 10.18653/v1/2025.acl-long.996.

Florence Jessie MacWilliams and Neil James Alexander Sloane. *The Theory of Error-Correcting Codes*. North-Holland, 1977.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations (ICLR)*, 2017.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: A regularization method for supervised and semi-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(8):1979–1993, 2019. doi: 10.1109/TPAMI.2018.2858821.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1525–1534, 2016. doi: 10.18653/v1/P16-1144.

Hengzhi Pei, Leonard Lausen, and George Karypis. Connecting the impact of silent data corruption with different training characteristics: An empirical study. *IEEE Micro*, 46(1):44–51, 2026. doi: 10.1109/MM.2025.3642709.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. Technical report, OpenAI, 2019.

Noam Shazeer and Mitchell Stern. Adafactor: Adaptive learning rates with sublinear memory cost. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 4596–4604, 2018.

Ahmad Tahmasivand, Noureldin Zahran, Saba Al-Sayouri, Mohammed Fouda, and Khaled N. Khasawneh. LM-Fix: Lightweight bit-flip detection and rapid recovery framework for language models. In *Proceedings of the IEEE International Conference on Computer Design (ICCD)*, pages 432–440, 2025. doi: 10.1109/ICCD65941.2025.00068.

Chung-Hsuan Tung, Yanxiang Huang, Nirmal Saxena, Philip Shirvani, Saurabh Hukerikar, Twinkle Jain, Abhishek Tyagi, and Sanjay Gongalore. The anatomy of silent data corruption: GPU error pattern study and modeling guidance. In *56th Annual IEEE International Conference on Dependable Systems and Networks – Supplemental Volume (DSN-S)*, pages 15–21, 2026. doi: 10.1109/DSN-S70715.2026.00018.

Abhishek Tyagi, Saurabh Hukerikar, Nirmal Saxena, Yanxiang Huang, Philip Shirvani, Chung-Hsuan Tung, and Yuhao Zhu. LLM-PRISM: Characterizing silent data corruption from permanent GPU faults in LLM training. In *International Conference for High Performance Computing, Networking, Storage, and Analysis (SC)*, 2026. arXiv:2604.10390.

Tong Xie, Jiawang Zhao, Zishen Wan, Zuodong Zhang, Yuan Wang, Runsheng Wang, Ru Huang, and Meng Li. ReaLM: Reliable and efficient large language model inference with statistical algorithm-based fault tolerance. In *62nd ACM/IEEE Design Automation Conference (DAC)*, pages 1–7, 2025. doi: 10.1109/DAC63849.2025.11132610.

Xinghua Xue, Cheng Liu, Feng Min, Tao Luo, and Yinhe Han. ApproxABFT: Approximate algorithm-based fault tolerance for neural network processing, 2025. arXiv:2302.10469v5.

Ussama Zahid, Giulio Gambardella, Nicholas J. Fraser, Michaela Blott, and Kees Vissers. FAT: Training neural networks for reliable inference under hardware faults. In *IEEE International Test Conference (ITC)*, pages 1–10, 2020. doi: 10.1109/ITC44778.2020.9325249.

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P. Xing, Laurent El Ghaoui, and Michael I. Jordan. Theoretically principled trade-off between robustness and accuracy. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, pages 7472–7482, 2019.

Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. TinyLlama: An open-source small language model. *arXiv preprint arXiv:2401.02385*, 2024.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. OPT: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

Stephan Zheng, Yang Song, Thomas Leung, and Ian Goodfellow. Improving the robustness of deep neural networks via stability training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4480–4488, 2016. doi: 10.1109/CVPR.2016.485.

Chuteng Zhou, Prad Kadambi, Matthew Mattina, and Paul N. Whatmough. Noisy machines: Understanding noisy neural networks and enhancing robustness to analog hardware errors using distillation. *arXiv preprint arXiv:2001.04974*, 2020.

A Training Details

All GPT-2 experiments use the following setup: GPT-2 small (124M parameters, 12 layers, $d_{\text{model}} = 768$, 12 heads) on WikiText-103 [Merity et al., 2017]. Optimization uses AdamW [Loshchilov and Hutter, 2019] with learning rate 2×10^{-5}, cosine decay to 1×10^{-6}, weight decay 0.01, and batch size 4 with sequence length 512. The dataloader shuffles WikiText-103 text lines and pads/truncates each example to at most 512 tokens, one pass over this line-level dataloader is $186{,}980$ steps, so the 873k-step run covers about 4.7 passes and the 930k–935k runs five, and step counts do not translate into steps $\times\, 4 \times 512$ non-padding tokens. Fault injection probability is $p_{\text{fault}} = 0.5$ per minibatch; when triggered, one structural tile index is sampled from the top-3 QK hotspots, $(1, \text{QK}, 0, 0)$, $(0, \text{QK}, 0, 0)$, or $(2, \text{QK}, 0, 0)$, and corrupted with Gaussian replacement. The CAT-Output consistency weight is $\lambda = 0.1$. Training runs on a single NVIDIA A100 GPU.

LLaMA-3.1-8B experiments use 7-GPU FSDP with global batch size 7, sequence length 256, Adafactor [Shazeer and Stern, 2018] learning rate 1×10^{-5}, KL weight 0.1, and the same $p_{\text{fault}} = 0.5$ single-position top-3 QK sampler. The resulting $1{,}792$ tokens/step is a memory-constrained, high-variance fine-tuning regime; we use it for controlled scale validation and explicitly report the clean-PPL drift in §3.2. Table 3 reports both a $5{,}000$-step four-variant probe and a $50{,}000$-step two-variant continuation.

Training rate vs. evaluation protocol. The training rate $p_{\text{fault}} = 0.5$ per minibatch is an aggressive regularizer, not an estimate of field SDC rates. Field SDC is orders of magnitude rarer than this training rate: fleet studies of server CPUs report silently miscomputing devices at a prevalence on the order of 10^{-3} [Hochschild et al., 2021, Dixit et al., 2021]. We deliberately oversample faults during training so that the consistency term receives a training signal on half of the steps; a rate low enough to match deployment would make most gradient updates identical to a no-fault run. Evaluation decouples from this choice: headline tables average the three top-3 QK single-position corruptions, while density sweeps use controlled tile counts $n \in \{1, 2, 4, \ldots, 8192\}$ and report degradation curves (Section 4.3 and Appendices H and P). Results below $n = 64$ are most comparable to realistic single-event SDC regimes; higher counts stress-test the learned tolerance rather than project to deployment.

B Algorithm: Consistency-Aware Training

Algorithm 1 specifies the per-minibatch CAT-Output training loop. Hidden-aligned variants replace line 8's output KL with $C^{(\ell)}_{\text{hid}}$ from §2; CAT-Hidden-L3/L6/L9/L11 retain line 7 and use $\beta = 0$, while the original CAT-Only diagnostic at L0–L2 omits line 7 and uses $\beta = 0.1$ to isolate local representation repair. The fair-control rows in Table 2 restore line 7 and use either $\beta = 0$ cosine or LayerNorm-normalized MSE.

Algorithm 1 Consistency-Aware Training (CAT-Output) for one minibatch.

Require: model f_θ with unembedding head W; minibatch (x, y); fault sampler ϕ over top-3 QK positions; KL weight λ=0.1; fault probability p_{f}=0.5.

1: $z, h_L \leftarrow f_\theta(x)$ ▷ clean forward pass; logits $z = Wh_L$
2: $L_{\text{clean}} \leftarrow \text{CE}(\text{softmax}(z), y)$
3: **if** $\text{Bernoulli}(p_{\text{f}}) = 1$ **then** ▷ fault-injected second pass (shared θ)
4: position $\leftarrow \phi(x)$ ▷ one top-3 QK tile, Gaussian replacement ($\sigma = \text{std}$ of the clean GEMM output)
5: $\tilde{z}, \tilde{h}_L \leftarrow f_\theta(x;\ \text{fault} = \text{position})$
6: $L_{\text{corrupt}} \leftarrow \text{CE}(\text{softmax}(\tilde{z}), y)$
7: $L_{\text{cons}} \leftarrow \text{KL}(\text{sg}[\text{softmax}(z)] \,\|\, \text{softmax}(\tilde{z}))$ ▷ **at the output**, clean target detached
8: **else**
9: $L_{\text{corrupt}}, L_{\text{cons}} \leftarrow 0$
10: **end if**
11: $L \leftarrow L_{\text{clean}} + L_{\text{corrupt}} + \lambda\, L_{\text{cons}}$
12: $\theta \leftarrow \theta - \eta\, \nabla_\theta L$ ▷ AdamW; gradient shared across clean and faulted pass

C Glossary of Variants, Fault Channels, and Protocol Tags

Table 7 lists every training variant named in the paper. All CAT variants share the fault sampler of §2 (one top-3 QK tile, Gaussian replacement, $p_{\text{fault}} = 0.5$) and differ only in their loss terms and consistency target.

Table 7: **Training variants.** CE_c/CE_f: cross-entropy on the clean/faulted pass; KL: output-distribution Kullback–Leibler consistency with the clean prediction detached; $C_{\text{hid}}^{(\ell)}$: hidden-state consistency at layer ℓ (§2).

Variant	Loss terms	Role
Pretrained	none (not fine-tuned)	reference
Baseline	CE_c	unprotected fine-tuned reference
CAT-Output	CE_c+CE_f+λKL	main method (output target)
CAT-Stability	CE_c+λKL	ablation without corrupted CE
CAT-CE-Only	CE_c+CE_f ($\lambda = 0$)	fault-injection-only arm; corresponds to FAT [Zahid et al., 2020]
CAT-Hidden-Lk, $k \in \{3, 6, 9, 11\}$	CE_c+CE_f+$C_{\text{hid}}^{(k)}$, $\beta = 0$	placement sweep (post-fault hidden targets)
CAT-Only (L0–2)	CE_c+$C_{\text{hid}}^{(0:2)}$, $\beta = 0.1$	hidden-only diagnostic at the fault site
Fair controls (L0–2)	CE_c+CE_f+bounded C_{hid} or LN-MSE (the cosine β=0 control omits CE_f)	fault-site hidden target, matched loss terms
CAT-Hidden-JVP (L2)	CE_c+CE_f+$\|J_F \Delta h\|^2$	hidden loss projected to logits
R-Drop [Liang et al., 2021]	CE+output KL under dropout	consistency without faults
SAM [Foret et al., 2021]	CE under sharpness-aware updates	parameter-space flatness
SAF [Fuengfusin and Tamukoh, 2024]	CE with $W_{\text{eff}} = S \tanh(W/S)$	passive weight clamping
LLaMA-3.1-8B rows of Table 3 (same fault sampler on the 8B top-3 QK tiles)		
CAT-Hidden (LN)	CE_c+CE_f+0.1 MSE on the hidden state after the last decoder layer (layer 31)	late hidden target
CAT-Only	CE_c+0.1 MSE on the hidden state after layer 16	hidden-only diagnostic, mid-stack rather than at the fault site

Abbreviations: QK and OV are the attention GEMMs $QK^\top$ and $\text{softmax}(\cdot)V$; FC_1 and FC_2 are the two MLP GEMMs; LN-MSE is the mean-squared error between LayerNorm-normalized hidden states; JVP is a Jacobian–vector product; bf16 and fp32 are the bfloat16 and float32 formats; FSDP is fully sharded data parallelism; PTB is the Penn Treebank.

Table 8 separates the four fault channels that appear in the paper. Only the Gaussian channel is used in training; the other three are evaluation channels.

Table 8: **Fault channels.** What the model sees in each channel and the detector state it represents.

