ECHOING IS NOT ALL YOU NEED: SEMANTICALLY GUIDED REPAIR OF GLITCH TOKENS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) have demonstrated remarkable capabilities across tasks. However, they can fail to use certain tokens, known as *glitch tokens*, even in simple repetition tasks. Existing methods can restore token generation; however, this does not ensure semantic recovery. Our key insight is that models can retain semantic understanding of glitch tokens, yet naive repairs may corrupt this understanding. Therefore, we target *functional repair*, aiming to restore correct use of these tokens in context. To this end, we propose GLITCHPATCH, a semantically guided framework combining unified closed-form output correction with selective semantic learning. This training-free correction calibrates one output-head row per glitch token using a few contexts, leaving all other parameters unchanged. For tokens failing semantic assessment, we show that reliable repair requires prior semantic learning, achievable through simple reference-based training of a shared adapter. With repairs applied jointly across the vocabulary, GLITCHPATCH achieves functional repair rates of 86.5% on 16,788 Qwen-7B-Chat targets and 78.6% on 4,351 DeepSeek-LLM-7B-Chat targets, compared with 16.2% and 24.6%, respectively, for GlitchCleaner. Complete-repair evaluations additionally cover Llama-2-7b-chat-hf, Qwen3.5-9B, and Qwen3-32B. On Qwen-7B-Chat, accuracy changes remain negligible on MMLU, QQP, and GSM8K when numeric-answer extraction is used. Low repair cost and modest measured inference overhead support vocabulary-wide deployment.

1 INTRODUCTION

Large language models (LLMs) have achieved remarkable performance on diverse tasks, including question answering, text summarization, and code generation (Grattafiori et al., 2024; Yang et al., 2025). However, they can still fail on seemingly innocuous inputs containing specific vocabulary items. Certain items, known as *glitch tokens*, can trigger anomalous model behavior (Rumbelow & Watkins, 2023; Li et al., 2024) and even be exploited to bypass safety mechanisms (Tang et al., 2026). Even in a simple repetition task, a single glitch token can cause garbled output or unrelated substitutions, undermining the model's reliability in routine use (Fan et al., 2026; Tang et al., 2026).

Prior work has developed several strategies for mitigating these failures, including splitting problematic tokens into smaller units (cafeii, 2026), adjusting internal activations (Zhang et al., 2024), training lightweight adapters (Fan et al., 2026), and editing only input embeddings (Tang et al., 2026), effectively improving token reproduction and reducing anomalous behavior. However, successful token reproduction does not guarantee recovery of the token's semantic function. For example, a repaired model may correctly repeat a technical identifier yet misinterpret its semantics. Therefore, we target *functional repair*, which seeks to restore the model's ability to use glitch tokens appropriately in context, rather than merely make them reproducible. Figure 1 illustrates the recovery of both capabilities on a Qwen3-32B target.

Our key insight is that models can retain semantic understanding of some glitch tokens, so repair should preserve this understanding and apply semantic learning only when needed. We examine semantic behavior in Qwen-7B-Chat (Bai et al., 2023) before and after repair and find that 6,593 of the 16,788 screened glitch tokens already pass a reference-based semantic assessment before repair, demonstrating that semantic understanding can persist despite generation failure. We also find that some of these tokens fail the same assessment after existing repair methods are applied, showing

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

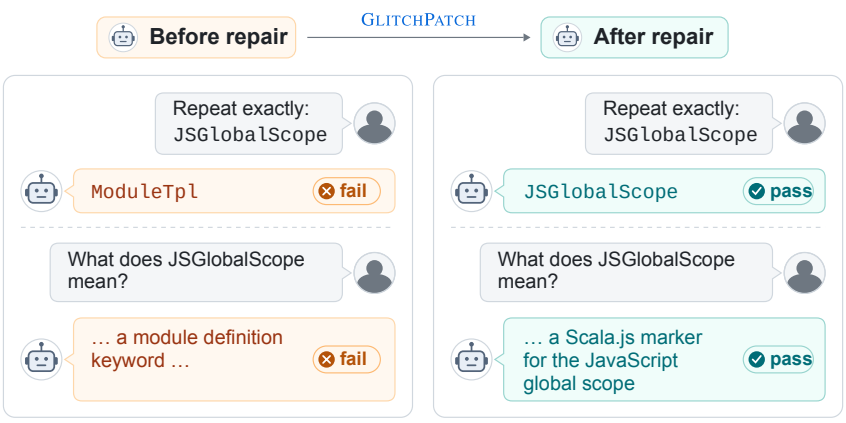


Figure 1: GLITCHPATCH restores both generation and interpretation of a glitch token.

that additional learning can corrupt the model's existing understanding. Additionally, tokens that fail the initial semantic assessment continue to fail after repair with existing methods. These findings motivate a repair strategy that first classifies glitch tokens by their pre-repair semantic accessibility.

Building on these findings, we introduce GLITCHPATCH, a general method for glitch-token repair through closed-form output-head correction (Figure 2). For each target glitch token, the method fits a single output-head row to a small set of calibration contexts while keeping all other parameters fixed. This fitting problem admits a closed-form solution, eliminating the need for gradient-based optimization during output correction. For tokens with accessible semantics, this correction achieves effective functional repair without further processing. However, for tokens that fail semantic assessment, output correction alone is insufficient for reliable functional repair. Therefore, we first classify tokens by their semantic accessibility and provide additional semantic training only for those that fail the assessment. We find that simple reference-based training of a shared LoRA adapter (Hu et al., 2022) can recover the model's semantic understanding of these tokens. After this low-cost training, we apply the same output-head correction with the adapter active. Importantly, semantic training alone is also insufficient for reliable functional repair of these tokens and must be combined with output-head correction, as demonstrated by our component experiments in section 4.4.

To assess the effectiveness and practicality of GLITCHPATCH, we conduct vocabulary-scale evaluations on five models from the Qwen (Bai et al., 2023; Yang et al., 2025), DeepSeek (DeepSeek-AI, 2024), and Llama (Touvron et al., 2023b) families, with all corrected output-head rows installed simultaneously in each model. On Qwen-7B-Chat and DeepSeek-LLM-7B-Chat, GLITCHPATCH achieves *functional repair* rates of 86.5% on 16,788 and 78.6% on 4,351 glitch tokens, respectively, compared with only 16.2% and 24.6% for a state-of-the-art repair method (Fan et al., 2026) under the same evaluation protocol. Further evaluations yield functional repair rates of 75.7%, 94.8%, and 69.3% on Llama-2-7b-chat-hf, Qwen3.5-9B, and Qwen3-32B, respectively. These gains come with negligible impact on Qwen-7B-Chat's accuracy on MMLU (Hendrycks et al., 2021), QQP (Wang et al., 2018), and GSM8K (Cobbe et al., 2021). For repair cost, the shared semantic adapter contains less than 0.02% of the base model's parameter count and requires only a few GPU-hours of training. For inference efficiency, the repaired model maintains baseline throughput with the semantic adapter merged into the model weights, while the unmerged implementation reduces throughput by less than 3.0% in fixed-length generation tests.

To summarize, our contributions are as follows:

1. We identify the gap between token reproduction and semantic recovery, motivating functional repair that restores both output generation and semantic correctness.
2. We introduce GLITCHPATCH, a unified repair framework that combines closed-form output-head correction with semantic assessment and selective semantic learning.
3. We conduct extensive vocabulary-scale experiments across five models, demonstrating the effectiveness and practicality of GLITCHPATCH through high functional repair rates, low repair cost, and negligible impact on benchmark accuracy and inference efficiency.

2 Related Work

Tokenization and Vocabulary Health. Subword tokenization affects robustness, multilingual coverage, and token cost (Sennrich et al., 2016; Kudo & Richardson, 2018; Bostrom & Durrett, 2020; Rust et al., 2021; Ahia et al., 2023). Vocabulary construction and training address these issues during model development (Section H). GlitchPatch repairs token failures after training.

Glitch-Token Detection and Repair. Prior work documents anomalous tokens (Rumbelow & Watkins, 2023), classifies their symptoms (Li et al., 2024), characterizes target-under-trained tokens (Land & Bartolo, 2024), and audits polluted vocabulary entries (Zhang et al., 2025). Detection methods use embedding norms (Land & Bartolo, 2024), entropy maximization (Wu et al., 2026), or activation probes (Zhang et al., 2024). Repair is commonly evaluated through target-string reproduction or task-specific behavioral checks. GlitchPatch measures joint output and semantic recovery and uses pre-repair semantic assessment to select which tokens receive additional training.

Local Editing and Output Heads. Output embeddings govern generation and interact with training dynamics (Press & Wolf, 2017; Inan et al., 2017; Yang et al., 2018), while tied embedding/unembedding interfaces can drift during training (Gu et al., 2026). Knowledge-editing methods localize factual updates to MLP or attention layers (De Cao et al., 2021; Meng et al., 2022; Mitchell et al., 2022; Meng et al., 2023; Yao et al., 2023). Parameter-efficient methods restrict training through adapters, prefixes, prompts, or low-rank updates (Houlsby et al., 2019; Li & Liang, 2021; Lester et al., 2021; Hu et al., 2022; Dettmers et al., 2023). GlitchEdit applies a training-free edit to the affected input embedding (Tang et al., 2026). GlitchPatch directly recalibrates the output row in closed form, adding a shared semantic adapter for tokens that fail the pre-repair semantic assessment.

3 Method

GlitchPatch uses pre-repair semantic assessment to select which targets need semantic learning before the shared closed-form row update (Figure 2). We classify glitch tokens as *semantic-preserved* (SP) or *semantic-deficient* (SD) based on whether they pass the pre-repair semantic assessment.

3.1 Problem Setup and Repair Criteria

Let M have vocabulary V, hidden dimension d, output head $\mathbf{W}_{\mathrm{lm}} \in \mathbb{R}^{|V|\times d}$, and decoder Dec. Token t has surface $T_t = \mathrm{Dec}([t])$. The output head is parameterized separately from the input embeddings, so its rows can be edited independently (Press & Wolf, 2017; Inan et al., 2017).

Let $\mathcal{Q}_{\mathrm{screen}} = \{q_0, q_1\}$ be the screening prompts and $Y(M', q, t)$ the response under fixed decoding. Let $\bar{T}_t = \mathrm{lstrip}(T_t)$, where lstrip removes leading whitespace and preserves trailing whitespace. Output success is

$$o(M', t; q) = \mathbf{1}\left[\bar{T}_t \neq \epsilon \ \wedge \ \bar{T}_t \sqsubseteq Y(M', q, t)\right], \tag{1}$$

where ϵ denotes the empty string and $\sqsubseteq$ denotes exact string occurrence. Whitespace-only targets are excluded.

Definition 1 (Glitch Token)**.** *Token t is a glitch token if $o(M, t; q) = 0$ for both screening prompts.*

The repair population G contains screened glitch tokens with a frozen reference r_t for their meaning, identity, construction, or function. The indicator $S(M', t; r_t)$ records whether model M' passes the associated semantic assessment.