Channel	What reaches the model	Detector state
Raw bit flip	tile values with flipped bits: in-range, large but finite, or NaN/Inf (Fig. 2)	no detector
Gaussian tile	tile replaced by $\mathcal{N}(0, \sigma^2)$, σ the tensor standard deviation	stands for in-range residual error; training channel
Zero-fill tile	tile set to 0 (erasure)	detector flags the tile and erases it
Post-detection residual	missed tiles (recall $r < 1$), large-finite values left after a NaN/Inf clear, false-positive erasures, imperfect replacement	imperfect detector (App. E, V)

Protocol tags. Several CAT-Output numbers describe the same method under different horizons or harnesses and are not interchangeable. 0.21% is the full-horizon two-seed headline (603k and 930k steps for seeds 42 and 1337, n=100 evaluation sequences, Table 1, whose caption lists the horizon of every row); 0.84% is the same arm at the matched 80k-step horizon (Table 2); 0.92% is

a second 80k-step CAT-Output run (same recipe, seed 42) scored on the frontier harness with three evaluation seeds (App. H); and the mismatch and multi-bit grids use a third harness with n=50 sequences (App. F, E), where the same Gaussian top-3 cell reads 14.31% (Baseline) versus 0.28%. The pipeline ratio $1.10\times$ at n=1024 uses the 603k checkpoint on the pipeline harness (App. U); on the frontier harness the same checkpoint gives $1.12\times$ and the 80k-step frontier run $1.14\times$.

D Fault-Model Provenance

Table 9 states, for each factor of the fault model, whether it is grounded in hardware studies, measured in this paper, our own construction, or a parameter that enters only through a measured rate. The three QK positions are a vulnerability-weighted worst case (Appendix J); no reference we know of establishes an SDC preference for QK GEMMs.

Table 9: **Provenance of the fault model**, one row per fault-model factor. *Grounded*: supported by hardware studies; *measured*: tested in this paper; *constructed*: our modeling choice; *parametrized*: enters only through a measured rate.

Factor	Status in this paper	Evidence
Fault site, spatial structure	grounded: fixed physical site; fixity of the tile coordinate is our inference	Tung et al. [2026], Tyagi et al. [2026]
In-range error	measured: 72.5–89.8% of bit-flip outcomes; modeled by Gaussian tiles	Fig. 2; App. E
Large-but-finite error	measured: 8.5–26.0% of outcomes; tested in the calibrated and the value-clipped bit-flip channels	App. E, V
Magnitude channel	constructed: Gaussian (in-range), zero-fill (erasure)	mismatch grid, App. F
Position (top-3 QK)	constructed: vulnerability-weighted worst case; a deployment would obtain the prior by profiling the affected device	App. J
Node health, workload	parametrized: share of affected devices P	Ma et al. [2025], Pei et al. [2026]; App. I
Kernel, execution stage	parametrized: kernel maps to the position sweep; training-time SDC out of scope	Altenbernd et al. [2026]
Detector behavior	measured: recall, false positives, replacement	App. V
Precision format	constructed: bf16 bit patterns (Fig. 2); GPT-2 runs in fp32, and the bit-flip channels flip only bits that a bf16 value also carries	Tyagi et al. [2026]

Why the corruption is silent. Error-correcting codes protect data in storage and transport, not the arithmetic that produces it [Hochschild et al., 2021, Dixit et al., 2021, George et al., 2025]. A fault in the multiply–accumulate datapath therefore returns a valid floating-point value that is simply wrong, with no syndrome and no exception. Table 10 gives a coarse summary; only its last row matters for our argument, and protection of on-chip storage varies by vendor and generation.

Table 10: **Where error-correcting codes apply** (coarse; storage rows vary by vendor and generation and are listed only for contrast).

Component	Code-based protection	Outcome of a fault
Off-chip memory (HBM/DRAM)	ECC on datacenter parts	corrected, or detected as uncorrectable
On-chip storage (caches, register file)	ECC or parity, design-dependent	mostly corrected or detected
Arithmetic datapath (multiply–accumulate, reductions)	**none**	**valid but wrong value, no signal (SDC)**

Position fixity. A defective unit corrupts computation at a fixed physical site [Tyagi et al., 2026, Tung et al., 2026]; even intermittent faults are random in time rather than in space. That the corrupted *tile coordinate* is also fixed at inference, where kernel schedules are static, is our inference rather

than a finding of those studies. Under this reading, independent random placement understates the damage of a fixed defect: at the same tile count and channel, position-concentrated faults give PPL ratios of 8.5–196× on the pretrained and baseline models, where i.i.d. placement gives 1.1–2.1× (Table 21, Appendix Q). This motivates the training sampler's focus on fixed hotspots, which is also where the advantage concentrates. Transient soft errors are closer to i.i.d. placement, where the CAT-Output margin is smaller (Appendix F).

What distinguishes fault tolerance from generic robustness. Three measurements separate the two framings. (i) *The detector is a second design variable.* On the same CAT-Output checkpoint and fault positions, detect-and-erase gives a 1.00–1.01× PPL ratio, while leaving the value-clamped bit flip in place gives 663–8,844× (the fine-tuned baseline reaches 607–7,242×; Appendix V). (ii) *Knowing which tile is corrupted matters beyond the energy removed.* If damage depended on perturbation magnitude alone, the error and erasure channels could differ by at most their energy ratio, which is ≈2× (1.91–1.98×) at the all-GEMM scope of Table 5; the measured critical-count ratios are 4×, 8×, and 8×. Only the size of the gap discriminates here, because any monotone account predicts that erasure is less harmful than error. (iii) *Placement decides whether the consistency term helps or hurts* (Table 2), and the same output-KL objective on a generic perturbation (R-Drop) remains 29× worse (Table 4).

E Multi-Bit Bit-Flip Outcomes and a Calibrated Bit-Flip Channel

Exact enumeration (E-a). Figure 2 enumerates all 16 single-bit flips of a bf16 word over 2×10^5 unit-normal values and classifies each corrupted value by relative error. We extend the same enumeration to all $\binom{16}{k}$ masks for $k = 1, \ldots, 16$ (1.31×10^{10} exact evaluations, no sampling) and weight the result by a flip-multiplicity mixture digitized from Tung et al. [2026] ($\mathbb{E}[k] = 3.68$) under a uniform bit-position prior. The uniform prior is conservative: a mantissa-favoring prior raises the in-range share. The $k = 1$ slice reproduces the single-bit numbers of Figure 2 exactly.

Table 11: **Outcome of a corrupted bf16 value** (value-level shares on the unit-normal grid of Fig. 2). The multi-bit column uses our digitization of the multiplicity distribution of Tung et al. [2026]. Value-level shares are not comparable to outcome-level SDC censuses, which use a different denominator.

Outcome class	Single-bit	Multi-bit ($\mathbb{E}[k]$=3.68)
In-range (relative error ≤ 2), silent	89.8%	72.5%
Large but finite (invisible to a NaN/Inf screen)	8.5%	26.0%
NaN/Inf (visible to a special-value screen)	1.7%	1.5%

The in-range class stays the majority under multi-bit flips. The NaN/Inf share is non-monotonic in k (1.71% at k=1, peak 2.27% at k=2, 0.20% at k=8), because a special value requires the exponent field to become exactly all ones, which more flips make less likely. Multi-bit corruption therefore lowers the coverage of a special-value screen while enlarging the large-but-finite class that only a range check or ABFT can see. Two other multiplicity mixtures digitized from the same source give 74.5% and 74.1% in-range, so the conclusion does not depend on the digitization. These shares are conditioned on the unit-normal value grid; on the real QK activations used below, the same recipe realizes a NaN/Inf rate 4–28× lower.

Calibrated bit-flip channel (E-b). We re-ran the mismatch-grid harness (Appendix F; same checkpoints and top-3 tiles, random tiles redrawn with three seeds, n=50 sequences) with the Gaussian fill replaced by this multi-bit channel (realized mean k 3.67). In the *clamped* arm, a detector modeled as a NaN/Inf clear sets special values to zero and leaves the large-but-finite class (26% on the value grid) in place; the *unclamped* arm applies the identical faults with no detector.

- *Trained tiles.* Clamped, the baseline degrades 17.3% and CAT-Output 0.54%, a 32× ratio (per position 24–49×). This is a single-cell point estimate, of the same order as the Gaussian (51×) and zero-fill (15.8×) cells of the mismatch grid.

- *Unseen QK tiles.* CAT-Output degrades less in 10 of the 11 cells where a fault lands and ties in the remaining cell, whose tile is causally masked; with three seeds and a large per-seed spread we claim the direction only.
- *MLP tiles.* Most cells contain non-finite losses from downstream fp32 overflow in both models; the comparison is censored and supports no claim.
- *Unclamped.* The same faults contaminate 23 of 25 cells in both models, with mean sequence-level NaN fractions of 0.667 (Baseline) and 0.671 (CAT-Output). CAT gives no protection against special values; the detector is a prerequisite that CAT does not replace.

F Train/Test Fault-Prior Mismatch

CAT-Output is trained on one prior: Gaussian replacement at the three top-3 QK tiles. Table 12 evaluates it on channels and positions it never saw, with Baseline and CAT-Output receiving identical tiles (five seeds for random position sets, n=50 sequences; channel is the only difference between the two halves of the table).

Table 12: **Mismatch grid.** Fault degradation (%) and CAT-Output advantage (Baseline deg. / CAT-Output deg.). Density-1 cells of random sets are omitted (degradation $< 0.1\%$); —: no ratio, both degradations below 0.05%.

		Gaussian (trained channel)			Zero-fill (unseen channel)		
Positions	n	Base	CAT	Adv.	Base	CAT	Adv.
top-3 QK (trained)	1	14.31	0.28	51.1×	1.89	0.12	15.8×
random QK	4	0.70	0.46	1.5×	0.02	0.02	—
random QK	16	3.52	2.34	1.5×	0.20	0.14	1.4×
random QK	64	15.84	10.11	1.6×	0.79	0.60	1.3×
random QK	256	92.68	76.23	1.2×	4.81	10.28	0.5×
MLP	4	0.04	0.02	—	0.02	0.01	—
MLP	16	0.24	0.18	1.3×	0.06	0.05	1.2×
MLP	64	4.68	2.29	2.0×	0.29	0.19	1.5×
MLP	256	19.19	9.36	2.0×	0.90	0.59	1.5×
OV	4	0.32	0.14	2.4×	0.19	0.08	2.3×
OV	16	0.74	0.43	1.7×	0.38	0.19	2.0×
OV	64	3.30	2.12	1.6×	1.44	0.95	1.5×
OV	256	18.86	12.02	1.6×	8.23	5.85	1.4×

The advantage is large only at the trained prior and narrows to 1.2–2.4× at unseen positions (typically 1.5–2×), but it does not systematically invert: 25 of the 26 measured cells favor CAT-Output or sit at parity. The single exception, zero-fill at 256 random QK tiles, comes from one seed (CAT-Output PPL ratio 1.398 against 1.018–1.050 on the other four seeds, all of which favor CAT-Output); at this density far from the training prior the two models are effectively at parity. Where the grid overlaps an independent ten-seed sweep (Gaussian random QK), it agrees closely at n=16 (1.0352/1.0234 here versus 1.0354/1.0239) and within 1.5 percentage points at n=4 and n=64.

G Second-Seed Controls and Downstream Accuracy

Second seed for the fair hidden controls. Table 13 repeats the 80k-step fair controls of Table 2 with seed 1337 (data order and injection schedule re-drawn). The ordering reproduces; the catastrophic cosine-only arm without CE_f and the two arms with CE_f move by less than 0.5 percentage points. For CAT-Hidden-L11 the two values differ in horizon as well as seed (1.52% is the 40k-step seed-42 run of Table 1; 0.93% is an 80k-step seed-1337 run), and both stay above output KL at 80k (0.84%). The main GPT-2 comparison (Baseline, CAT-Output) is two-seed; the 8B comparison remains single-seed.

Downstream accuracy. On LAMBADA last-word accuracy (n=2000 examples, same top-3 QK Gaussian protocol, final checkpoints), CAT-Output is higher than Baseline both clean (20.25% vs. 19.60%) and corrupted (19.08% vs. 18.30%), for a retention of 94.2% versus 93.4%. The gaps (0.65 and 0.78 points) are below the standard error of a difference between two accuracies on 2,000 examples (≈1.3 points), and the per-position comparison is mixed: Baseline drops to 88.0% retention

Table 13: **Fair hidden controls at two seeds** (80k steps, top-3 QK protocol, fault degradation in %). †40k-step run; no mean is given across horizons.

Variant	Seed 42	Seed 1337	Mean
L0–2 cosine $\beta = 0$ (no CE$_f$)	22.95	22.49	22.72
L0–2 CE$_f$+cosine $\beta = 0$	1.74	1.71	1.72
L0–2 CE$_f$+LN-MSE	1.76	1.72	1.74
CAT-Hidden-L11	1.52†	0.93	—

at L1 but is higher than CAT-Output at L0 (19.20% vs. 19.05%), while CAT-Output retains 94–95% at all three positions, consistent with the layer-equalization effect of Appendix Q. We therefore read the accuracy result only as showing that CAT-Output does not lose downstream accuracy; perplexity remains the primary metric because it is more sensitive.

H Degradation–Cost Frontier

The main results use one operating point ($p_{\text{fault}} = 0.5$, $\lambda = 0.1$, ideal detector). To test whether that point is representative, we trained a horizon-matched grid at 80k steps (seed 42, three evaluation seeds) over fault severity, training exposure, and consistency weight. All CAT arms are 80k-matched; the Baseline reference is the 873k-step checkpoint.

Severity. Across $n = 1, \ldots, 1024$ corrupted tiles on both the erasure and the Gaussian error channel, CAT-Output has lower degradation than every hidden arm and FAT at every density with $n \geq 8$, and is the lowest of all arms except at erasure n=32, where the Baseline is lower (0.25% vs. 0.35%); below n=8, every arm except early hidden L0–2 stays under 0.05% (Figure 4). Hidden placements at L3–L11 collapse onto the fault-injection-only arm (FAT), so depth barely matters among post-fault hidden targets; only the output target separates. The margin over the best hidden arm is 1.10–1.32× on erasure and 1.19–1.55× on error for $n \geq 8$, growing with severity on the error channel. At n=1024, erasure degradation is 13.9% (CAT-Output), 16.6% (CAT-Hidden-L11), 16.8% (FAT), 48.7% (early hidden L0–2), and 18.3% (Baseline).

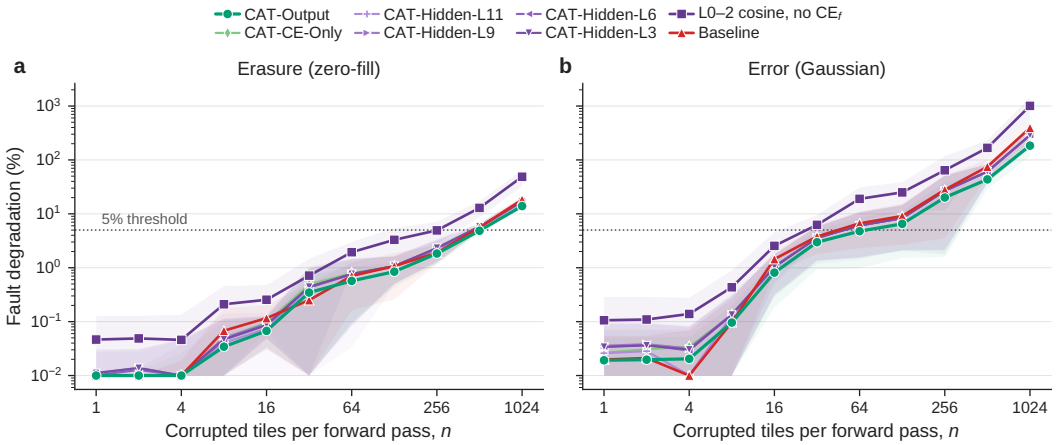


Figure 4: **Severity frontier.** Fault degradation versus corrupted tiles per forward pass on the erasure (left) and Gaussian error (right) channels, 80k-matched arms. For $n \geq 8$, CAT-Output is below every hidden arm and FAT on both channels.

Exposure. Table 14 varies the training injection rate on the paper's top-3 QK protocol. CAT-Output is the lowest arm at every rate. The hidden–output gaps at $p_{\text{fault}} = 0.1$ and 0.01 (0.71 and 1.18 percentage points) are 2.7× and 4.4× the run-to-run standard deviation of this metric (0.27); the gap at 0.5 (0.17 points) is within noise at seed 42 and widens at seed 1337 (0.43% vs. 1.10%). Under the single-evaluation protocol of Table 1, the CAT-Output checkpoints at $p_{\text{fault}} = 0.1$ and

0.01 read 1.44% and 3.11%, and the 80k checkpoint of Table 2 reads 0.84%; all stay below the 12.9% unprotected baseline.

Table 14: **Exposure frontier** (80k steps, top-3 QK protocol, three evaluation seeds, fault degradation in %).

p_{fault}	CAT-Output	CAT-Hidden-L11	FAT ($\lambda = 0$)	Hidden / output
0.5	**0.92**	1.09	1.11	1.19×
0.1	**1.46**	2.18	2.27	1.49×
0.01	**3.17**	4.35	4.49	1.37×

Consistency weight. At $p_{\text{fault}} = 0.5$, raising λ moves CAT-Output's erasure degradation at n=1024 from the FAT point (16.77% at $\lambda = 0$) to 13.94%, 13.48%, and 13.43% at $\lambda = 0.1$, 1, and 4, at a clean-PPL cost (19.01 → 21.84 on the sweep harness); CAT-Hidden-L11 stays flat in both degradation (16.62–16.68%) and clean PPL (19.01–19.07) (Figure 5). Most of the output gain arrives at the knee: λ from 0 to 0.1 captures 2.83 of the 3.34-point span. We then ran an 11-run grid over hidden depth (L3/L6/L9 at $\lambda \in \{1, 4\}$, plus a within-family $\lambda = 0$ anchor) and a seed-1337 replicate of the output and L11 arms, analyzed with rules fixed before the runs finished (Table 15). On erasure at n=1024 the output arm is responsive (span 6.2× the standard deviation of a λ difference); L11 and L9 are inert; L3 and L6 are inconclusive and, where they move, move in the harmful direction. Every hidden depth stays within 0.70 points (0.21× the output span). On clean PPL (cascade harness), the hidden span is at most 0.06 at every depth versus 2.33 for the output arm. Seed 1337 reproduces both behaviors (L11: 17.21% at both $\lambda = 0.1$ and 1; output: 13.48% → 12.95%).

Table 15: **Consistency-weight grid** (erasure at n=1024, 80k steps). Span: range of degradation over the swept λ; ETN: span divided by the standard deviation of a λ difference. Verdicts follow rules fixed before the runs finished; with two seeds we report effect size against measured spread rather than p-values.

Arm	Span (pp)	ETN	vs. output	Verdict
Output	3.34	6.24	1.00×	responsive
Hidden L11	0.19	0.35	0.06×	inert
Hidden L9	0.34	0.64	0.10×	inert
Hidden L6	0.56	1.04	0.17×	inconclusive
Hidden L3	0.70	1.31	0.21×	inconclusive

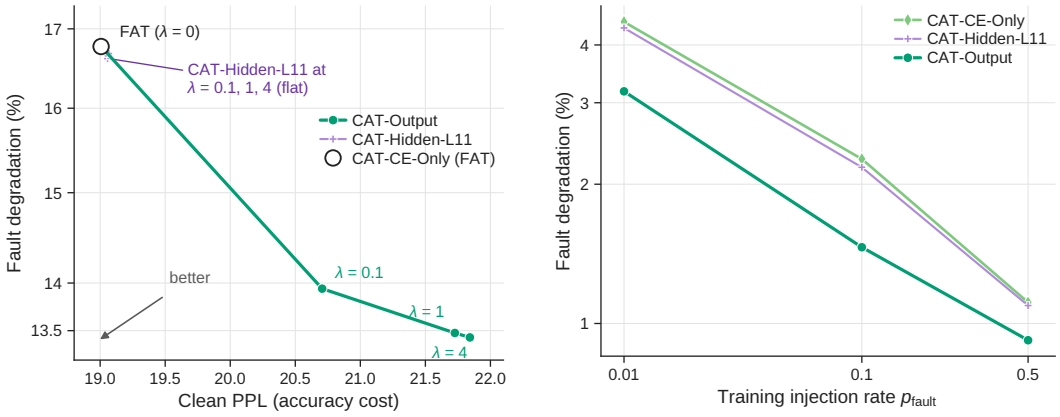


Figure 5: **Left: degradation–cost plane** (erasure at n=1024 versus clean PPL, sweeping λ at $p_{\text{fault}} = 0.5$). Output KL moves below the FAT point at a clean-PPL cost; hidden L11 is flat. **Right: exposure frontier** on the top-3 QK protocol (Table 14).

Caveats. The margin over the best hidden control is modest (1.1–1.55×); the large gaps are against early hidden alignment and no consistency training. CAT-Output trades clean PPL for robustness:

it extends the frontier to a low-degradation region no hidden arm reaches, rather than dominating on both axes. The arms' consistency terms are not scale-matched (about $180\times$ apart at equal λ, a loss-reduction convention), so inertness is a claim over the swept weight range. On the top-3 QK metric every λ effect, including the output arm's, lies within run-to-run noise (standard deviation 0.27 points; 0.38 on erasure), so the λ claims rest on the erasure and clean-PPL axes.

I Expected Benefit at Field Event Rates

Field SDC rates are too low to measure end to end, so we combine measured per-event damage with an event rate taken from fleet data. Let P be the share of affected devices ($P = 10^{-3}$ below, the CPU-fleet prevalence of §2) and r the detector recall. Each fault event is either detected and erased (probability r) or missed (probability $1 - r$). The expected per-token benefit of CAT is

$$B(P, r) = P \cdot \big[D_{\text{unprot}}(r) - D_{\text{CAT}}(r)\big], \qquad D(r) = r\, D_{\text{erase}} + (1 - r)\, D_{\text{miss}}, \tag{4}$$

where D is the per-event increase in per-token negative log-likelihood, $\Delta\text{NLL} = \ln(\text{ppl}_{\text{corrupt}}/\text{ppl}_{\text{clean}})$. Because passes are stateless, B is linear in P, so node health and workload (Appendix D) enter only through P. Table 16 evaluates the two terms that matter at $r \approx 1$. The fleet-weighted ratio is computed on ΔNLL, a log scale, and is therefore smaller than the per-event degradation ratio.

Table 16: **Per-event damage and fleet-weighted burden** (GPT-2). Fleet-weighted: $P \cdot \Delta\text{NLL}$ at $P = 10^{-3}$, in nats per token. The cascade row uses the two-seed headline; the erasure row uses three seeds at n=1024 tiles (1% of all tiles), where the unprotected reference is the pretrained model.