Definition 2 (Functional Repair)**.** *For a target $t \in G$, let $M^\star$ denote the repaired system. Functional repair requires output success and semantic success for the same token:*

$$F(M^\star, t) = o(M^\star, t; q_1)\, S(M^\star, t; r_t) = 1. \tag{2}$$

Output recovery without semantic agreement is a surface-only repair.

3.2 Semantic Diagnosis and Routing

We ask each target's open-ended semantic question once and assess the answer against its reference using two judgments under a fixed rubric. The answer passes if either judgment accepts it.

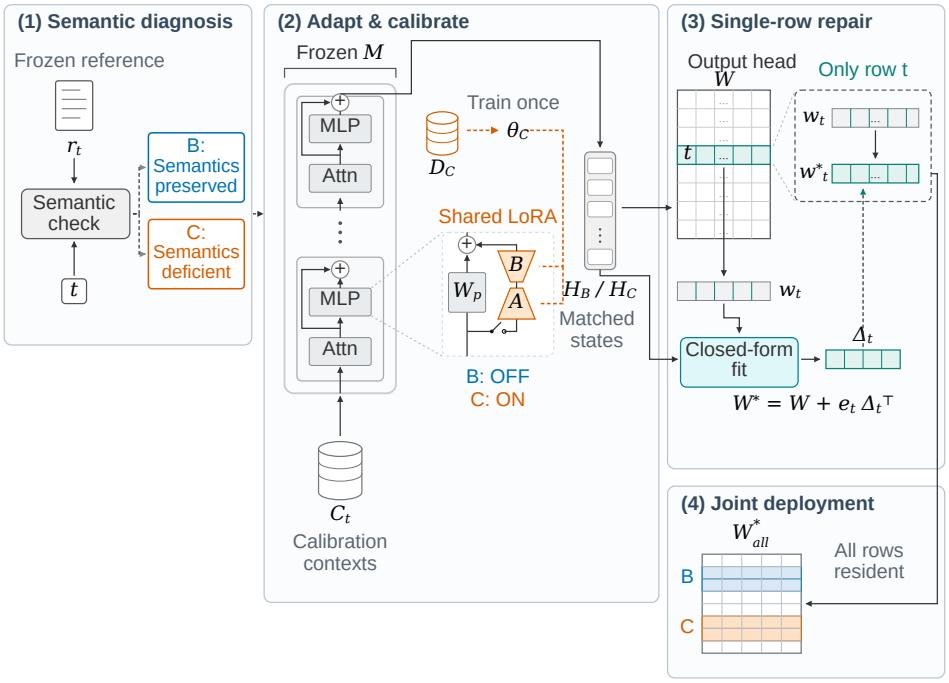


Figure 2: GLITCHPATCH: a general output-row repair with conditional semantic training. Closed-form row correction restores output access. Reference-based assessment identifies targets requiring semantic training (SD) before correction; semantic-pass targets (SP) receive the correction directly. All repaired rows remain installed; the adapter is active for semantic-deficient requests.

Definition 3 (Semantic Accessibility and Token Type)**.** *Semantic accessibility is the reference-based indicator* $A_{\text{sem}}(M', t) = S(M', t; r_t)$.

The pre-repair assessment assigns $t \in G$ *to SP when* $A_{\text{sem}}(M, t) = 1$ *and SD when* $A_{\text{sem}}(M, t) = 0$. *We denote these frozen subsets by* G_B *and* G_C, *with* $G = G_B \cup G_C$.

Questions, references, rubrics, and route assignments are frozen before repair. The same assessment measures semantic retention for G_B and recovery for G_C (Section A).

Isolated row repair recovers output for over 99% of both Qwen groups, with functional success of 87.7% for SP and 15.0% for SD (Figure 6).

3.3 SELECTIVE SEMANTIC LEARNING

For G_C, we train one shared semantic LoRA adapter while keeping the base model fixed. Figure 3 motivates this step: row correction alone often restores output without recovering semantics, while the combined components achieve 82.9% functional success in the Qwen-7B-Chat SD component experiment.

Each SD reference supplies five question paraphrases paired with its canonical answer, forming $\mathcal{D}_C$. We optimize the LoRA parameters θ (Hu et al., 2022; Zheng et al., 2024):

$$\theta_C^\star = \arg\min_\theta \sum_{(\mathbf{x},\mathbf{y})\in\mathcal{D}_C} -\log p_{M,\theta}(\mathbf{y} \mid \mathbf{x}). \tag{3}$$

The adapter has rank 4; training details appear in Section D. After training, we freeze it and collect SD calibration states with the adapter active.

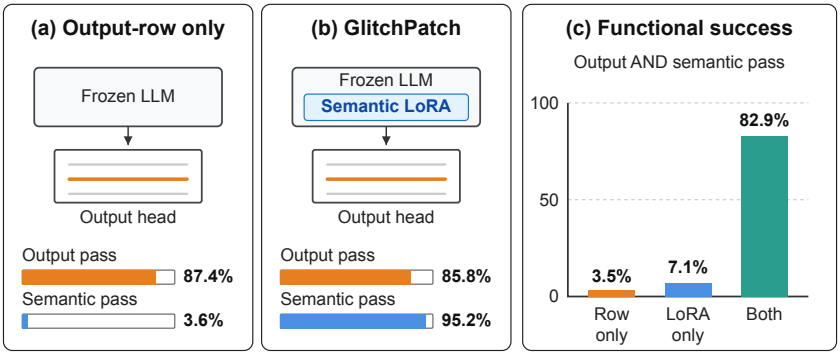


Figure 3: Semantic learning and output correction recover complementary capabilities. Independent component runs on the same 10,195 SD targets in Qwen-7B-Chat; row-repair conditions jointly install SD rows. (a,b) Output and semantic success under row correction alone and semantic LoRA followed by row correction. Orange marks the corrected output row. (c) Functional success requires both outcomes for each token. These runs are separate from the full B+C bank in Table 1; Table 8 reports integer counts.

3.4 Single-Row Output Repair

For either route, we fit the target's output row to calibration contexts in the model state used for its inference.

Definition 4 (Single-Row Output Repair)**.** *Let $\mathbf{e}_t \in \mathbb{R}^{|V|}$ be the one-hot vector for target token t, and let $\Delta_t \in \mathbb{R}^d$ be its row update. A single-row repair produces*

$$\mathbf{W}_{\mathrm{lm}}^{\star} = \mathbf{W}_{\mathrm{lm}} + \mathbf{e}_t \Delta_t^{\top}. \tag{4}$$

During this step, the Transformer, any active adapter, and every output row other than $\mathbf{W}_{\mathrm{lm}}[t]$ remain fixed.

For each target, $K = 5$ copy templates insert its atomic token ID and request its surface at the next output position. Stack the final prefix states as $H = [\mathbf{h}_1; \ldots; \mathbf{h}_K] \in \mathbb{R}^{K \times d}$, using the base model for SP and the active adapter for SD. The templates exclude evaluation prompt q_1.

Let $w_0 = \mathbf{W}_{\mathrm{lm}}[t]$ and let

$$y_i = \max_{v \neq t} \mathbf{W}_{\mathrm{lm}}[v]^{\top} \mathbf{h}_i + m, \tag{5}$$

where $m > 0$ sets the desired target logit above the strongest competing token in each calibration context. For $\lambda > 0$, we fit these target logits while penalizing changes to the original row:

$$\Delta_t^{\star} = \arg\min_{\Delta} \ \|H(w_0 + \Delta) - \mathbf{y}\|_2^2 + \lambda \|\Delta\|_2^2. \tag{6}$$

Writing $\mathbf{g} = \mathbf{y} - H w_0$, its closed-form solution is

$$\boldsymbol{\alpha} = (HH^{\top} + \lambda I_K)^{-1} \mathbf{g}, \qquad \Delta_t^{\star} = H^{\top} \boldsymbol{\alpha}, \qquad w_t^{\star} = w_0 + \Delta_t^{\star}. \tag{7}$$

This computes a $K \times K$ linear solve and projects the result through $H^{\top}$. For a hidden state $\mathbf{h}$, the target-logit change is $\mathbf{h}^{\top} \Delta_t^{\star}$.

3.5 Joint Deployment

We solve each correction against the original head in its route-specific model state, then install the full bank:

$$\mathbf{W}_{\mathrm{lm,all}}^{\star} = \mathbf{W}_{\mathrm{lm}} + \sum_{t \in G} \mathbf{e}_t (\Delta_t^{\star})^{\top}. \tag{8}$$

All corrected rows remain installed and compete in the same output distribution. For a query targeting t, the semantic adapter is active exactly when $t \in G_C$, matching that row's calibration state.

For general tasks, an input containing any token in G_C activates the adapter for all answer continuations.

The deployed state stores one repaired d-dimensional row per target, the shared adapter, and the route registry.

4 EXPERIMENTS

We evaluate functional repair with all corrected output rows installed together and measure the contributions of output correction, semantic learning, and selective adapter activation.

4.1 EXPERIMENTAL SETUP

Models and evaluated tokens. Our primary evaluation uses `Qwen/Qwen-7B-Chat`, whose output head is separately parameterized. We start from the 27,686 Qwen-7B-Chat candidate tokens released by GlitchCleaner (Fan et al., 2026) and evaluate every candidate separately with each of two repetition prompts. The intersection of the two failure sets contains 16,788 tokens and forms our repair population; each token has a fixed semantic reference.

The two screening prompts are given in Section B.4. Complete-repair experiments additionally evaluate 4,351 DeepSeek-LLM-7B-Chat, 1,249 Llama-2-7b-chat-hf, 1,868 Qwen3.5-9B, and 3,309 Qwen3-32B targets from their respective screened populations.

Semantic assessment and repair. Reference-based questions assign 6,593 pre-repair semantic passes to SP and 10,195 failures to SD. SP receives row correction; SD first receives shared semantic-adapter training on the same references with different question wording. All 16,788 rows remain installed, with the adapter active only for SD requests.

Baselines and metrics. GlitchCleaner is evaluated on matched models, token populations, prompts, references, decoders, and semantic judges. We use official adapters for Qwen-7B, DeepSeek, and Llama-2, and reproduce the method on Qwen3-32B with the adaptations in Section I. Output success requires at least one exact occurrence of the target surface after removing leading whitespace under greedy decoding; trailing whitespace is retained. Semantic success requires a correct reference-grounded answer about the token's meaning, identity, or function. Functional repair requires both outcomes for the same token (Definition 2). Prompt 1 is used for screening and post-repair output evaluation and is excluded from row calibration and semantic training. The same semantic assessment measures retention for SP and recovery for SD. We also evaluate MMLU (Hendrycks et al., 2021; Gao et al., 2024), QQP (Wang et al., 2018), and GSM8K (Cobbe et al., 2021); scoring and supplementary comparisons appear in the appendix.

4.2 FUNCTIONAL REPAIR AT VOCABULARY SCALE

Overall results. With all 16,788 repaired rows installed simultaneously, GLITCHPATCH achieves 14,519 functional repairs (86.5%), compared with 2,724 (16.2%) for GlitchCleaner, a gain of 70.3 percentage points. Output recovery and semantic success reach 89.8% and 95.4%, respectively (Table 1); complete counts appear in Table 4. Full repair achieves 78.6% on DeepSeek and 75.7% on Llama-2, compared with 24.6% and 13.5% for matched GlitchCleaner adapters. Qwen3.5-9B and Qwen3-32B reach 94.8% and 69.3%, respectively; the reproduced Qwen3-32B GC baseline achieves 21.4%. Llama-2 uses corrected calibration positions; excluding its 160 development targets yields 74.2% versus 13.4% on the remaining 1,089 targets (Section F).