Event	Unprotected	CAT-Output	Per-event ratio	Fleet-weighted (P=10^{-3})
Cascade tile (top-3 QK)	12.9%	0.21%	60×	$1.2\times10^{-4} \to 2.1\times10^{-6}$ (≈57×)
Detected erasure, n=1024	86.8%	9.9%	8.8×	$6.2\times10^{-4} \to 9.4\times10^{-5}$ (≈6.6×)

Recall. The missed-fault term dominates once $r < 1$. In a uniform-miss stress test with 64 tiles spread over all GEMMs, where a missed tile carries value-clamped exponent bit flips, CAT-Output degrades 0.5% at $r = 1$ and 26.3% at $r = 0.99$, and the fine-tuned baseline tracks it (24.5% at $r = 0.99$); below $r \approx 0.9$ no training regime is usable (Figure 15). At the trained top-3 coordinates the same channel is flat in recall: 16 tiles drawn from those coordinates give $\approx 1.7\%$ for miss rates from 0.01 to 0.5 (three seeds). A tile-level ABFT detector evaluated on real GEMMs flags at least 99% of flips in the top five bf16 exponent bits and fewer in lower exponent bits [Gao et al., 2026], which suggests $r \approx 0.99$–1 for high-exponent flips. The benefit is therefore claimed for the detector-converted channel; the cost is $+1.11$ clean PPL and an expected 1.5 forward passes per training step, with no inference cost.

J Vulnerability Map

Figure 6 shows fault sensitivity across all layers and GEMM types for pretrained GPT-2. QK operations at early layers (0–2) are about $6\times$ more vulnerable than those at layers 6–11 on average (37.3 vs. 6.3 delta-perplexity), and up to $11\times$ from the most to the least vulnerable QK layer, with layer 1 QK as the critical hotspot (46.7). This motivated the top-3 QK training sampler.

K Replacement Strategy Hierarchy

Table 17 extends Table 5 to four replacement strategies spanning a replacement-quality spectrum: Gaussian (unbounded error), zero-fill (erasure), mean-fill (global side information), and low-rank-1 (per-layer side-information oracle). Performance improves monotonically with the amount of information the replacement carries about the original tile, from a $13.0\times$ PPL ratio under Gaussian to $1.03\times$ under low-rank-1 on the pretrained model (this sweep is separate from the pipeline sweep of Appendix U, which measures $13.75\times$ for the Gaussian cell and $1.87\times$ for the zero-fill cell). CAT-Output's multiplicative advantage is largest on the hardest Gaussian channel ($5.2\times$) and shrinks as replacement quality approaches the no-fault ceiling (zero-fill $1.9\times$, mean-fill $1.7\times$, low-rank-1 $1.03\times$).

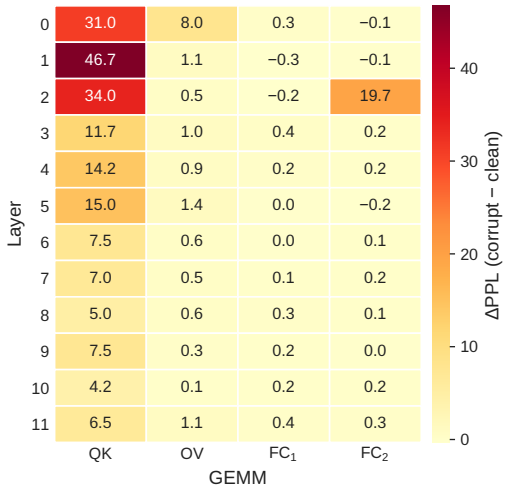


Figure 6: Fault vulnerability heatmap across layers and GEMM types for pretrained GPT-2. QK operations at early layers dominate.

Table 17: **Replacement strategy hierarchy at n=1024 tiles.** PPL ratio ($\text{ppl}_{\text{corrupt}}/\text{ppl}_{\text{clean}}$) under four replacement strategies; lower is better.

Strategy	Pretrained	CAT-Output
Gaussian	13.0	2.50
Zero-fill	2.09	1.11
Mean (global)	1.82	1.09
Low-rank-1	1.03	1.00

L Alignment-Target Placement Sweep

Figure 8 visualizes the placement sweep described in §3.1: fault degradation as the alignment target moves from early hidden layers (L0–2, fault-injection site) through L3, L6, L9, and L11 to the output. The sweep has two steps: a 34× drop from L0–2 to L3, where moving one layer past the injection site removes the hidden-only catastrophic regime, and a further drop from late hidden alignment to the output, which we attribute to the softmax shift invariance of §4.1.

M GPT-2 Corpus Robustness

Table 18 reports the held-out corpus check summarized in §3.1. The low-degradation ordering CAT-Output < CAT-Hidden-L11 < Baseline < CAT-Only reproduces on every corpus.

Table 18: **Corpus-robustness of fault degradation.** Relative PPL degradation (%) = $(\text{ppl}_{\text{corrupt}}/\text{ppl}_{\text{clean}} - 1) \times 100$, top-3 QK single-position Gaussian average; n=100 evaluation sequences for the WikiText-103 primary row and n=200 for the held-out rows. Negative values indicate fault-injected PPL slightly below clean.

Corpus (split)	CAT-Output	CAT-Hidden-L11	Baseline	CAT-Only
WikiText-103 (val, primary)	0.21	1.52	12.9	32.7
WikiText-2 (test)	−0.39	0.97	8.88	19.11
PTB (test)	−3.01	4.06	7.25	59.88
LAMBADA (test)	1.32	2.64	10.96	42.47
Mean, 3 held-out	**−0.69**	**2.56**	**9.03**	**40.49**

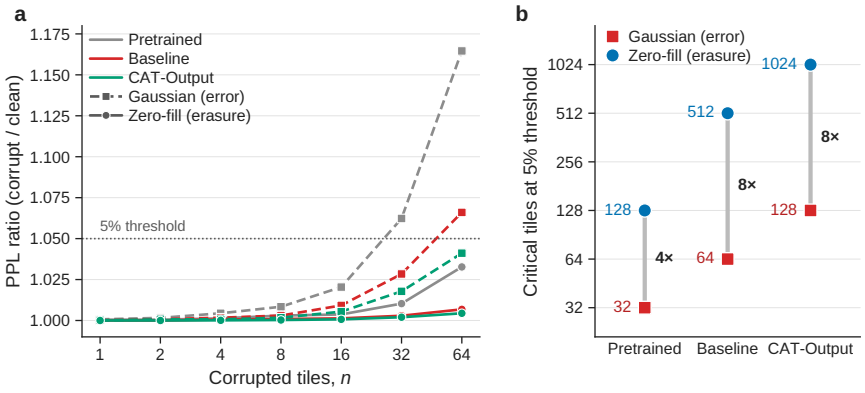


Figure 7: **Erasure vs. error channels.** (a) Degradation rate under Gaussian (dashed) vs. zero-fill (solid) corruption on the overlapping tile range. (b) Critical tile counts at 5% threshold (all-GEMM heterogeneous scope for both channels): erasure tolerates 4–8× more faults than error.

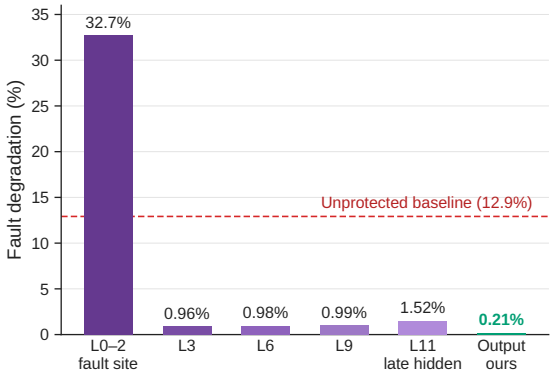


Figure 8: **Alignment-target placement sweep.** Fault degradation (%) as the consistency target moves from early hidden layers (L0–2, fault site) to the output, on a linear scale; same evaluations and training horizons as Figure 1 (two-seed for CAT-Output and Baseline). Dashed line: unprotected baseline (12.9%).

N Additional Mechanism Analysis

Proof details for Proposition 1. The second-order expansion follows from Taylor-expanding KL around $z = Wh$; the Hessian of KL with respect to the logits is F_p [Bishop, 1995]. We do not bound the remainder constant K analytically; the statement is local, and the check below finds the quadratic term within $1.05\times$ of the true KL at the measured perturbation scale. For any $\Delta h \in \mathcal{H}_\epsilon$, Eq. 3 gives $\mathrm{KL}(p\|p') \le \frac{1}{2}\sigma_{\max}(\widetilde{W})^2\epsilon^2 + K\epsilon^3$, which proves the forward inclusion under the stated threshold. For strictness, $\mathbf{1} \in \mathrm{range}(W)$ implies the existence of $\eta \neq 0$ with $W\eta = \mathbf{1}$. For any t, Lemma 1 gives $\mathrm{KL}(\mathrm{softmax}(Wh)\|\mathrm{softmax}(W(h+t\eta))) = 0$, while t can be chosen so $\|t\eta\| > \epsilon$; hence $\mathcal{O}_\delta$ contains points outside $\mathcal{H}_\epsilon$. With residual $\rho = \|(I - \Pi_{\mathrm{range}(W)})\mathbf{1}\|/\|\mathbf{1}\|$, the zero-cost axis becomes a low-cost axis: using $\eta = W^+\Pi_{\mathrm{range}(W)}\mathbf{1}$ leaves only the residual logit component, so the quadratic leakage is bounded by $\frac{1}{2}c^2\|F_p^{1/2}(I - \Pi_{\mathrm{range}(W)})\mathbf{1}\|^2$, i.e. $O(\rho^2)$ relative to the constant-shift scale. The LLaMA-3.1-8B residual is measured on the pretrained output head; the trained 8B robustness result is therefore empirical scale validation rather than a direct theorem invocation for that checkpoint.

KL bound verification. The Fisher-weighted second-order approximation of Proposition 1 achieves $1.05\times$ tightness for CAT-Output (ratio of true KL to the quadratic approximation), while a loose (non-Fisher) bound has a $60{,}000\times$ gap because p is peaked. The Fisher geometry is therefore essential for quantitative predictions.

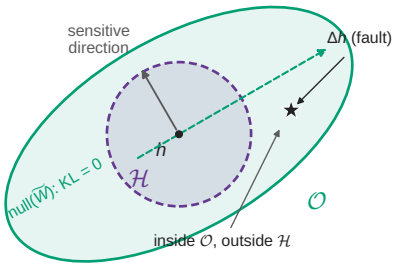


Figure 9: **Geometric intuition for Proposition 1:** for a sufficiently large KL threshold, the output constraint set $\mathcal{O}_\delta$ is an ellipsoid with a null or near-null direction along $\ker(\widetilde{W})$, strictly containing the hidden-state sphere $\mathcal{H}_\epsilon$. A perturbation inside $\mathcal{O}_\delta$ is tolerated at the output; the same perturbation would violate $\mathcal{H}_\epsilon$.

Unembedding range residual across model families. Table 19 reports $\rho = \|(I - \Pi_{\mathrm{range}(W)})\mathbf{1}\|/\|\mathbf{1}\|$ for the output heads used in our mechanism checks. GPT-2 is nearly exact after high-precision projection ($< 0.7\%$ residual on trained checkpoints). TinyLlama and LLaMA-3.1-8B do not satisfy the exact range condition, but their residuals are far below matched random-subspace references; the LLaMA-3.1-8B value is measured on the pretrained output head. For LLaMA-3.1-8B, changing accumulation precision and adding ridge 10^{-8} changes ρ by less than 1.4×10^{-5} absolute. A small residual does not by itself make the shift axis cheap, so the last column reports its cost under each model's own output distribution: the Fisher-weighted KL cost of a random hidden direction divided by that of the approximate shift direction, at equal norm (clean WikiText forward passes). On GPT-2 the axis is 500–64,000$\times$ cheaper than a generic direction, and at the measured GPT-2 perturbation scale its leakage is at most 0.6% of the observed operating KL; on LLaMA-3.1-8B it is $\sim 4\times$ cheaper; on TinyLlama it is $\sim 5\times$ costlier, so the geometric benefit is absent for that head.