Effect of pre-repair semantic state. The two repair routes achieve similar functional recovery: 6,100/6,593 (92.5%) for SP with output correction alone and 8,419/10,195 (82.6%) for SD with semantic adaptation followed by output correction. SP semantic success is 95.7% after repair, while SD semantic success rises from zero to 95.2%.

Table 1: **Functional repair across five models.** Success rates (%); GP installs all repaired B+C rows together. Functional success requires output and semantic success for the same token.

Model	N	GlitchCleaner	GlitchPatch		
		Functional ↑	Output ↑	Semantic ↑	Functional ↑
Qwen-7B-Chat	16,788	16.2	89.8	95.4	**86.5**
DeepSeek-LLM-7B-Chat	4,351	24.6	86.7	90.8	**78.6**
Llama-2-7b-chat-hf[†]	1,249	13.5	77.1	97.6	**75.7**
Qwen3.5-9B	1,868	42.5[‡]	95.4	99.2	**94.8**
Qwen3-32B	3,309	21.4[‡]	73.8	93.1	**69.3**

GP disables its semantic adapter for semantic-preserved tokens and enables it for semantic-deficient tokens. GC comparisons use matched targets within each model; — denotes an unavailable matched GC baseline. [‡]GC is reproduced for Qwen3-9B and Qwen3-32B with the adaptations in Section I; other GC entries use official adapters. [†]Llama-2 uses corrected calibration positions; 160 targets informed debugging. Excluding these targets, GP and GC achieve 74.2% and 13.4% functional success on the remaining 1,089 targets. Counts appear in Tables 4, 5 and 15.

Residual errors. Residual errors differ by route: 268/493 SP failures (54.4%) recover output but fail semantics, while 1,284/1,776 SD failures (72.3%) pass semantics but fail output. Thus, the remaining bottlenecks are semantic retention for SP and output access for SD.

4.3 Output Repair and Pre-repair Semantics

Row repair across semantic groups. We apply the same five-context row correction to both pre-repair semantic groups on Qwen-7B-Chat, DeepSeek-LLM-7B-Chat, and Llama-3.1-8B-Instruct. Each token is evaluated with one corrected row installed and no semantic training.

Functional recovery on SP. On SP, output correction alone achieves functional repair rates of 87.7% on Qwen, 80.3% on DeepSeek, and 94.9% on Llama-3.1 (Table 6). The corresponding GlitchCleaner results on the matched Qwen and DeepSeek groups are 18.3% and 50.9%. Row correction recovers token function across these model families with the Transformer unchanged. A separate multiple-choice identity assessment on 429 Mistral-7B-Instruct-v0.1 tokens yields 97.0% functional repair with isolated row correction, versus 29.6% for GlitchCleaner (Table 14); its assessment differs from the open-ended semantic evaluation above.

Output and semantic recovery on SD. The same row-only intervention recovers target generation for 99.8%, 95.9%, and 95.7% of SD tokens on Qwen, DeepSeek, and Llama, respectively. Their functional repair rates are 15.0%, 11.6%, and 1.4%; detailed results and matched GlitchCleaner comparisons appear in Table 6.

4.4 Complementary Components and Selective Activation

Semantic learning and output correction. The SD component experiment separates semantic learning from output correction on 10,195 tokens. Row correction alone recovers 87.4% of target surfaces but achieves only 3.5% functional success; combining semantic adaptation with row correction raises functional success to 82.9%. Semantic adaptation alone achieves 94.9% semantic success but only 7.2% output recovery; adding row correction raises output recovery to 85.8% while retaining 95.2% semantic success (Table 2a). Table 8 gives the complete counts.

Selective activation improves functional repair. A paired experiment holds the 9,518 resident rows and trained adapter fixed and changes only adapter activation. Activating the adapter for SD requests raises overall functional repair from 6,026/9,518 (63.3%) to 8,069/9,518 (84.8%), a gain of 21.5 percentage points over unconditional activation (Table 2b; Table 9).

The gain comes from SP: semantic success increases from 16.8% to 81.2%, and functional repair from 15.5% to 79.9%. SD functional repair remains 87.2% in both conditions. Selective activation preserves SP semantics while retaining the adapter's benefit for SD.

Table 2: **Component contributions and selective activation.** Success rates (%) under matched conditions within each panel. Output, Semantic, and Functional are abbreviated O, S, and F.

(a) semantic-deficient components: rows for the semantic-deficient group installed together

	Qwen-7B-Chat (N = 10,195)			Qwen3-32B (N = 1,828)		
Condition	O ↑	S ↑	F ↑	O ↑	S ↑	F ↑
Base	0.0	0.0	0.0	0.0	13.7	0.0
Row only	87.4	3.6	3.5	54.4	13.8	8.9
Semantic LoRA only	7.2	94.9	7.1	8.4	99.5	8.3
Semantic LoRA + row	85.8	95.2	**82.9**	73.6	99.6	**73.5**

(b) Qwen-7B-Chat activation: fixed 9,518-row bank and adapter

	semantic-preserved (N = 3,172)		semantic-deficient (N = 6,346)	All (N = 9,518)
Activation	S ↑	F ↑	F ↑	F ↑
Always active	16.8	15.5	87.2	63.3
semantic-deficient requests only	81.2	79.9	87.2	**84.8**

Panel (a) uses group-specific component runs, separate from the full B+C bank in Table 1. Qwen3-32B semantics use pooled re-evaluation with fixed initial routing; its semantic-deficient base semantic score is therefore nonzero. Panel (b) changes only adapter activation on a separate population.

4.5 Prompt Transfer and Generation Control

We compare Prompt 0 and Prompt 1 on 527 Qwen-7B-Chat and 429 Mistral-7B-Instruct-v0.1 targets that pass their frozen pre-repair assessments. Each target receives an isolated row repair, with an official GlitchCleaner adapter evaluated on the same population. Prompt 0 uses GlitchCleaner's training-template wording; Prompt 1 is excluded from both methods' repair fitting. Both prompts screen candidates; the assessment remains fixed within each model (Section G.5).

On Prompt 1, GlitchPatch's functional success reaches 81.8% and 97.0% on Qwen and Mistral, versus 17.5% and 29.6% for GlitchCleaner (Table 14). From Prompt 0 to Prompt 1, GlitchPatch's repeated-generation rates (at least three target occurrences) fall from 84.1% to 3.2% and 80.7% to 0.7%, respectively; GlitchCleaner produces no repeated targets in these conditions (Table 14).

With the full 16,788-row Qwen bank, three additional prompts excluded from screening and repair yield output recovery of 86.6%, 86.1%, and 74.9%, versus 46.8%, 24.3%, and 27.6% for GC. Two further direct-repeat and direct-copy prompts yield 87.73% and 91.26% for GP (Table 13). These measure output transfer; no new semantic evaluation is performed for each prompt.

4.6 Functional Repair on Qwen3-32B

GlitchPatch achieves 2,292/3,309 functional repairs (69.27%) on Qwen3-32B with all corrected rows installed together. The reproduced GlitchCleaner baseline repairs 707/3,309 (21.37%), a gap of 47.90 percentage points on the same targets. GlitchPatch's output recovery reaches 73.83% and semantic success reaches 93.11% (Table 1); full counts appear in Table 15. We obtain these targets by re-screening the 18,651 shared candidates, using two repetition prompts and a leading-whitespace-normalized output criterion. The initial semantic assessment assigns 1,481 targets to SP and 1,828 to SD; Section I gives the experimental settings.

Semantic learning and output correction remain complementary on the larger model. For the 1,828 SD targets, semantic LoRA alone achieves 99.51% semantic success and 8.37% output recovery. Adding row correction raises output recovery to 73.63%, with semantic success at 99.56%. Functional repair increases from 8.32% with LoRA alone to 73.47% with both components, compared with 8.86% for row correction alone (Table 2a).

Task	N	Base	GlitchCleaner	GlitchPatch
MMLU	14,042	54.294	54.045	54.615
QQP	40,430	80.9226	—	80.9201
GSM8K	1,319	47.99	—	47.61

Table 3: General-task accuracy (%) on Qwen-7B-Chat with the full 16,788-row GP bank. MMLU and QQP use candidate log likelihood; GSM8K uses five fixed demonstrations, greedy generation, and numeric-answer extraction. Only complete matched GC evaluations are included. Section G.1 gives counts and scoring details.

4.7 Joint Deployment and General-task Performance

Joint row installation. A 10,911-token deployment experiment compares request-selected rows with a fully resident row bank. Installing all rows together changes output recovery from 10,862/10,911 (99.6%) to 10,294/10,911 (94.3%). The primary results in Table 1 use the fully resident configuration. Section G.2 gives the deployment experiment's settings. Holding target rows and adapters fixed, a three-model diagnostic also finds higher SD output recovery when other repaired rows are removed, identifying row competition as one source of residual failures (Section G.3).

General-task performance. With all 16,788 repaired rows installed, Qwen-7B-Chat MMLU accuracy changes from 54.294% to 54.615%, while GlitchCleaner scores 54.045% (Table 3). QQP accuracy changes from 80.9226% to 80.9201%; GSM8K numeric-answer accuracy changes from 47.99% to 47.61%. These aggregate scores do not imply identical behavior: 2,041 MMLU predictions change, including 646 incorrect-to-correct and 601 correct-to-incorrect transitions (Section G.1). On Qwen3-32B, the full repair changes MMLU accuracy from 11,342/14,042 (80.772%) to 11,333/14,042 (80.708%), a difference of 0.064 percentage points.

Generation control. Isolated Llama row repair produces repeated targets in 786/3,826 responses (20.5%) and strict target-only output in 125/3,826 (3.3%); corresponding Qwen and DeepSeek diagnostics appear in Section G.6.

Repair construction and storage. For Qwen-7B-Chat, row repair uses no gradient training and changes one existing 4,096-dimensional output row per token. The full 16,788-row bank stores 137.5 MB of FP16 row weights. SD additionally uses one shared rank-4 adapter with 1,208,320 gradient-trained parameters, 1.60 GPU-hours of training, and a 4.84 MB checkpoint. Table 10 reports storage for the repaired rows and adapter. On one RTX 4090, GP throughput is 54.25 tokens/s with the semantic adapter merged into the model weights and 52.59 with it unmerged, versus 54.21 for the base model; unmerged-adapter throughput is 3.0% below the base (Table 11).

5 Conclusion

In this paper, we show that restoring token generation does not ensure semantic correctness and that naive repairs may damage existing understanding. Based on these findings, we propose GlitchPatch, a semantically guided framework for *functional repair*. Our method combines closed-form output-head correction with pre-repair semantic assessment, applying selective semantic learning through a shared adapter. Extensive experiments demonstrate high functional repair rates across five models, while further evaluations show low repair cost and negligible impact on benchmark accuracy and inference efficiency, validating the effectiveness and practicality of GlitchPatch.