Table 19: **Unembedding range residuals and null-axis cost.** Lower ρ means the all-ones logit-shift direction is closer to the unembedding range. Random reference is the residual of a random d-dimensional subspace in $\mathbb{R}^V$. Advantage: Fisher-metric cost of a generic hidden direction divided by that of the shift direction (> 1: the shift axis is cheaper).

Head	V	d	ρ (%)	Random ref. (%)	Advantage
GPT-2 pretrained	50,257	768	0.12	99.23	64,511$\times$
GPT-2 CAT-Output	50,257	768	0.64	99.23	513$\times$
GPT-2 Baseline	50,257	768	0.66	99.23	633$\times$
GPT-2 CAT-Only	50,257	768	0.67	99.23	935$\times$
TinyLlama pretrained	32,000	2,048	20.23	96.75	0.17$\times$
TinyLlama CAT-Output	32,000	2,048	19.23	96.75	0.23$\times$
TinyLlama Baseline	32,000	2,048	19.04	96.75	0.21$\times$
LLaMA-3.1-8B pretrained	128,256	4,096	10.72	98.39	4.2$\times$

TinyLlama gradient-alignment probe. Table 20 repeats the GPT-2 training-branch gradient-alignment diagnostic on TinyLlama-1.1B using the same corrupt cross-entropy plus output-KL branch. CAT-Output has positive task–consistency alignment on every tested layer and is consistently more aligned than the matched baseline probe. An output-KL-only diagnostic is weaker, so we use the training-branch measurement as the mechanism evidence matching the implemented objective.

Sigma robustness. CAT-Output maintains low degradation across a $100\times$ range of injection magnitudes (0.1σ to 10σ), indicating that the learned tolerance is not tuned to a specific noise level.

OOD fault positions. On fault positions outside the training sampler, CAT-Output's degradation rises from 0.16% (mean over in-distribution positions) to 4.09% on average (range 0.03–9.93%).

Table 20: **TinyLlama-1.1B task–consistency gradient alignment.** Values are layerwise cosine similarities over 16 minibatches; L0–L2 are the injected top-3 QK layers.

Variant	L0	L1	L2	L10	L21	Mean
CAT-Output	0.26	0.27	0.35	0.39	0.61	0.38
Baseline probe	-0.01	0.05	0.15	0.11	0.26	0.11

The original CAT-Only diagnostic shows catastrophic OOD degradation (up to 812% at some positions).

O SAF Baseline Details

Saturated Activation Functions (SAF) [Fuengfusin and Tamukoh, 2024] clamps weights with $W_{\text{eff}} = S\tanh(W/S)$ to limit activation magnitudes. We trained SAF variants with $S \in \{1.0, 2.0, 4.0\}$. Under the same top-3 QK single-position protocol as Table 4, the best setting is S=1.0 with 10.05% degradation (clean PPL 20.76, corrupt PPL 22.84), while S=2.0 and S=4.0 reach 10.38% and 11.68%. A separate random multi-tile sweep gives 15.7–16.1% degradation on zero-fill at n=1024 tiles; we report it only as a density-stress result, not as the main baseline comparison. SAF incurs higher clean perplexity cost than CAT-Output, consistent with the tanh constraint limiting model capacity.

P Hyperparameter and Fault-Density Sweeps

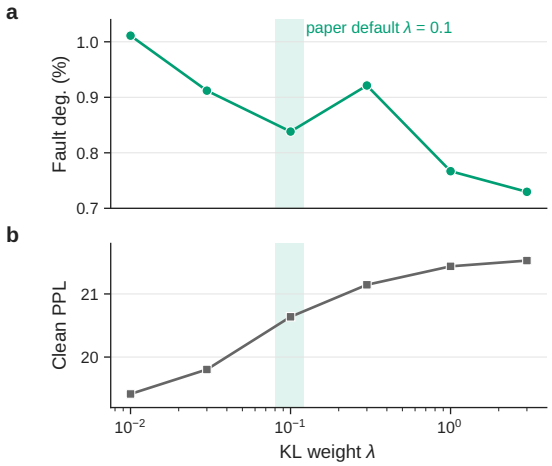


Figure 10: **KL-weight (λ) sensitivity.** (a) Fault degradation and (b) clean PPL at 80k training steps. The paper default is λ=0.1; degradation stays below or near 1% across a $300\times$ span.

Q Structured Fault Analysis

Position-concentrated faults (the same tile coordinates corrupted in every layer) are far worse than random placement at the same tile count and channel (Table 21): the pretrained PPL ratio is $18.2\times$ versus $1.26\times$ at $n = 64$ tiles and $195.8\times$ versus $2.13\times$ at $n = 256$. Appendix D explains why this is the relevant regime for a fixed defect.

CAT-Output reduces this penalty ($3.76\times$ at $n = 64$) and *equalizes layer vulnerability*: the bottom-3/top-3 layer ratio drops from $3.8\times$ (pretrained) to $1.0\times$ (CAT-Output), consistent with the lower curvature in the Hessian analysis (Appendix R).

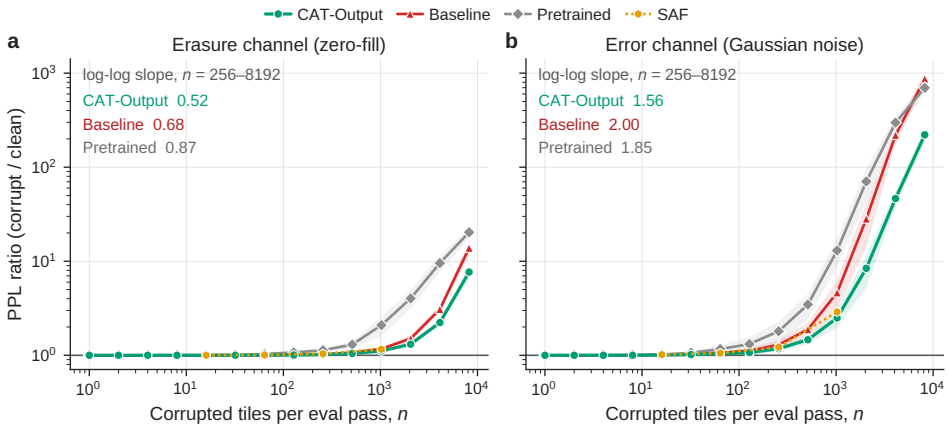


Figure 11: **Method comparison across fault densities.** PPL ratio as a function of corrupted tiles per evaluation pass on erasure (zero-fill) and error (Gaussian) channels. CAT-Output stays at or near the lowest PPL ratio across densities; ABFT detection converts the right-panel error channel into the left-panel erasure channel by zero-filling corrupted tiles.

Table 21: **Position-concentrated versus i.i.d. tile placement** (PPL ratio, Gaussian tiles, three seeds).

	$n = 64$		$n = 256$	
Model	i.i.d.	concentrated	i.i.d.	concentrated
Pretrained	1.26	18.21	2.13	195.8
Baseline	1.11	8.50	1.35	144.3
CAT-Output	1.07	**3.76**	1.20	**33.2**

R Hessian Spectra and Stability Check

The Hessian measurements report finite-budget power-iteration estimates from a single sweep with all four variants measured under identical settings (same random initialization, same iteration budget, same layer set). As a numerical-stability cross-check, we re-ran the CAT-Only measurement with a different power-iteration initialization; the re-run reproduces the qualitative signature (two large positive estimates at L0–L1 and a large-magnitude estimate at L2), but the numerical values differ (L0: 337, L1: 491, L2: 1300 vs. 161, 229, -645 in the main sweep). For a fixed deterministic symmetric Hessian, fully converged power iteration should select the largest-magnitude eigenpair; the discrepancy therefore indicates either finite-iteration/nonconvergence effects or non-identical stochastic minibatch Hessians. We therefore do not use the L2 sign as evidence of the curvature sign; the robust conclusion is that CAT-Only has unstable high-curvature behavior, consistent with the near-zero gradient alignment and trajectory spikes.

S The CAT-Hidden-L11 α Paradox

CAT-Hidden-L11 achieves $\alpha = 0.99$, far higher than CAT-Output's $\alpha = 0.48$, yet its fault degradation is higher (1.52% after 40k steps and 0.93% at 80k with seed 1337, versus 0.84% for CAT-Output at 80k and 0.21% at full horizon). These two values use different aggregations (0.99 is L11's value at layer 11, 0.48 is CAT-Output's 12-layer mean); on a common convention the comparison is 0.96 vs. 0.48 (12-layer means) or 0.99 vs. 0.55 (maxima), and the paradox is unchanged. We read this as consistent with treating α as a *necessary-but-not-sufficient* feasibility gate: the two objectives must not conflict ($\alpha \not\ll 0$) for training to reduce both, and among variants that pass this gate, the geometry of the constraint (Proposition 1) determines the achievable tolerance. Hidden-L11 alignment asks for conservation of the penultimate hidden state and so gives up the null-space tolerance that output alignment provides: a perturbation that shifts h_{11} along $\ker(\widetilde{W})$ incurs zero KL cost at the output under the exact range condition (a small cost under the measured residual of Table 19) but violates

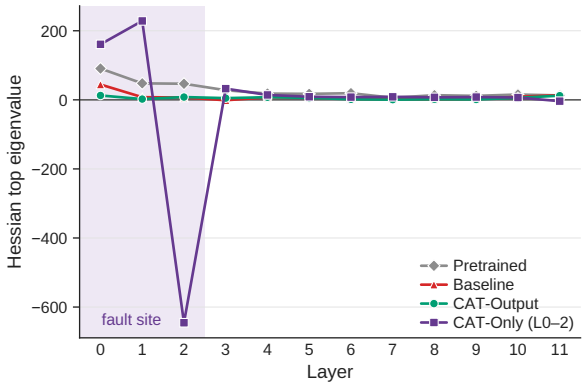


Figure 12: **Hessian power-iteration estimates per layer** (sign preserved via Rayleigh quotient). CAT-Output flattens the landscape ($0.39\times$ baseline on L0–L2); CAT-Only exhibits large, sign-sensitive curvature at L2, corroborating the high-curvature instability diagnostic in §4.2.

the hidden-state constraint. The consistency-weight grid (Appendix H) shows that at layer 11 this constraint has little effect in practice: across the swept weights the L11 arm stays at the $\lambda = 0$ (FAT) level, a span of 0.19 points in erasure degradation against 3.34 for the output arm (Table 15). The high α thus describes the direction of a consistency gradient that barely moves the model, and the L11 arm's tolerance comes mainly from the corrupted-pass cross-entropy it shares with CAT-CE-Only. We present this as a post-hoc reading of these measurements rather than a predictive claim.

T CAT-Only Training Dynamics

Section 3.1 uses CAT-Only's training dynamics to rule out undertraining. This appendix documents the event log behind that claim.

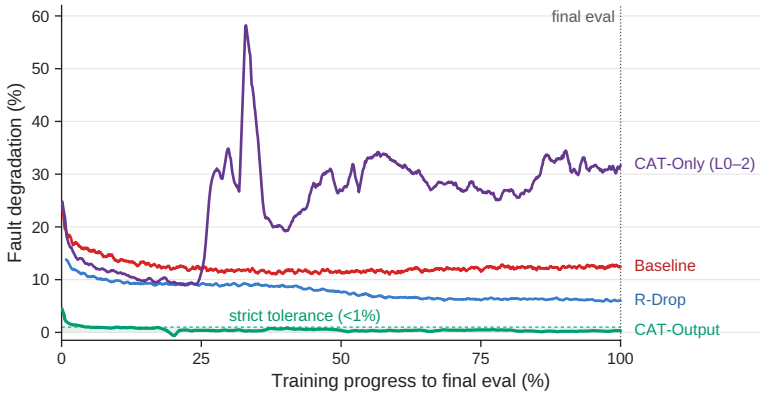


Figure 13: **Fault degradation during training.** Curves are lightly smoothed from eval logs and aligned so each run's final evaluated checkpoint lands at 100% progress (CAT-Output 603k, R-Drop 681k, Baseline 874k, CAT-Only 402k steps). CAT-Output enters the $< 1\%$ tolerance band early, while CAT-Only diverges above the unprotected baseline and R-Drop plateaus near $\sim 6\%$.

Timeline of the primary divergence (from the released training and evaluation logs; logged consistency values include the 0.1 loss weight):

- Steps 0–80k: CAT-Only converges from 35% fault degradation (step 0) to 9.2% mean / 11.2% max (steps 80–100k bin), briefly *below* the unprotected baseline of 12.9%.
- Step $103{,}324$: C=97.4 (first large outlier in the run; local max over preceding $5{,}000$ steps had been ~ 3.3).
- Step $104{,}988$: C=123.1, the peak trigger event.
- Step $105{,}485$: C=39.1 (residual oscillation).

- Steps 100–110k: fault degradation rises from 11.8% to 30% over 10k steps.
- Steps 131,500–133,500: fault degradation reaches 65.2% at step 131,500 and peaks at $\mathbf{65.9}\%$ at step 133,500 (spike visible in Figure 13); by step 131,500 $\text{ppl}_{\text{clean}}$ has worsened from 20.0 to 20.9, so the spike also degraded clean-task performance.
- Steps 140–200k: partial stabilization at 25–30%.
- Steps 197k–216k: a second cascade of outliers (C=335 at step 197,377; 5,229 at 206,589; 33,707 at 212,590) drives the $\sim$30% plateau seen through end of training.

Mechanism. Hidden-state alignment uses $C = (1 - \cos(\tilde{h}_\ell, h_\ell)) + \beta\,(\|\tilde{h}_\ell\| - \|h_\ell\|)^2$ on layers 0–2, with the clean hidden target detached. The cosine term is bounded in $[0, 2]$; the norm term is unbounded and dominates during rare fault patterns that produce anomalous magnitude excursions. With $\alpha_\ell \approx 0$ in the hidden-gradient diagnostic, the task loss gradient is near-orthogonal to this excursion direction and cannot act as a damping force. The cat_hidden_L11 variant disables the norm term (β=0 in its instantiation) for exactly this reason at L11 (where $\|h\|$ accumulates through 12 blocks), but CAT-Only operates at L0–2 where we initially retained it with β=0.1; the log above shows why even small-magnitude hidden states can produce large norm-loss spikes under structured fault patterns. The fair-control screen in Table 2 confirms that this norm spike is a real confound for the catastrophic CAT-Only number: adding corrupted CE and using bounded or normalized hidden losses reduces degradation to about 1.75% at 80k steps. At the same time, the cosine-only $\beta = 0$ row remains poor (22.95%), so the issue is not only the norm term; direct early hidden consistency also needs a strong fault-aware CE signal to train stably.

Counterfactual. CAT-Output applies KL at the logits, where the task gradient cooperates with the consistency gradient ($\alpha_\ell \in [0.36, 0.55]$). In the 603k-step CAT-Output run, the logged fault-pass loss (corrupted CE plus the weighted KL term) has a maximum of 34.7 over 301,848 fault-injected steps (mean 4.1), about $1000\times$ below CAT-Only's logged peak of 33,707, and no spike cascade appears.

U ABFT + CAT Pipeline Simulation

Section 3.4 argues that CAT and ABFT are complementary: ABFT detects corrupted tiles and zero-fills them, turning an unbounded bit-level error into a bounded erasure, which CAT-Output tolerates although its training uses only Gaussian tile replacement (Table 8). We verify this end-to-end in simulation; Figure 14 summarizes the result, and we provide the full protocol and per-mode curves below.

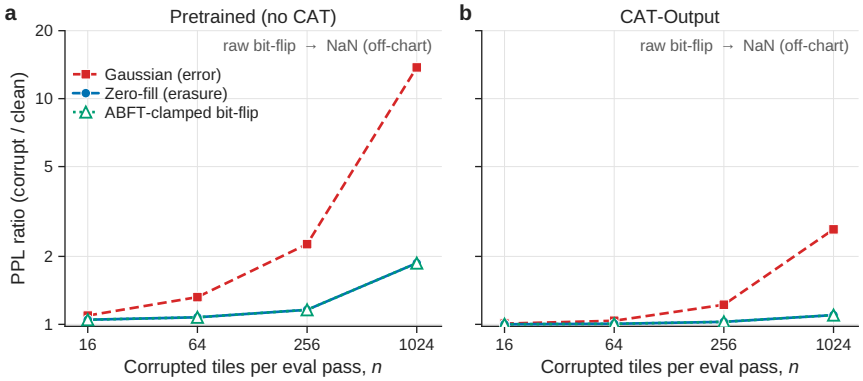


Figure 14: **ABFT+CAT detection–tolerance pipeline.** ABFT checksums detect corrupted GEMM tiles and replace them with zeros (erasure); CAT-Output absorbs the resulting erasures via its trained output-consistency objective. At n=1024 corrupted tiles this yields a $1.10\times$ PPL ratio versus NaN without ABFT. The ABFT-clamped curve (hollow triangles) coincides with zero-fill.

Protocol. For each $n \in \{16, 64, 256, 1024\}$ and three seeds, we inject four fault modes on the same model forward pass: (i) `gaussian`: tile replaced by $\mathcal{N}(0, \sigma^2)$, σ the GEMM-output standard deviation (error channel); (ii) `zerofill`: tile replaced by 0 (erasure channel); (iii) `bitflip_exp`: random bit-flips in the IEEE 754 exponent bits of the tile (the most destructive bit-level pattern); (iv) `bitflip_exp_clamped`: the bit-flip pattern followed by ABFT-style detection and whole-tile

zero-fill (implemented as a detected-tile overwrite, not PyTorch's default finite `nan_to_num` mapping).

Results. Raw `bitflip_exp` produces NaN in every run across all tile counts (not plotted; all PPL ratios undefined). `bitflip_exp_clamped` coincides with `zerofill` at the numerical precision of the evaluation because detected tiles are overwritten with 0; with perfect detection, an ABFT-style detector therefore converts the catastrophic error channel into the zero-fill erasure channel. On CAT-Output at n=1024, the ABFT+CAT pipeline yields a $1.10\times$ PPL ratio, compared to $2.64\times$ under the Gaussian error channel on the same model and $13.75\times$ on pretrained under Gaussian. The two stages contribute separately: ABFT converts the bit-level fault into an erasure (from NaN to a finite PPL ratio), and CAT lowers the erasure degradation from 86.8% (pretrained) to 9.9% (CAT-Output; three-seed means).

Figure 14 plots the four fault modes; numerical values at all tile counts and seeds are in the released data (Appendix AA).

V ABFT Design Considerations

Section 3.4 argues that CAT and ABFT compose into an end-to-end detection–tolerance pipeline, and Appendix U verifies the pipeline in simulation. Here we address four design questions that the main text defers: whether a hardware NaN-clamp alone is sufficient, how the pipeline degrades under partial detection recall, how false positives and imperfect replacement affect it, and whether per-tile recomputation would be an equally good substitute.

Saturation clamp without replacement. A weaker alternative to ABFT zero-fill is a hardware saturation clamp that prevents NaN/Inf but lets large finite values through (*bitflip-clip*). We test this on CAT-Output and the pretrained model with n=64 randomly placed tiles, as in the recall sweep below, and the bit-flip result clipped to the FP16 finite range ($\pm 6.55{\cdot}10^4$). CAT-Output's PPL ratio is 663–8,844$\times$ (three seeds), versus 1.00–1.01$\times$ under the erasure channel on the identical checkpoint; the fine-tuned baseline reaches 607–7,242$\times$ and the pretrained model $1.3{\cdot}10^4$–$3.7{\cdot}10^5\times$ under the same clip channel. CAT-Output stays 20–41$\times$ below pretrained in this degraded channel, but *all* models are unusable: preventing NaN alone is not enough, and CAT's tolerance covers bounded replacements (Gaussian, zero-fill) but not large finite values. Converting raw SDC into a regime CAT absorbs therefore requires an ABFT-style *replacement* of the detected tile; a NaN-safety clamp is not enough.

ABFT recall sensitivity. Detection is probabilistic: real ABFT schemes achieve $r = 1 - p_{\text{miss}} < 1$ recall, and a missed bit-flip falls through to the model. We sweep $p_{\text{miss}} \in \{0.01, 0.05, 0.1, 0.3, 0.5, 1.0\}$ at n=64 random tiles with a pipeline simulating both an ABFT detector (caught tiles zero-filled) and a downstream NaN safety net ($\pm 6.55{\cdot}10^4$ clamp on missed bit-flips); Figure 15 plots the resulting PPL ratio across CAT-Output, the unprotected baseline, and a never-fine-tuned pretrained GPT-2. We make three observations. First, the curve climbs steeply regardless of method: at 99% recall PPL is inflated by 1.2–1.5$\times$, and below 90% recall end-to-end quality is unusable for every row. Second, at this spread-fault density CAT-Output and the WikiText-103-fine-tuned baseline track each other ($1.26\times$ vs. $1.24\times$ at 99% recall; within $1.4\times$ of each other at every recall), and the gap to the pretrained model grows from $\approx 1.1\times$ at full recall to $36\times$ with no detector, so fine-tuning itself, rather than the CAT objective, absorbs most of the spread erasure damage here. Third, the elbow around $p_{\text{miss}} \approx 0.05$ is the practical design target for an ABFT + CAT deployment: above it, every training regime degrades quickly. The CAT-specific advantage that Table 1 reports is concentrated at the top-3 QK positions where a single undetected tile can cascade, not evenly distributed over random positions as here: with 16 tiles drawn from the trained coordinates, CAT-Output stays at $\approx 1.7\%$ degradation for miss rates from 0.01 to 0.5.

False positives and imperfect replacement. Table 22 extends the recall sweep with the two remaining detector non-idealities at a fixed miss rate $p_{\text{miss}} = 0.05$ (n=64 tiles, three seeds, n=200 evaluation sequences; medians, because the missed-bit-flip channel is heavy-tailed). A false positive erases a healthy tile, drawn from all 105,984 GEMM output tiles; imperfect replacement adds $\mathcal{N}(0, (0.3\sigma)^2)$ residual noise to every detected tile. At the trained top-3 positions CAT-Output stays ahead under every imperfection; on randomly placed tiles both models degrade similarly and both

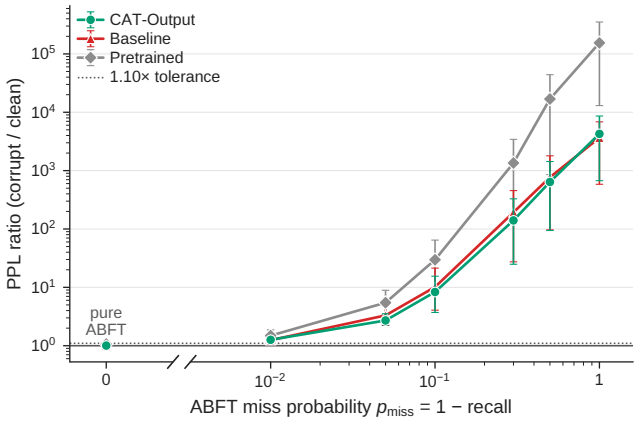


Figure 15: **ABFT recall sensitivity.** PPL ratio under a partial-detection pipeline: each corrupted tile is zero-filled with probability $1 - p_{\text{miss}}$ and left as a raw exponent bit-flip (clipped to FP16 finite range) with probability p_{miss}. p_{miss}=0 anchors at the pure-erasure regime (perfect detection); p_{miss}=1 is the worst case with no detector at all. Whiskers give the range over three seeds at n=64 tiles; p_{miss}=0 is drawn left of the axis break. CAT-Output (green) and the WT-103 baseline (red) track each other at this spread fault density; the never-fine-tuned pretrained GPT-2 (grey) is similar at full recall and up to 36× worse as recall falls.

axes stay second-order to recall. Replacement latency does not enter these numbers: the model only sees the zero-filled tile, so it is a detector-side throughput parameter that we did not measure.