Limitations

Our evaluation focuses on screened tokens and reference-based semantic assessment. Broader contextual validation and interactions among jointly repaired rows warrant further study; Section J discusses the scope and remaining limitations.

Ethics Statement

Glitch-token repair affects names, minority-language expressions, and technical identifiers. Errors in references and automated judgments can lead to incorrect interpretations of these tokens. We retain reference provenance and evaluator decisions to support inspection and correction.

AI Use Statement

Generative AI assisted software development, protocol design and analysis, reference and prompt construction, semantic evaluation, literature search, and manuscript drafting and editing.

The authors take responsibility for checking the code, references, analyses, and final manuscript.

Reproducibility Statement

Sections 3 and 4 specify the repair procedure and evaluation protocol; Sections C, E and F provide implementation details, baseline matching, and supplementary results. We will release code, frozen references and prompts, route labels, repaired-row artifacts, adapter configurations, model outputs, and evaluator decisions, subject to source-model licenses.

References

Orevaoghene Ahia, Sachin Kumar, Hila Gonen, Jungo Kasai, David Mortensen, Noah Smith, and Yulia Tsvetkov. Do all languages cost the same? tokenization in the era of commercial language models. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 9904–9923, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.614. URL https://aclanthology.org/2023.emnlp-main.614/.

Ehsaneddin Asgari, Yassine El Kheir, and Mohammad Ali Sadraei Javaheri. MorphBPE: A morpho-aware tokenizer bridging linguistic complexity for efficient LLM training across morphologies. *arXiv preprint arXiv:2502.00894*, 2025. URL https://arxiv.org/abs/2502.00894.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023. URL https://arxiv.org/abs/2309.16609.

Kaj Bostrom and Greg Durrett. Byte pair encoding is suboptimal for language model pretraining. In Trevor Cohn, Yulan He, and Yang Liu (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 4617–4624, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.414. URL https://aclanthology.org/2020.findings-emnlp.414/.

cafeii. vllm-glitchsplit. GitHub repository, 2026. URL https://github.com/cafeii/vllm-glitchsplit. Tokenizer hotfix that suppresses selected BPE merge rules so glitch tokens decompose into sub-tokens at inference time.

Yekun Chai, Yewei Fang, Qiwei Peng, and Xuhong Li. Tokenization falling short: On subword robustness in large language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 1582–1599, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.86. URL https://aclanthology.org/2024.findings-emnlp.86/.

Jonathan H. Clark, Dan Garrette, Iulia Turc, and John Wieting. Canine: Pre-training an efficient tokenization-free encoder for language representation. *Transactions of the Association*

for Computational Linguistics, 10:73–91, 2022. doi: 10.1162/tacl_a_00448. URL https://aclanthology.org/2022.tacl-1.5/.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021. doi: 10.48550/arXiv.2110.14168. URL https://arxiv.org/abs/2110.14168.

Yiming Cui, Ziqing Yang, and Xin Yao. Efficient and effective text encoding for Chinese LLaMA and Alpaca. *arXiv preprint arXiv:2304.08177*, 2023. URL https://arxiv.org/abs/2304.08177.

Nicola De Cao, Wilker Aziz, and Ivan Titov. Editing factual knowledge in language models. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 6491–6506, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.522. URL https://aclanthology.org/2021.emnlp-main.522/.

DeepSeek-AI. DeepSeek LLM: Scaling open-source language models with longtermism. *arXiv preprint arXiv:2401.02954*, 2024. URL https://arxiv.org/abs/2401.02954.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 10088–10115. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/1feb87871436031bdc0f2beaa62a049b-Paper-Conference.pdf.

Yibo Fan, Jingru Li, and Huan Li. GlitchCleaner: Lightweight glitch tokens repairing by lossless gated LoRA in large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 30656–30664, 2026. doi: 10.1609/aaai.v40i36.40321.

Darius Feher, Ivan Vulić, and Benjamin Minixhofer. Retrofitting large language models with dynamic tokenization. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 29866–29883, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1444. URL https://aclanthology.org/2025.acl-long.1444/.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, July 2024. URL https://zenodo.org/records/12608602. lm-evaluation-harness.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz

Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ

Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. URL https://arxiv.org/abs/2407.21783.

Jian Gu, Aldeida Aleti, Chunyang Chen, and Hongyu Zhang. Rethinking weight tying: Pseudo-inverse tying for LM stable training and updates. *arXiv preprint arXiv:2602.04556*, 2026. URL https://arxiv.org/abs/2602.04556.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2790–2799. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/houlsby19a.html.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Hakan Inan, Khashayar Khosravi, and Richard Socher. Tying word vectors and word classifiers: A loss framework for language modeling. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=r1aPbsFle.

Taku Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates. In Iryna Gurevych and Yusuke Miyao (eds.), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 66–75, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1007. URL https://aclanthology.org/P18-1007/.

Taku Kudo and John Richardson. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In Eduardo Blanco and Wei Lu (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 66–71, Brussels, Belgium, November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-2012. URL https://aclanthology.org/D18-2012/.

Sander Land and Max Bartolo. Fishing for magikarp: Automatically detecting under-trained tokens in large language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 11631–11646, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.649. URL https://aclanthology.org/2024.emnlp-main.649/.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3045–3059, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.243. URL https://aclanthology.org/2021.emnlp-main.243/.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (eds.), *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 4582–4597, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.353. URL https://aclanthology.org/2021.acl-long.353/.

Yuxi Li, Yi Liu, Gelei Deng, Ying Zhang, Wenjia Song, Ling Shi, Kailong Wang, Yuekang Li, Yang Liu, and Haoyu Wang. Glitch tokens in large language models: Categorization taxonomy and effective detection. *Proceedings of the ACM on Software Engineering*, 1(FSE):2075–2097, 2024. doi: 10.1145/3660799.

Hongyuan Lu, Zixuan Li, Zefan Zhang, and Wai Lam. SLoW: Select low-frequency words! automatic dictionary selection for translation on large language models. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 898–913, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.46. URL https://aclanthology.org/2025.emnlp-main.46/.

Hongyuan Adam Lu, Z.L., Victor Wei, Zefan Zhang, Zhao Hong, Qiqi Xiang, Bowen Cao, and Wai Lam. Adam's Law: Textual frequency law on large language models. *arXiv preprint arXiv:2604.02176*, 2026. URL https://arxiv.org/abs/2604.02176.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 17359–17372. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/6f1d43d5a82a37e89b0665b33bf3a182-Paper-Conference.pdf.

Kevin Meng, Arnab Sen Sharma, Alex J. Andonian, Yonatan Belinkov, and David Bau. Mass-editing memory in a transformer. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=MkbcAHIYgyS.

Sabrina J. Mielke, Zaid Alyafeai, Elizabeth Salesky, Colin Raffel, Manan Dey, Matthias Gallé, Arun Raja, Chenglei Si, Wilson Y. Lee, Benoît Sagot, and Samson Tan. Between words and characters: A brief history of open-vocabulary modeling and tokenization in NLP. *arXiv preprint arXiv:2112.10508*, 2021. URL https://arxiv.org/abs/2112.10508.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D. Manning. Fast model editing at scale. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=0DcZxeWfOPt.

Aleksandar Petrov, Emanuele La Malfa, Philip Torr, and Adel Bibi. Language model tokenizers introduce unfairness between languages. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 36963–36990. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/74bb24dca8334adce292883b4b651eda-Paper-Conference.pdf.

Ofir Press and Lior Wolf. Using the output embedding to improve language models. In Mirella Lapata, Phil Blunsom, and Alexander Koller (eds.), *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pp. 157–163, Valencia, Spain, April 2017. Association for Computational Linguistics. URL https://aclanthology.org/E17-2025/.

Ivan Provilkov, Dmitrii Emelianenko, and Elena Voita. BPE-dropout: Simple and effective subword regularization. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 1882–1892, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.170. URL https://aclanthology.org/2020.acl-main.170/.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Technical Report*, 2019. URL https://cdn.openai.com/better-language-models/language-models.pdf.

Jessica Rumbelow and Matthew Watkins. SolidGoldMagikarp (plus, prompt generation). LessWrong blog post, 2023. https://www.lesswrong.com/posts/aPeJE8bSo6rAFoLqg/solidgoldmagikarp-plus-prompt-generation.

Phillip Rust, Jonas Pfeiffer, Ivan Vulić, Sebastian Ruder, and Iryna Gurevych. How good is your tokenizer? on the monolingual performance of multilingual language models. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (eds.), *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 3118–3135, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.243. URL https://aclanthology.org/2021.acl-long.243/.

Mike Schuster and Kaisuke Nakajima. Japanese and Korean voice search. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 5149–5152, 2012. doi: 10.1109/ICASSP.2012.6289079.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In Katrin Erk and Noah A. Smith (eds.), *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1715–1725, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1162. URL https://aclanthology.org/P16-1162/.

Kunsheng Tang, Peigui Qi, Yide Song, Wenbo Zhou, Zhicong Huang, Qing Guo, Tianwei Zhang, Weiming Zhang, Nenghai Yu, and Jie Zhang. One bad token spoils the barrel: Assessment, detection, and remediation of glitch tokens in large language models. In *35th USENIX Security Symposium (USENIX Security 26)*, 2026. URL https://www.usenix.org/conference/usenixsecurity26/presentation/tang-kunsheng.

Yi Tay, Vinh Q. Tran, Sebastian Ruder, Jai Gupta, Hyung Won Chung, Dara Bahri, Zhen Qin, Simon Baumgartner, Cong Yu, and Donald Metzler. Charformer: Fast character transformers via gradient-based subword tokenization. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=JtBRnrlOEFN.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023a. URL https://arxiv.org/abs/2302.13971.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023b. URL https://arxiv.org/abs/2307.09288.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 353–355. Association for Computational Linguistics, 2018. doi: 10.18653/v1/W18-5446. URL https://aclanthology.org/W18-5446/.

Tianwen Wei, Liang Zhao, Lichang Zhang, Bo Zhu, Lijie Wang, Haihua Yang, Biye Li, Cheng Cheng, Weiwei Lü, Rui Hu, Chenxia Li, Liu Yang, Xilin Luo, Xuejie Wu, Lunan Liu, Wenjun Cheng, Peng Cheng, Jianhao Zhang, Xiaoyu Zhang, Lei Lin, Xiaokun Wang, Yutuan Ma, Chuanhai Dong, Yanqi Sun, Yifu Chen, Yongyi Peng, Xiaojuan Liang, Shuicheng Yan, Han Fang, and Yahui Zhou. Skywork: A more open bilingual foundation model. *arXiv preprint arXiv:2310.19341*, 2023. URL https://arxiv.org/abs/2310.19341.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016. URL https://arxiv.org/abs/1609.08144.

Zihui Wu, Haichang Gao, Ping Wang, Shudong Zhang, Zhaoxiang Liu, and Shiguo Lian. GlitchMiner: Mining glitch tokens in large language models via gradient-based discrete optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 33998–34005, 2026. doi: 10.1609/aaai.v40i40.40693. arXiv:2410.15052.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. ByT5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306, 2022. doi: 10.1162/tacl_a_00461. URL https://aclanthology.org/2022.tacl-1.17/.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. URL https://arxiv.org/abs/2505.09388.