Table 22: **ABFT operating region** (median PPL ratio, n=64 tiles, $p_{\text{miss}} = 0.05$; final checkpoints, 200-sequence harness).

	Top-3 QK positions		Random positions	
Condition	CAT-Output	Baseline	CAT-Output	Baseline
Operating point ($p_{\text{fp}} = 0$)	**1.017**	1.204	2.70	2.90
False positives, $p_{\text{fp}} = 10^{-3}$	**1.020**	1.204	2.72	2.92
False positives, $p_{\text{fp}} = 10^{-2}$	**1.069**	1.286	2.84	3.12
Imperfect replacement (0.3σ)	**1.023**	1.502	2.77	2.87

Recomputation as an alternative. An obvious alternative to tolerance is *recomputation*: if ABFT flags a tile as corrupted, rerun the affected GEMM block. We argue that recomputation is not an equally good substitute, for three reasons. *(i) Detection is probabilistic, not exact.* Practical ABFT schemes detect tile corruption with a finite false-negative rate; V-ABFT [Gao et al., 2026] and ReaLM [Xie et al., 2025] target LLM inference, while ApproxABFT [Xue et al., 2025] targets neural-network processing more broadly. Across these schemes, recall, checksum overhead, approximation thresholds, and selective recovery are explicit design tradeoffs rather than guaranteed exact correction for every corrupted tile. Undetected corrupted tiles reach the model as errors, not erasures, and recomputation cannot be triggered because the corruption was never flagged; CAT reduces the damage of such misses only at the coordinates it was trained on (see the recall sweep above). *(ii) Recomputation is not a free operation for autoregressive LLMs.* Each generated token reads every parameter through fault-prone GEMMs; per-tile recomputation serializes on the critical path of token generation, and prior ABFT work for LLM inference treats recovery cost as a central design constraint [Xie et al., 2025, Gao et al., 2026]. A pipeline that pays for tolerance *once* at training time (an expected 1.5 forward passes per step) avoids this per-token recovery cost. *(iii) The two mechanisms can be combined.* A system could recompute the tiles it flags with high confidence and zero-fill the rest; CAT raises the safety of the second bucket without forcing every tile into the first. The practical question is how large the cheap bucket can be while keeping end-to-end quality; CAT's tolerated erasure budget (up to n=1024 tiles at 1.10× PPL) measures it.

W LLaMA-8B Scale-Up Supplement

Section 3.2 reports the LLaMA-3.1-8B headline numbers; this appendix gives the per-corpus and per-position measurements behind its robustness summary. All 8B results are single-seed. Three evaluation harnesses appear, and absolute PPL is comparable only within one. (i) The 1024-token protocol re-scores the 50k-step checkpoints on the first n sequences of each corpus; it gives the 50k rows of Table 3 and all of Table 23, so CAT-Output's WikiText-103 clean PPL is 13.97 in both. (ii) The in-training evaluation (256-token context, 512 WikiText-103 validation sequences drawn with a fixed seed) gives the 5k rows of Table 3; it is the only harness that scored all four 5k variants. Logged at 50k steps, it gives CAT-Output 1.6% (clean PPL 14.76) and Baseline 24.7% (clean PPL 17.19), a 15.2× ratio, which is the like-for-like comparison with the 9.2× at 5k. (iii) The position-sweep harness (256-token context, 512 sequences drawn with a different seed) gives Table 24, on a separately exported checkpoint pair (clean PPL 16.12 for CAT-Output and 18.41 for Baseline). The CAT-Output $<$ Baseline ordering holds under each harness, with ratios of 12.8×, 15.2×, and 11.5× at the top-3 positions.

Corpus robustness (8B). Table 23 re-measures degradation on three held-out corpora (WikiText-2, PTB, LAMBADA) plus the in-distribution WikiText-103, using the same top-3 QK single-position Gaussian protocol as the main run. CAT-Output degrades less than Baseline on every corpus, with the ratio varying from 3.3× on LAMBADA (a long-form narrative task most sensitive to capability loss) to 12.8× on WikiText-103 and 12.6× on WikiText-2. PTB's absolute PPL is high across variants (560–760) because its vocabulary (⟨unk⟩, special tokens) mismatches LLaMA's BPE; the degradation *ratio* remains informative. The geometric mean advantage across the three held-out corpora is **6.0**×, and the CAT-Output $<$ Baseline ordering holds on every individual corpus.

Table 23: **Corpus robustness on LLaMA-3.1-8B (50k steps, 1024-token context).** Top-3 QK Gaussian positions, averaged. WikiText-103 and LAMBADA use n=512 samples; WikiText-2 and PTB use n=200.

Corpus (split)	CAT-Output		Baseline		Advantage
	$\text{ppl}_{\text{clean}}$	Deg. (%)	$\text{ppl}_{\text{clean}}$	Deg. (%)	(ratio)
WikiText-103 (val, primary)	13.97	**2.13**	16.13	27.20	12.8×
WikiText-2 (test)	12.72	**1.62**	14.64	20.38	12.6×
PTB (test)	560.52	**15.25**	758.27	79.33	5.2×
LAMBADA (test)	83.20	**6.46**	116.40	21.38	3.3×
Geo. mean, 3 held-out	—	**5.4**	—	**32.6**	**6.0**×

Position robustness (8B). The main result averages over the *same* top-3 QK tiles used during training, which leaves open whether CAT-Output only learns those positions. Table 24 evaluates both checkpoints on three position sets drawn from an exhaustive 512-position scan of the pretrained LLaMA-3.1-8B (the top-10 set contains the training top-3; the bottom-20 set is disjoint from both); the scan confirms that the training-time top-3 are the global top-3 by Δ_{ppl}. The baseline's degradation falls monotonically from the most to the least vulnerable set (25.4% → 12.8% → −0.03%), so vulnerability is concentrated at a few tiles and the bottom-20 are effectively silent. CAT-Output's degradation is nearly flat across the three sets (2.2%, 2.9%, 3.1%). The advantage ratio therefore depends strongly on which positions are evaluated: 11.5× at top-3 (the training positions), 4.4× at top-10 (the training top-3 plus seven other tiles), and a small reversal at the bottom-20 (3.1% vs. −0.03%), where CAT-Output leaves small fluctuations even though the baseline shows none.

Clean PPL under extended training. The scan's pretrained-model evaluation (clean PPL 16.95, same 256-token context as the in-training evaluation) serves as a reference. At 5k steps both variants are below it (11.11 for CAT-Output, 11.38 for Baseline). At 50k steps on the same harness CAT-Output's clean PPL rises to 14.76, still 13% below the reference, while the baseline rises to 17.19, 1% above it; under the 1024-token protocol CAT-Output also stays below the baseline (13.97 vs. 16.13). One reading is that the output-consistency term, which keeps the faulted-pass output close

Table 24: **Position robustness on LLaMA-3.1-8B (separately exported checkpoint pair, position-sweep harness).** Three position sets drawn from a 512-position scan over {layer}×{qk, ov, mlp_fc1, mlp_fc2}×{(0,0), (0,1), (1,0), (1,1)}; top-k selected by descending Δ_{ppl}, bottom-20 by the smallest. Degradation is averaged over each set, n=512 WikiText-103 samples.

Position set	CAT-Output (%)	Baseline (%)	Advantage	Notes
Top-3 (training positions)	**2.2**	25.4	11.5×	L0/L1/L2 QK tile (0,0)
Top-10 (incl. top-3)	**2.9**	12.8	4.4×	8 QK + 2 MLP-fc2
Bottom-20 (least vulnerable)	3.1	−0.03	—	Distributed across L1–L31

to the clean pass, also limits drift during fine-tuning; with a single seed and a small token batch (§3.2) we do not separate this from run-to-run variance.

In summary, the 8B advantage holds on held-out corpora and on the wider top-10 position set, with smaller ratios away from the training positions; all three checks are single-seed.

X Cross-Family Check: TinyLlama-1.1B

To test whether the output–hidden advantage is specific to the LLaMA-3.1 family, we re-ran the two-variant comparison (Baseline vs. CAT-Output) on TinyLlama-1.1B [Zhang et al., 2024], a distinct model trained by a separate team with a different pretraining recipe. Only CAT-Output is trained (no CAT-Only or CAT-Hidden-L11) because the main question is whether the *method* transfers. Fault protocol is identical to GPT-2 and LLaMA-3.1-8B (top-3 QK Gaussian single-position average).

Table 25: **TinyLlama-1.1B cross-family check (5k steps, top-3 QK single-position Gaussian average, n=200 per corpus).** The CAT-Output advantage holds on all four corpora and is of the same order of magnitude as on GPT-2 and LLaMA-3.1-8B.

Corpus (split)	CAT-Output		Baseline		Advantage
	$\text{ppl}_{\text{clean}}$	Deg. (%)	$\text{ppl}_{\text{clean}}$	Deg. (%)	(ratio)
WikiText-103 (val)	8.74	**3.17**	8.69	25.03	7.9×
WikiText-2 (test)	9.89	**1.72**	9.79	16.06	9.3×
PTB (test)	194.12	**14.77**	179.85	76.84	5.2×
LAMBADA (test)	32.33	**4.49**	31.82	18.75	4.2×

On WikiText-103 (val), CAT-Output degrades 3.17% versus Baseline's 25.03%: a 7.9× ratio, the same order of magnitude as the 9.2× on LLaMA-3.1-8B at the same 5k-step horizon. The ordering CAT-Output $<$ Baseline holds on every corpus, so the benefit of fault-aware output consistency is not specific to the LLaMA-3.1 pretraining recipe. This check compares CAT-Output with the baseline only; it does not test the output-over-hidden ordering, and the geometric account of §4.1 does not extend to this head (Table 19).

Y Qualitative Decode Samples

Perplexity numerics compress the visible consequence of fault injection into a single scalar. Table 26 shows the raw effect on generated text for four standard prompts under greedy decoding. Without detection or training-time fault awareness (row B), bit-flips corrupt the IEEE-754 exponent field; the resulting NaN logits are all-equal after softmax, and `argmax` collapses to token ID 0 ("`!`") at every step, producing the all-`!` continuations below. The CAT-Output model with ABFT-style detected-tile zero-fill (row C) produces readable text of similar quality to the clean model (row A) under the same per-position fault placement; phrasing artifacts such as "`@-@`" trace back to WikiText-103 tokenization. The four prompts are common short test queries, and row B fails identically on all of them.

Table 26: **Qualitative decode samples.** Four prompts, greedy decoding (GPT-2 Small, n=64 raw bit-flipped tiles, seed 2026). Unprotected inference (**B**) produces `NaN` logits and a degenerate all-"**!**" output because bit-flips in the IEEE-754 exponent propagate through softmax; CAT-Output paired with ABFT-style detected-tile zero-fill (**C**) produces readable text.

Prompt: *"The quick brown fox jumps over the lazy"*
(A) Clean unprotected: , lazy fox and they both fall to the ground. "I'm sorry, I'm sorry, I'm
(B) Unprotected + raw bit-flip: !!!!!!!!!!!!!!!!!!!!!!!!
(C) CAT-Output + ABFT zero-fill: fox and runs away . He then runs away again , but the fox jumps back into the forest . He then runs away

Prompt: *"Once upon a time, in a land far away, there"*
(A) Clean unprotected: was a man who had been a soldier in the army of the Emperor. He was a man of great courage and of
(B) Unprotected + raw bit-flip: !!!!!!!!!!!!!!!!!!!!!!!!
(C) CAT-Output + ABFT zero-fill: is a great deal of confusion and confusion . The first thing to be said is that the land is not a land of

Prompt: *"The capital of France is"*
(A) Clean unprotected: the capital of the French Republic, and the capital of the French Republic is the capital of the French Republic.
(B) Unprotected + raw bit-flip: !!!!!!!!!!!!!!!!!!!!!!!!
(C) CAT-Output + ABFT zero-fill: the city of Saint @-@ Domingue , which is located in the city of Saint @-@ Doming

Prompt: *"In artificial intelligence, a language model is a"*
(A) Clean unprotected: way to describe the world in a way that is more accurate than the human mind. The model is based on
(B) Unprotected + raw bit-flip: !!!!!!!!!!!!!!!!!!!!!!!!
(C) CAT-Output + ABFT zero-fill: model of how a human brain works . It is a model of how a human brain works in a way that is not

Z Related-Work Positioning Table

Table 27 separates inference-time detection/correction from training-time tolerance. Only FAT and CAT train on the tile-fault channel itself; CAT adds the output consistency term.

AA Code and Data

Code, training configurations, fault-injection protocols, seeds, per-run CSVs, figure generators, and the analysis scripts for every experiment, including those added during review and the consistency-weight analysis with its rules fixed in advance, are released at `https://github.com/TanHaonan/FTNet`. We also release, in the same repository, the GPT-2 checkpoints we retained, namely the final checkpoint of every run and the 80k-step checkpoints of the placement sweep, the fair controls, and Appendix H, together with the two TinyLlama-1.1B checkpoints of Appendix X. The 80k values of CAT-Output and CAT-CE-Only in Table 2 come from in-training evaluations whose checkpoints were not kept. The LLaMA-3.1-8B checkpoints are not part of the release; we release the 8B evaluation outputs behind Tables 3, 23, and 24, with the harness of each stated in Appendix W. The in-training values at 50k steps quoted there come from training logs that are not part of the release.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and §1 state quantified claims for CAT-Output versus the unprotected baseline (0.21% vs. 12.9%), the original hidden-only diagnostic (32.7%), the fair bounded hidden controls (1.74–1.76% at 80k), the local constraint-geometry result and measurement-backed mechanism evidence, generic robustness baselines (SAM, R-Drop,

Table 27: **Positioning of fault-tolerance methods for transformer inference.** *Fault-aware training* injects perturbations matched to the inference-time fault channel. *Tile-erasure tolerance* means PPL degradation under n=512 random zero-fill tiles remains $\leq$5% on GPT-2 Small (Baseline 6.89%, CAT-Output 4.68%); n/m: not measured; n/a: not a model property; ‡our CAT-CE-Only arm. Numeric entries are GPT-2 Small fault degradation under the matched top-3 QK single-position Gaussian protocol used in Table 4; the bit-flip pipeline of the last row is the simulated interface of Appendix U.

Method	Fault-aware training	Alignment target	Tile-erasure tolerance	Bit-flip pipeline	Deg. (%)↓
Unprotected baseline	✗	—	✗	✗	12.9
Inference-time detection/correction					
ABFT [Huang and Abraham, 1984]	✗	—	n/a	detect+correct	—
ReaLM [Xie et al., 2025]	✗	—	n/a	detect/recover	—
V-ABFT [Gao et al., 2026]	✗	—	n/a	detect/recover	—
ApproxABFT [Xue et al., 2025]	✗	—	n/a	approximate	—
Training-time robustness (not fault-specific)					
SAM [Foret et al., 2021]	✗	parameter	n/m	✗	∼8.8
R-Drop [Liang et al., 2021]	dropout	output	n/m	✗	6.3
SAF [Fuengfusin and Tamukoh, 2024]	✗	weight clamp	n/m	✗	10.05
Noisy Machines [Zhou et al., 2020]	analog	output	n/a	✗	—
Fault-aware training					
FAT [Zahid et al., 2020]‡	✓ (tile GEMM)	none	n/m	✗	1.01
CAT-Output (alone)	✓ (tile GEMM)	output	✓	✗	**0.21**
CAT-Output + ABFT	✓	output	✓	✓	**0.21**

SAF), and a simulated evaluation of a production-deployable detection-to-erasure interface yielding 1.10× PPL under bit-flip-to-erasure recovery; the abstract and §2 state which parts are simulated and which deployment mechanisms have production or real-hardware precedent. Each is evidenced in §3–3.4 with numbers matched to the reported protocols.

Guidelines:

- The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes]

Justification: §6 enumerates scope boundaries: the work targets a strong post-detection regime in which corrupt computation can be localized to GEMM tiles and bounded or converted to known erasures; raw bit-flip/NaN/Inf regimes require upstream detection; the sampler is vulnerability-weighted toward early QK hotspots; real-kernel ABFT integration and broader operator-balanced training remain future work; LLaMA evidence is 5k/50k rather than full pretraining; hidden-alignment variants are not exhaustive; CAT has an expected 1.5× training forward-pass cost; below detector recall $r \approx 0.9$ no tested training regime is usable; the interface is evaluated in simulation, not on faulty hardware, and the position prior is our construction (provenance in Appendix D); the 8B evidence is single-seed; and the theory is local, with a range condition only approximately satisfied (residual 0.6%, 20%, 11% for GPT-2, TinyLlama, LLaMA-3.1-8B) and a low-cost null direction

absent for the TinyLlama head. §3.4 states out-of-scope fault modes (non-GEMM, adversarial, training-time corruption).

Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Proposition 1 is the only formal proposition beyond Lemma 1; it is stated with all assumptions (softmax shift invariance, a KL threshold large enough relative to the hidden radius, and exact $\mathbf{1} \in \mathrm{range}(W)$ for strict zero-cost containment), with a proof sketch in the main text and proof details in App. N. The proposition is stated as a local result: the second-order KL expansion carries an $O(\|\Delta h\|^3)$ remainder whose constant we do not bound analytically, and its measured tightness is reported in App. N. The unembedding range condition is checked empirically across GPT-2, TinyLlama, and a pretrained LLaMA-3.1-8B head in Table 19; the trained 8B checkpoint is not claimed to satisfy the exact theorem condition, and the Fisher-metric cost of the null direction is reported per head, including the TinyLlama head where it is not cheaper than a generic direction. Gradient and Hessian sections are explicitly labelled mechanism diagnostics rather than theoretical results. No claim in the paper is labelled as "theorem".

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.

- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: §2 specifies the fault model and training objective; Algorithm 1 gives the training loop; Appendix A lists the main GPT-2 and LLaMA hyperparameters. Fault-injection probability, tile size, exact top-3 structural tile indices, and evaluation protocols are enumerated. Appendix AA gives the release of code, configurations, seeds, GPT-2 and TinyLlama checkpoints, and experiment CSVs; Appendix C tags the reported numbers with their training horizon and evaluation harness.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Code, training configurations, fault-injection protocols, seeds, per-run CSVs, figure generators, and the analysis scripts for all experiments, including those added during review, are released publicly (Appendix AA), together with the GPT-2 checkpoints used

for the main results and the TinyLlama-1.1B checkpoints. The datasets and base models used are publicly available or accessible under their provider terms; PTB may require LDC access.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: §2 specifies GPT-2 Small + WikiText-103, the top-3 single-position fault sampler, the objective, and the evaluation metric. Appendix A contains the main configuration for GPT-2 and LLaMA-3.1-8B runs, including the decoupling of training-time fault sampling from evaluation-time top-3 averages and density sweeps.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: CAT-Output and Baseline on GPT-2 are pooled across two seeds (42, 1337); Table 1 reports the two-seed mean with a rounded half-range as the $\pm$ value. SAF reports three clamping strengths, and ABFT recall sensitivity (Figure 15) uses three seeds. Hessian stability cross-check in Appendix R documents power-iteration seed variability. The fair hidden controls are repeated at a second seed (Appendix G), and the consistency-weight grid includes a seed replicate analyzed with rules fixed before the runs finished, reporting effect size against measured run-to-run spread rather than p-values (Appendix H). Remaining ablations and the LLaMA-3.1-8B comparison are single-seed and are identified as such.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix A states GPT-2 runs use a single NVIDIA A100 and LLaMA-3.1-8B runs use 7-GPU FSDP. GPT-2 training is 603k steps for primary CAT-Output (about 28 GPU-hours from logs), 873k for the 5-pass baseline (about 21 GPU-hours), and about 1–1.2 GPU-hours for each 80k fair-control screen. LLaMA runs are 5k/50k steps under the reported FSDP setup; from the training logs, a 5k-step 8B run takes 6.1–8.2 hours on 7 GPUs and a 5k-step TinyLlama run 1.1–1.9 hours on one GPU. CAT uses an expected 1.5 forward passes per step because the second pass is sampled with $p_{\text{fault}} = 0.5$. The degradation–cost frontier (Appendix H) adds 80k-step GPT-2 runs at about 1–1.2 GPU-hours each, including 11 runs for the consistency-weight grid; the remaining review-stage analyses are evaluation-only.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work uses standard language-modeling datasets (WikiText, PTB, LAMBADA) and model weights (GPT-2, LLaMA, TinyLlama) under their respective licenses or access terms, trains no models on private or human-subject data, and does not involve crowdsourcing or personally identifiable information.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.

- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The Broader impact paragraph of §6 discusses both sides: learned fault tolerance could lower the cost of reliable LLM inference by reducing hardware redundancy and recomputation, and the main risk is misplaced reliance, since a model that tolerates detected erasures is not protected against undetected special values or fault patterns far from its training channel; we recommend keeping detection in place and validating tolerance on the target hardware.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The released assets are code and fine-tuned GPT-2 and TinyLlama-1.1B checkpoints trained on WikiText-103; they add no capability beyond the underlying public base model and pose no high risk of misuse. CAT is a training-time regularization method applied to existing open models.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.

- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: GPT-2 [Radford et al., 2019], LLaMA-3.1-8B [Grattafiori et al., 2024], TinyLlama [Zhang et al., 2024], WikiText-103 [Merity et al., 2017], PTB [Marcus et al., 1993], LAMBADA [Paperno et al., 2016], OCP SDC guidance [George et al., 2025], and baselines SAM [Foret et al., 2021], R-Drop [Liang et al., 2021], SAF [Fuengfusin and Tamukoh, 2024] are all cited with original papers or primary technical reports. The main assets are used under their stated licenses or access terms: GPT-2 under OpenAI's Modified MIT/MIT terms, LLaMA-3.1-8B under the Llama 3.1 Community License, TinyLlama under Apache-2.0, WikiText under CC BY-SA 3.0/GFDL terms, LAMBADA under its public dataset terms (CC BY 4.0 for the standard release; Modified MIT for the OpenAI-preprocessed fallback used by some loaders), and PTB/Treebank-3 under the LDC User Agreement/provider terms. We do not redistribute third-party datasets or base-model weights.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The released code, checkpoints, and CSVs (Appendix AA) are documented in a README that lists contents, dependencies, commands for every figure and table, license, and the third-party assets downloaded by standard loaders. No new datasets are released.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper involves no human subjects and no crowdsourced data.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The paper involves no human subjects and does not require IRB review.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs are studied as the *subject* of fault-tolerance training (GPT-2, LLaMA-3.1-8B as base models), not as a component of the method development pipeline. No LLM was used as an original or non-standard methodological component.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.