Zhilin Yang, Zihang Dai, Ruslan Salakhutdinov, and William W. Cohen. Breaking the softmax bottleneck: A high-rank RNN language model. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=HkwZSG-CZ.

Yunzhi Yao, Peng Wang, Bozhong Tian, Siyuan Cheng, Zhoubo Li, Shumin Deng, Huajun Chen, and Ningyu Zhang. Editing large language models: Problems, methods, and opportunities. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 10222–10240, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.632. URL https://aclanthology.org/2023.emnlp-main.632/.

Qingjie Zhang, Di Wang, Haoting Qian, Liu Yan, Tianwei Zhang, Ke Xu, Qi Li, Minlie Huang, Hewu Li, and Han Qiu. Speculating LLMs' Chinese training data pollution from their tokens. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 26113–26133, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.1327. URL https://aclanthology.org/2025.emnlp-main.1327/.

Zhibo Zhang, Wuxia Bai, Yuxi Li, Mark Huasong Meng, Kailong Wang, Ling Shi, Li Li, Jun Wang, and Haoyu Wang. GlitchProber: Advancing effective detection and mitigation of glitch tokens in large language models. In *Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pp. 643–655, 2024. doi: 10.1145/3691620.3695060.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, and Zheyan Luo. LlamaFactory: Unified efficient fine-tuning of 100+ language models. In Yixin Cao, Yang Feng, and Deyi Xiong (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pp. 400–410, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-demos.38. URL https://aclanthology.org/2024.acl-demos.38/.

A Screening and Semantic-Evaluation Details

A.1 Output-Failure Screening

For the primary Qwen-7B-Chat experiment, we use the 27,686 candidate tokens released by Glitch-Cleaner (Fan et al., 2026). We run Prompt 0 and Prompt 1 on every candidate and record each outcome. The 16,788 tokens that fail the output criterion under both prompts form the repair population. Output success requires at least one exact occurrence of the target surface after removing leading whitespace; trailing whitespace is retained. Whitespace-only targets are excluded. Screening and post-repair evaluation use the same model revision, tokenizer, chat template, atomic-token insertion, FP16 precision, and greedy decoder. We preserve the original 16-example batch membership and order, with generation limits of 10 tokens for repetition and 48 for semantic assessment. Before repair, we verify exact replay of the saved base-model output token IDs and decoded text. Excluded targets remain in their original batches but do not contribute to reported results.

Prompt 1 is used for screening and post-repair output evaluation. Row calibration and SD semantic training use separate prompts.

A.2 Reference Construction

Each target has a reference prepared before repair. References describe lexical meanings, named identities, Unicode characters, visible sequences, identifiers, constructions, or formatting functions. We construct 11,174 references from collected evidence and generate the remaining 5,614 from model priors. For each entry, we record the reference text, category, provenance, construction prompt, and validation status.

A.3 Semantic Assessment and Type Assignment

The base model answers one fixed open-ended question about each token's meaning, identity, construction, or function. A model-based evaluator receives the token surface, reference, question, and response. For linguistic tokens, the rubric accepts compatible central or partial meanings, including translations, roots, stems, affixes, and host-word interpretations. For Unicode characters, it accepts a correct identity, reading, category, or conventional meaning. For code, punctuation, markup, and formatting sequences, it requires the recognizable construction or its dominant function. Bare repetition and unrelated or contradictory interpretations fail.

We use `gpt-5.6-luna` with low reasoning effort for both judgments of the same generated answer. A pre-repair answer accepted by either judgment assigns the token to SP; rejection by both assigns it to SD. These assignments remain fixed during repair. Post-repair answers follow the same rule to measure semantic retention for SP and recovery for SD. Identical complete evaluation inputs reuse the same frozen judgment pair. The evaluator is blinded to the repair method, output-recovery result, and route label.

A.4 Semantic Evaluation Protocol

Training and evaluation use the same token identities and references with different question wording. Row calibration uses copy prompts, and SD training uses semantic question paraphrases. We retain evaluation prompts, raw responses, model-based judgments, and retry logs.

B Evaluated Population and Model Details

B.1 Primary Model and Token Population

The primary end-to-end evaluation uses `Qwen/Qwen-7B-Chat` at revision `93a65d34827a3cc269b727e67004743b723e2f83`. Its input embeddings and output head are separately parameterized, allowing row repair to modify output access without changing the corresponding input embedding. We evaluate all 27,686 candidate tokens released by GlitchCleaner (Fan et al., 2026) separately with Prompt 0 and Prompt 1. The 16,788 tokens that fail both prompts form the primary repair population, and all have frozen semantic references. The pre-repair semantic assessment assigns 6,593 tokens to SP and 10,195 to SD.

We store token IDs, decoded surfaces, atomic encodings, screening outputs, references, semantic responses, evaluator decisions, and route labels for the complete population.

B.2 Additional Models

Complete-repair evaluations use separate screened populations for DeepSeek-LLM-7B-Chat ($N = 4{,}351$), Llama-2-7b-chat-hf ($N = 1{,}249$), Qwen3.5-9B ($N = 1{,}868$), and Qwen3-32B ($N = 3{,}309$). All repaired rows remain resident, with semantic adaptation enabled for SD requests. These experiments are distinct from the isolated row-only diagnostics on DeepSeek and Llama-3.1 and the isolated calibration audit on Qwen3.5-9B.

Calibration experiments cover three Qwen checkpoints with editable output heads. Portability experiments additionally evaluate Mistral-7B-Instruct-v0.1, Llama-2-7b-chat-hf, Gemma-2b-it, and DeepSeek-LLM-7B-Chat on their screened token populations. Gemma's tied input and output parameters couple each output-row edit to the corresponding input embedding. Each model is evaluated on its own candidate set and references.

B.3 Evaluation Metrics

Output recovery requires the target surface, with leading whitespace removed and trailing whitespace retained, to occur at least once in the Prompt-1 response. Semantic success requires the post-repair answer to pass the frozen reference-grounded assessment. Functional repair requires both outcomes for the same token. Calibration rank measures fit on the calibration contexts.

For matched method comparisons, model revision, token denominator, prompts, references, decoding policy, and evaluator are held fixed. We evaluate general-task accuracy with full MMLU using fixed continuation scoring. On Qwen-7B-Chat, we additionally evaluate QQP by scoring yes/no continuations and GSM8K by extracting numeric answers from greedy five-shot generations (Section G.1).

B.4 Repetition prompts

Prompt 0.
```
Question: Can you repeat the string '{TOKEN}' and return back to
me?
Answer: Here is the repeated string:
```

Prompt 1.
```
Please reproduce exactly the text inside <copy> tags. Output the
text once and nothing else.
<copy>{TOKEN}</copy>
Output:
```

Algorithm 1 Joint Row-Bank Construction

Require: Base model M, frozen semantic-preserved and semantic-deficient sets, calibration contexts, trained semantic-deficient adapter
Ensure: Joint repaired output head $\mathbf{W}_{\mathrm{lm}}^{\star}$
for each semantic-preserved token t **do**
  Extract calibration hidden states with the adapter disabled
  Solve Equation (7) for Δ_t
  Record target ranks for this isolated correction
end for
for each semantic-deficient token t **do**
  Extract calibration hidden states with the semantic-deficient adapter enabled
  Solve the same objective for Δ_t
  Record target ranks for this isolated correction
end for
Install all deltas simultaneously using Equation (8)
Evaluate output and semantic success using each target's frozen adapter state
return $\mathbf{W}_{\mathrm{lm}}^{\star}$

C Row-Repair Implementation

Definition 5 (Target-Rank Calibration Success)**.** *For an evaluated output head $\widetilde{\mathbf{W}}$, define the next-token logit $z_{v,i} = \widetilde{\mathbf{W}}[v]^{\top}\mathbf{h}_i$ and the target rank*

$$\mathrm{rank}_i(t) = 1 + \sum_{v \neq t} \mathbf{1}[z_{v,i} > z_{t,i}]. \tag{9}$$

The row repair achieves target-rank calibration success when $\mathrm{rank}_i(t) = 1$ for every calibration context i. This criterion measures calibration fit; functional repair is evaluated separately by Definition 2.

We solve the single-row objective in Definition 4 using five calibration contexts, each placing the target token at a designated next-token position. Hidden states for semantic-preserved tokens come from the base model; semantic-deficient states come from the model with the trained semantic adapter active. The calibration contexts use different prompts from Prompt 1 and the semantic-assessment question.

The main configuration uses margin $m = 5$ and ridge coefficient $\lambda = 0.01$ in the regularized row-fitting objective from Equation (6). The solver constructs the $K \times K$ Gram matrix, solves the dual system in Equation (7), and stores the repaired d-dimensional row for each token.

All semantic-preserved and semantic-deficient rows are solved against the original output head before the combined bank is installed according to Equation (8). The Transformer and adapter weights remain fixed throughout row construction. Targeted repair evaluation uses the frozen target label to select the adapter state: disabled for semantic-preserved tokens and enabled for semantic-deficient tokens. General-task evaluation selects the adapter state from semantic-deficient token occurrences in the input question, as specified in Section 3.5.

D Selective Semantic Training

We train a shared semantic adapter on semantic-deficient tokens. For each semantic-deficient token, five supervision examples present the complete token surface and request its stable meaning, identity, construction, or function using different wording. Each target response expresses the reference in an affirmative format. Training uses semantic questions with different wording from the evaluation question.

All semantic-deficient examples train one shared LoRA adapter with rank 4, alpha 4, and zero dropout, applied to the MLP `w1` and `w2` projections in zero-indexed layers 19–28. We train for four epochs with a learning rate of 2×10^{-4}, micro-batch size 2, four gradient-accumulation steps, and a maximum

sequence length of 192 tokens. The warmup ratio is 0.03, weight decay is 0.01, and the random seed is 20260819. Training contains semantic supervision only, with no repetition examples. In the primary run, the resulting dataset contains 50,975 examples; training requires 1.60 GPU-hours, reaches 15.46 GiB peak memory, and produces a 1,208,320-parameter, 4.84 MB adapter. Semantic training precedes semantic-deficient row calibration. The repaired rows are therefore fitted to the hidden states used at inference.

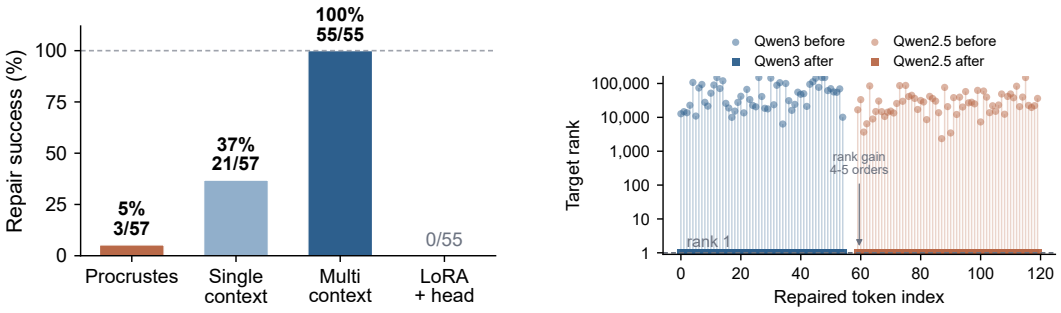
The semantic-preserved targets use the base Transformer state and their directly solved output rows.

E Baselines and Matching Protocol

The primary baseline is the official GlitchCleaner adapter (Fan et al., 2026). It is evaluated on the same base-model revision, 16,788-token denominator, Prompt 1, frozen references, decoding policy, and semantic evaluator as GlitchPatch. We report output, semantic, and functional success for the semantic-preserved group, the semantic-deficient group, and the full population. Qwen3-32B uses a separately trained GC reproduction; its architecture adaptations and matched evaluation are described in Section I.

Tokenizer-level splitting (cafeii, 2026) is evaluated on a matched split-eligible subset. GlitchEdit (Tang et al., 2026) edits the affected input embedding; GlitchPatch edits the output row.

F Supplementary Repair Results



(a) Repair variants.

(b) Target rank before and after repair.

Figure 4: Qwen3-8B calibration comparisons under the original protocol. Table 7 reports the five-context results.

Complete functional-repair counts. Table 4 gives the counts underlying Figure 5, including the two semantic groups for each method.

Route	N	Output	Semantic	Functional
GlitchPatch				
semantic-preserved: row	6,593	6,368 (96.6%)	6,311 (95.7%)	6,100 (92.5%)
semantic-deficient: LoRA + row	10,195	8,708 (85.4%)	9,703 (95.2%)	8,419 (82.6%)
Overall	**16,788**	**15,076 (89.8%)**	**16,014 (95.4%)**	**14,519 (86.5%)**
GlitchCleaner				
semantic-preserved	6,593	2,795 (42.4%)	5,006 (75.9%)	2,150 (32.6%)
semantic-deficient	10,195	2,804 (27.5%)	1,497 (14.7%)	574 (5.6%)
Overall	**16,788**	**5,599 (33.4%)**	**6,503 (38.7%)**	**2,724 (16.2%)**

Table 4: Functional repair on Qwen-7B-Chat with all 16,788 repaired rows installed together. GP and GC use the same tokens, prompts, references, decoding, and semantic judge. Functional success combines output and semantic success for each token.

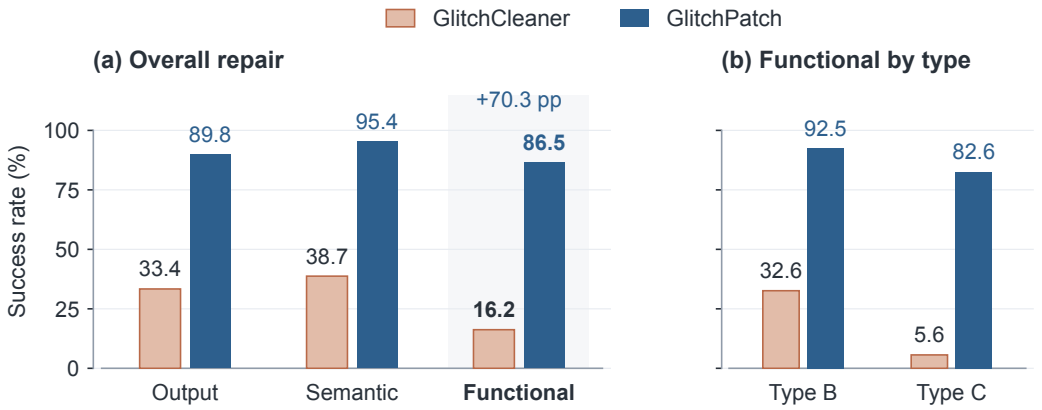


Figure 5: GlitchPatch improves functional repair in both semantic groups. Qwen-7B-Chat with all 16,788 corrected rows installed: (a) output, semantic, and joint functional success; (b) functional success on 6,593 semantic-preserved and 10,195 semantic-deficient tokens. The semantic-preserved route applies row repair; the semantic-deficient route adds semantic learning. Both methods use the same evaluation protocol; pp denotes percentage points.

Additional complete-repair experiments. Table 5 reports full resident-bank results for DeepSeek-LLM-7B-Chat, Qwen3.5-9B, and Llama-2-7b-chat-hf. Each GP condition disables the semantic adapter for semantic-preserved tokens and enables it for semantic-deficient tokens. DeepSeek and Llama-2 have matched official GC baselines; no matching official GC weights are available for Qwen3.5-9B. These complete-repair experiments use their own populations and must be distinguished from the isolated row-only diagnostics below.

The Llama-2 experiment uses corrected calibration positions. Of its 1,249 targets, 160 were used to diagnose the calibration issue. Excluding these targets from scoring gives 808/1,089 functional repairs (74.2%) for GP and 146/1,089 (13.4%) for GC. This sensitivity analysis retains the resident repair bank and does not constitute evaluation on targets excluded from repair fitting.

Model	N	GlitchPatch			GC
		Output	Semantic	Functional	Functional
DeepSeek-LLM-7B-Chat	4,351	3,773	3,952	3,421	1,071
Qwen3.5-9B	1,868	1,782	1,853	1,771	—
Llama-2-7b-chat-hf	1,249	963	1,219	945	169
Excluding development targets	1,089	—	—	808	146

Table 5: Counts for the additional complete-repair experiments in Table 1. All repaired rows remain installed; the Llama-2 sensitivity analysis excludes 160 development targets from scoring. A dash indicates an unavailable baseline or a component count not reported separately for the subset.

Cross-model semantic stratification. Table 6 expands the isolated row-only comparison in Figure 6 with semantic success and matched GlitchCleaner functional repair rates.

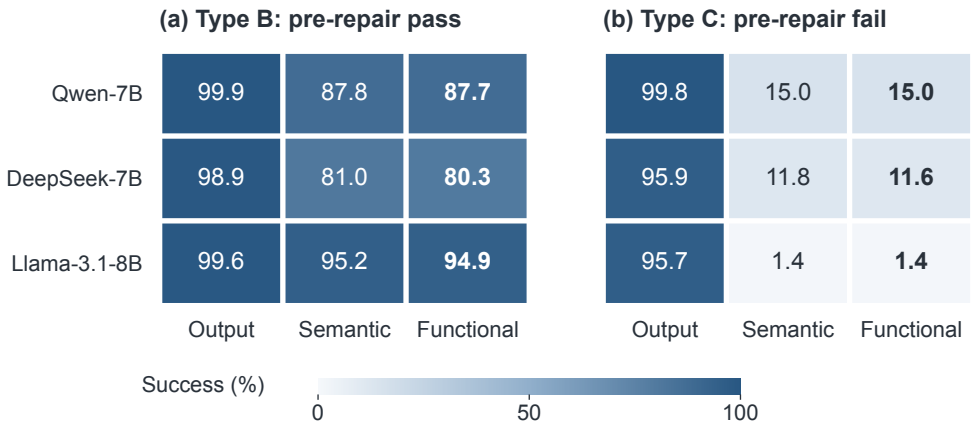


Figure 6: Pre-repair semantics predict functional recovery under row repair. Cells show output, semantic, and joint functional success (%) after isolated row-only repair. The semantic-preserved (B) and semantic-deficient (C) groups denote pre-repair semantic pass and fail, respectively. Qwen-7B-Chat, DeepSeek-LLM-7B-Chat, and Llama-3.1-8B-Instruct have B/C counts of 2,163/8,748, 1,318/3,233, and 956/2,870.

Model / group	N	**Output**	**Semantic**	**Functional**	**GC functional**
Qwen-7B-Chat / B	2,163	99.9%	87.8%	87.7%	18.3%
Qwen-7B-Chat / C	8,748	99.8%	15.0%	15.0%	2.7%
DeepSeek-7B-Chat / B	1,318	98.9%	81.0%	80.3%	50.9%
DeepSeek-7B-Chat / C	3,233	95.9%	11.8%	11.6%	8.1%
Llama-3.1-8B-Instruct / B	956	99.6%	95.2%	94.9%	n/a
Llama-3.1-8B-Instruct / C	2,870	95.7%	1.4%	1.4%	n/a

Table 6: Isolated row-only repair by pre-repair semantic group. Output, semantic, and functional columns report GP; the last column reports the official GC adapter. Each target uses the same open-ended question and reference before and after repair. A matched official GC adapter is unavailable for Llama.

Under the five-context protocol, the isolated solver fits all candidates on Qwen2.5-7B-Instruct, Qwen3-8B, and Qwen3.5-9B. Table 7 gives candidate counts and pre-repair calibration failures.

Checkpoint	Candidates	Pre-repair failures	Post-repair fit
Qwen2.5-7B-Instruct	61	59	61/61
Qwen3-8B	55	30	55/55
Qwen3.5-9B	154	64	154/154

Table 7: Calibration with five fixed contexts and isolated row installation. Pre-repair failures violate rank 1 in at least one context; post-repair fit requires rank 1 in all five. Candidate sets come from the original calibration experiment.

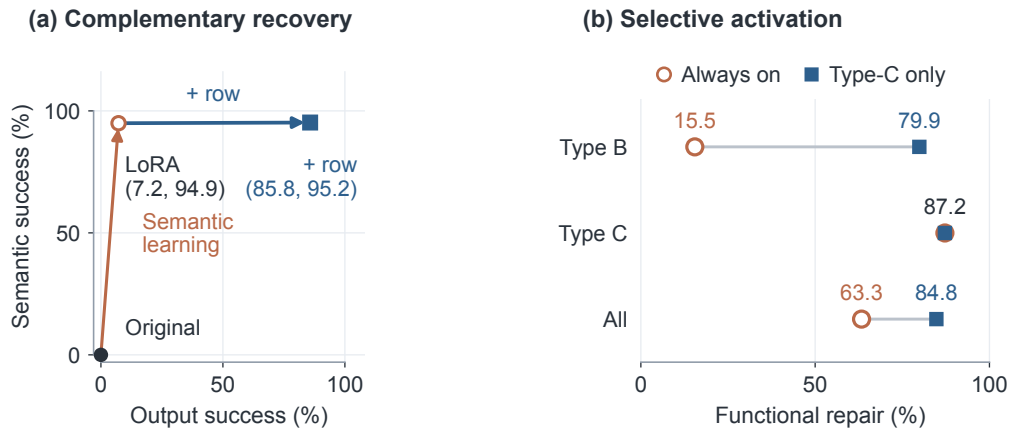


Figure 7: Semantic learning and output correction recover different capabilities. (a) Output/semantic success for 10,195 semantic-deficient tokens: original, semantic LoRA, and LoRA plus row correction. (b) Functional success with a fixed 9,518-row bank and adapter: circles denote unconditional activation; squares, activation only for semantic-deficient requests. The semantic-deficient markers overlap. The activation population contains 3,172 semantic-preserved and 6,346 semantic-deficient tokens.

Component and activation ablations. Tables 8 and 9 give the results plotted in Figure 7. The component experiment uses 10,195 semantic-deficient tokens; the activation experiment uses a separate 9,518-token population.

semantic-deficient setting	Output	Semantic	Functional
Base	0 (0.0%)	0 (0.0%)	0 (0.0%)
Row only	8,915 (87.4%)	362 (3.6%)	356 (3.5%)
Semantic LoRA only	736 (7.2%)	9,680 (94.9%)	720 (7.1%)
Semantic LoRA + row	8,743 (85.8%)	9,708 (95.2%)	8,456 (82.9%)

Table 8: Component evaluation on 10,195 semantic-deficient tokens. Row-repair conditions install semantic-deficient rows. Functional success requires both output and semantic success for each token.

Activation policy	B semantic ($N = 3{,}172$)	B functional ($N = 3{,}172$)	C functional ($N = 6{,}346$)	Overall functional ($N = 9{,}518$)
Always active	16.8%	15.5%	87.2%	63.3%
semantic-deficient requests only	81.2%	79.9%	87.2%	84.8%

Table 9: Adapter activation with a fixed 9,518-row bank and trained semantic adapter. Activation only for semantic-deficient requests improves semantic retention and functional repair for semantic-preserved tokens.

Calibration and solver sensitivity. On a fixed 57-token semantic-preserved sample from Qwen-7B-Chat, all 27 combinations of $K \in \{1, 3, 5\}$, margin $m \in \{1, 3, 5\}$, and ridge coefficient $\lambda \in \{0.01, 0.1, 1\}$ achieve rank 1 in every fitted context. The additional $K = 10$ setting and gradient-based $K = 5$ solver also fit all 57 targets.

Mistral identity assessment. A separate Mistral-7B-Instruct-v0.1 experiment evaluates 429 tokens that pass a pre-repair multiple-choice identity assessment. Isolated row repair achieves 416/429 functional repairs (97.0%), compared with 127/429 (29.6%) for the official GC adapter. This assessment tests the identity of the visible sequence; the cross-model comparison in Table 6 uses open-ended semantic questions.

Matched tokenizer-splitting comparison. On 5,452 split-eligible semantic-preserved tokens from the current Qwen-7B-Chat population, row repair recovers 5,238 target surfaces (96.1%), compared with 3,324 (61.0%) for tokenizer splitting. Functional success is 5,037/5,452 (92.4%) for row repair and 2,001/5,452 (36.7%) for splitting. The split excludes the original target ID and decodes to the exact target surface. The row condition uses the full 16,788-row bank with the semantic adapter disabled for these semantic-preserved requests.

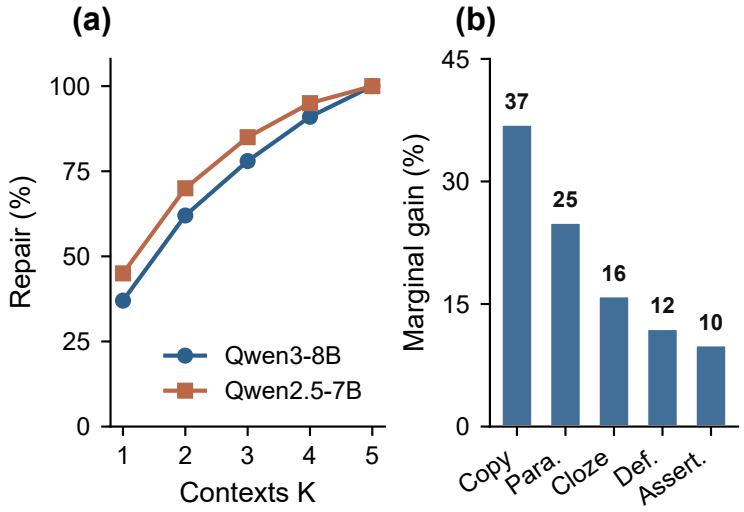


Figure 8: Effect of calibration-context count under the original protocol. The five-context experiment and Qwen-7B-Chat solver sweep appear above.

On DeepSeek-LLM-7B-Chat semantic-deficient tokens, row repair alone recovers 3,101/3,233 outputs (95.9%) but only 375/3,233 functional repairs (11.6%).

G Capability, Locality, and Deployment

The complete bank stores 68,763,648 repaired-row weights in a 137.5 MB FP16 artifact. The shared semantic-deficient adapter adds a 4.84 MB checkpoint and reaches 15.46 GiB peak training memory. Installing corrected rows replaces existing output-head weights; the adapter adds its low-rank computation when active.

Component	Gradient-trained parameters	Edited rows	Storage
semantic-preserved repaired rows	0	6,593	54.0 MB
semantic-deficient repaired rows	0	10,195	83.5 MB
Full row bank	0	16,788	137.5 MB
Shared semantic adapter	1,208,320	0	4.84 MB

Table 10: Repair footprint on Qwen-7B-Chat. The row bank stores repaired FP16 weights for installation in the existing output head; adapter storage is the saved checkpoint size.

G.1 General-task accuracy and prediction changes

All primary general-task comparisons retain the complete 16,788-row Qwen-7B-Chat bank. The adapter state follows the input-based gate in Section 3.5.

MMLU. The primary comparison uses 14,042 questions and 56,168 option-scoring requests. The base model, GP, and GC answer 7,624, 7,669, and 7,589 questions correctly, respectively (54.294%, 54.615%, and 54.045%). GP changes the predicted option on 2,041 questions (14.5%): 646 change from incorrect to correct and 601 from correct to incorrect, for a net gain of 45 correct answers. Thus, the small aggregate accuracy change coexists with substantially more individual prediction changes.

In the separate 9,518-row activation ablation, gated and always-active adapters score 7,627 and 7,636 correct answers, respectively. Their predicted options differ on 2,803 questions (20.0%): 788 change from incorrect to correct and 779 from correct to incorrect when the adapter is always active.

QQP. We evaluate all 40,430 question pairs by comparing the likelihoods of `yes` and `no` as answers to whether the pair expresses the same question. The base model answers 32,717 pairs correctly (80.9226%), and GP answers 32,716 correctly (80.9201%); only five predictions change. This result measures binary candidate scoring, rather than free-form generation.

GSM8K. We evaluate the full 1,319-question test set with five fixed demonstrations and greedy generation. Accuracy is computed by extracting the numeric answer and comparing it with the reference answer. The base model scores 633/1,319 (47.99%), and GP scores 628/1,319 (47.61%), a decrease of five correct answers.

MMLU on additional models. DeepSeek-LLM-7B-Chat changes from 49.73% to 49.74% after complete repair. Qwen3.5-9B retains 11,050/14,042 correct answers (78.69%) in both conditions. Qwen3-32B changes from 11,342/14,042 (80.772%) to 11,333/14,042 (80.708%); its prediction transitions appear in Section I.

G.2 Deployment and inference cost

Row-deployment experiment. The 10,911-token deployment comparison uses a population of 3,210 semantic-preserved and 7,701 semantic-deficient tokens. Its shared semantic-deficient adapter was trained with both reference and repetition examples. Request-selected rows recover 10,862 target surfaces (99.6%); installing all rows together recovers 10,294 (94.3%). Strict target-only output changes from 10,167/10,911 (93.2%) to 8,301/10,911 (76.1%). Both conditions retain conditional semantic-deficient adapter activation.

Inference throughput. Table 11 reports fixed-length generation on one RTX 4090. Resident row installation yields throughput close to the base model in this configuration; enabling the semantic adapter reduces throughput by 3.0% relative to the base. The measurement characterizes model generation throughput, not end-to-end request latency or adapter-switching cost.

Qwen-7B-Chat condition	Throughput (tokens/s)
Base	54.21
GlitchCleaner	53.63
GlitchPatch, semantic LoRA merged	54.25
GlitchPatch, semantic LoRA unmerged	52.59

Table 11: Inference throughput on one RTX 4090 with FP16, batch size 1, 128 input tokens, and exactly 64 generated tokens. Both GP conditions retain all 16,788 repaired rows. This measurement excludes end-to-end serving costs such as gate and adapter switching.

G.3 Competition between repaired rows

For each of Qwen-7B-Chat, DeepSeek-LLM-7B-Chat, and Llama-2-7b-chat-hf, a diagnostic samples 64 semantic-preserved and 64 semantic-deficient targets. It holds each target's repaired row and semantic adapter fixed and changes only which other repaired rows are installed. Table 12 gives the semantic-deficient output counts: retaining only the target row raises output recovery from 51 to 61, 50 to 59, and 42 to 60 out of 64, respectively. This isolates competition from other repaired rows as one source of output failures; it does not measure semantic recovery under the altered deployment.

Model	All repaired rows	Target row only
Qwen-7B-Chat	51/64	61/64
DeepSeek-LLM-7B-Chat	50/64	59/64
Llama-2-7b-chat-hf	42/64	60/64

Table 12: semantic-deficient output recovery in the row-competition diagnostic. Each comparison holds the target row and semantic adapter fixed and changes only the installation of other repaired rows. The table reports the 64 sampled semantic-deficient targets per model; semantic outcomes are not re-evaluated.

On all 685 Llama-2 semantic-deficient targets, a post-hoc ablation restores one competing row to its base-model weights. Output recovery rises from 454/685 to 520/685, but the restored row's own target loses its repair. This intervention therefore illustrates a trade-off rather than a replacement for the complete-repair method.

Calibration matched to the adapter state. A separate Llama-2 comparison recalibrates the full semantic-deficient row bank for each of two model states, with and without semantic LoRA. Output recovery is 454/685 with LoRA and 536/685 without it. The full four-condition matrix is incomplete and matched semantic outcomes are unavailable, so these counts do not establish better functional repair without semantic learning.

G.4 Additional prompts with the resident Qwen bank

Table 13 evaluates five prompts on the same 16,788 Qwen-7B-Chat targets with the complete repaired bank installed. The prompts are excluded from candidate screening, row calibration, and semantic training. Three tasks (text extraction, verbatim transcription, and filling a blank) have matched GC evaluations; direct repeat and direct copy are additional GP-only evaluations. No semantic answers are regenerated for these prompt comparisons. Where the underlying report combines output and semantic outcomes, it reuses the primary semantic judgments; the table therefore reports output recovery only.

Prompt task	GlitchCleaner	GlitchPatch
Text extraction between markers	7,860 (46.8%)	14,541 (86.6%)
Verbatim transcription	4,075 (24.3%)	14,453 (86.1%)
Fill-in-the-blank stress test	4,641 (27.6%)	12,572 (74.9%)
Direct repeat	—	14,728 (87.73%)
Direct copy	—	15,321 (91.26%)

Table 13: Output recovery under five additional prompts on all 16,788 Qwen-7B-Chat targets. GP retains the full resident row bank and semantic-deficient adapter routing. The first three prompts have matched GC results; the last two were evaluated only for GP. None of these prompts participates in screening, calibration, or training. These are output measurements, with no new semantic assessment for each prompt.

G.5 Paired prompt-transfer experiment

Qwen-7B-Chat open-ended assessment. The Qwen paired prompt experiment uses 527 pre-repair semantic-pass tokens. From the official GlitchCleaner token list, screening excludes boundary-whitespace tokens and retains 10,920 tokens failing both prompts. Frozen questions and answer keys are available for 5,033 candidates (767 lexical and 4,266 character-identity questions); 527 pass the pre-repair assessment. All 527 receive the same five-context row repair, with one corrected row installed for each target's evaluation.

Prompt 0 matches the wording of GlitchCleaner's training template, but this evaluation inserts the atomic target ID, whereas GC training tokenizes the constructed string. Neither Prompt 0 nor Prompt 1 enters row calibration; Prompt 1 is also absent from GC's visible training template. Both prompts

participate in candidate screening. Greedy decoding allows 10 generated tokens for repetition and 48 for semantic answers. The semantic question, generation, and judgment are reused across prompts, so changing the repetition prompt changes only the output component of functional success.

For semantic judging, `gpt-5.6-sol` receives the target, frozen question, answer key, and anonymized answers from row repair, the earlier split/row policy, and GC in randomized order. Each answer is judged independently as pass, fail, or uncertain; uncertain counts as unsuccessful. Target occurrence means at least one exact surface occurrence, while repeated generation means at least three occurrences in the generated suffix. The two events can overlap.

Mistral identity assessment. The paired Mistral-7B-Instruct-v0.1 experiment evaluates 429 identity-pass tokens from 1,191 dual-prompt failures with 1,082 stable targets. Each target receives an isolated five-context row repair with margin 5 and ridge coefficient 0.01; the official GC adapter is evaluated on the same targets and prompts. Both repetition prompts use atomic target-ID insertion and greedy decoding with a 10-token generation budget. The same post-repair four-choice answer is reused when computing functional success for the two prompts. The multiple-choice identity assessment differs from Qwen's open-ended assessment. All comparisons in Table 14 are paired within a model and cohort.

Table 14: **Paired prompt outcomes across two models.** Counts for Prompt 0 and Prompt 1; the assessment answer is shared across prompts within each method. GP uses isolated row repair and GC uses the official adapter. Assessment success denotes open-ended semantics for Qwen and multiple-choice identity for Mistral.

Prompt	Method	Output	Assessment	Functional	Repeated (≥ 3)
Qwen-7B-Chat ($N = 527$)					
0	GC	511	393	385	0
0	GP	512	431	416	443
1	GC	114	393	92	0
1	GP	527	431	431	17
Mistral-7B-Instruct-v0.1 ($N = 429$)					
0	GC	406	362	345	0
0	GP	399	421	391	346
1	GC	145	362	127	0
1	GP	424	421	416	3

G.6 Generation-control diagnostics

For the isolated row-only populations in Table 6, the target appears at least three times in 531/10,911 Qwen responses (4.9%), 116/4,551 DeepSeek responses (2.5%), and 786/3,826 Llama responses (20.5%). Strict target-only output succeeds for 125/3,826 Llama tokens (3.3%).

H Additional Tokenization Context

Modern LLMs use subword tokenizers such as WordPiece, BPE, SentencePiece, and subword regularization (Schuster & Nakajima, 2012; Wu et al., 2016; Sennrich et al., 2016; Kudo & Richardson, 2018; Kudo, 2018; Provilkov et al., 2020), with BPE common in large-scale models (Radford et al., 2019; Touvron et al., 2023a). Tokenizer choice affects robustness, multilingual coverage, and cross-language token cost (Bostrom & Durrett, 2020; Rust et al., 2021; Ahia et al., 2023; Chai et al., 2024; Petrov et al., 2023). Token-free and byte/character models change the input representation (Mielke et al., 2021; Xue et al., 2022; Clark et al., 2022; Tay et al., 2022). Other approaches modify segmentation through morphology-aware BPE (Asgari et al., 2025) or dynamic tokenization (Feher et al., 2025), expand the vocabulary (Cui et al., 2023; Wei et al., 2023), or use frequency-aware training and data selection (Lu et al., 2025; 2026).

I QWEN3-32B EXTENSION

Population and scoring. We evaluate `Qwen/Qwen3-32B` in BF16 with its non-thinking chat template and greedy decoding. Re-screening the 18,651 shared candidates with two repetition prompts yields 3,309 targets that fail both prompts. Output success requires an exact occurrence of the target surface after removing its leading whitespace; trailing whitespace is retained. Inputs and semantic references use the original token surface. The initial semantic assessment assigns 1,481 targets to SP and 1,828 to SD. Copy and semantic generation budgets are 32 and 48 tokens, respectively.

Repair configuration. SD training uses the canonical references, five question paraphrases per target, four epochs, and seed 20260923. The shared LoRA has rank 4, alpha 4, and zero dropout, adapting `gate_proj` and `up_proj` in zero-indexed layers 38–57. The learning rate is 2×10^{-4}, with micro-batch size 1, gradient accumulation 8, and maximum sequence length 256. Row fitting uses five calibration contexts, margin 5, and regularization coefficient 0.01, with FP32 solving and BF16 installation. Component runs install the relevant group's rows; full repair installs all 3,309 rows, with LoRA active for SD targets. Training and evaluation use different question wording for the same targets and references.

GlitchCleaner reproduction. We train GlitchCleaner on the same 3,309 targets using the released repetition template and quoted target followed by EOS, with loss applied only to the response. No semantic references are used in GC training. The implementation follows the official code at commit `26571a03` with rank 4, alpha 4, MLP gate/up projections, 15 epochs, batch size 64, and eight gradient-accumulation steps. AdamW uses a learning rate of 10^{-4}, weight decay 0.01, linear decay with 10% warmup, gradient clipping at norm 1, and seed 42. We use the final epoch after 105 optimizer steps, without selecting a checkpoint by evaluation performance.

We make three adaptations explicit. The released 32-layer example uses zero-indexed layers 19–28; we scale this interval to layers 38–57 of the 64-layer model before observing results. The frozen base uses BF16 to match GP evaluation, with FP32 LoRA parameters and non-reentrant gradient checkpointing; the released training example uses an FP32 base. Training keeps the gate active, as specified in Algorithm 1 of GlitchCleaner, while inference retains input-token gating. Literal application of the released script's training gate would disable 179 examples because contextual tokenization removes all target IDs from those sequences. We retain the released training-string construction and use the paper's training-gate rule. All matched evaluation inputs contain an atomic target ID and therefore activate GC. These results are from our adapted reproduction, rather than a released adapter.

Condition	n	Output	Semantic	Functional
B: base	1,481	0 (0.00%)	1,263 (85.28%)	0 (0.00%)
B: row only	1,481	1,183 (79.88%)	1,265 (85.42%)	1,025 (69.21%)
B: GC (reproduced)	1,481	578 (39.03%)	1,318 (88.99%)	514 (34.71%)
C: base	1,828	0 (0.00%)	250 (13.68%)	0 (0.00%)
C: row only	1,828	995 (54.43%)	252 (13.79%)	162 (8.86%)
C: semantic LoRA only	1,828	153 (8.37%)	1,819 (99.51%)	152 (8.32%)
C: semantic LoRA + row	1,828	1,346 (73.63%)	1,820 (99.56%)	1,343 (73.47%)
C: GC (reproduced)	1,828	523 (28.61%)	561 (30.69%)	193 (10.56%)
B+C: base	3,309	0 (0.00%)	1,513 (45.72%)	0 (0.00%)
B+C: full repair	3,309	2,443 (73.83%)	3,081 (93.11%)	2,292 (69.27%)
B+C: GC (reproduced)	3,309	1,101 (33.27%)	1,879 (56.78%)	707 (21.37%)

Table 15: Qwen3-32B repair results. Output uses Prompt 1 with leading whitespace removed from the target surface; semantic inputs are blinded and identical inputs share judgments. B/C labels denote the initial semantic routing. GP component runs install each group's rows; full repair installs all 3,309 rows together. All GC rows evaluate the same adapter trained on the full population (Section I).

Functional repair and prompt transfer. The full resident bank achieves 2,292/3,309 functional successes (69.27%) on Prompt 1. The corresponding Prompt 2 and Prompt 3 results are 1,951/3,309

(58.96%) and 2,068/3,309 (62.50%). The reproduced GC baseline achieves 707/3,309 functional successes (21.37%) on Prompt 1, with 1,101 output successes (33.27%) and 1,879 semantic passes (56.78%). Its Prompt 2 and Prompt 3 functional counts are 798 (24.12%) and 790 (23.87%); these reuse the same per-token semantic assessment. With Prompt 0's training-template wording in the matched chat wrapper, GC achieves 2,743 output successes (82.90%) and 1,686 functional successes (50.95%). On the 3,172 targets outside the 137-token development set, GLITCHPATCH achieves 2,196 functional repairs (69.23%). All evaluated targets participate in repair training or row fitting. Re-scoring GLITCHPATCH's 3,309 responses with the raw surface gives 2,243 joint successes (67.78%). Removing both leading and trailing whitespace gives 2,380 (71.93%).

Semantic evaluation. We pool semantic inputs across conditions, blind their labels, and reuse judgments for identical inputs. The evaluator is `gpt-5.6-luna` with low reasoning effort and the frozen semantic rubric. It accepts compatible partial linguistic meanings and requires a recognizable construction or dominant function for code and punctuation. Routing labels remain fixed from the initial assessment; the table reports semantic scores from the pooled evaluation. For the added GC condition, 1,020 exact semantic-input matches reuse frozen judgments and 2,289 new inputs are evaluated in 36 blinded batches using the same rubric and judge configuration. Every target has a verdict. Two explanations exceed the requested length limit by one or two words; we retain their original verdicts and full explanations without re-evaluation. The re-evaluated base model passes 1,263/1,481 SP targets and 250/1,828 SD targets. Relative to the routing assessment, 218 B judgments change from pass to fail and 250 C judgments from fail to pass. The repeated judgments use the same semantic inputs and judge configuration with different batch composition, order, and item identifiers.

General-task and ordinary-word controls. On 57-subject zero-shot MMLU, accuracy changes from 11,342/14,042 (80.772%) to 11,333/14,042 (80.708%). There are 128 wrong-to-right and 137 right-to-wrong changes. Both conditions use the same reconstructed inputs and continuation-log-likelihood scoring. The full row bank remains installed, and the input-based rule activates the adapter for all 14,042 questions. For 16 ordinary words evaluated with four copy prompts (64 responses), full repair changes 20 answers and converts 11 copy successes into failures. The adapter is inactive for all 64 control prompts.

J SCOPE AND LIMITATIONS

Evaluation scope. Prompt-transfer experiments test additional copying instructions, while broader contextual use remains to be evaluated.

Reference-based semantic assessment. Reference answers provide a consistent basis for routing, supervision, and evaluation. Training and evaluation use different question wording but share target tokens and references. The resulting semantic scores should therefore be interpreted within this assessment protocol. Broader reference coverage and independent human assessment would help evaluate context-dependent or ambiguous meanings.

Joint deployment and behavioral preservation. Installing corrected rows together can introduce competition (Section G.3).

In a Qwen3-32B ordinary-word control, we noticed that few previously correct copies become incorrect even with the adapter inactive (Section I). Broader control sets would help characterize and reduce unintended changes when many repaired rows are deployed together.