SMASH: Probing Speech Recognition Robustness via Semantically Targeted Bit Flips

Zafaryab Haider
Electrical and Computer Engineering
University of Maine, ME, USA
zafaryab.haider@maine.edu

Md Hafizur Rahman
Electrical and Computer Engineering
University of Maine, ME, USA
md.hafizur.rahman@maine.edu

Ayşegül Bumin *
Amazon
CA, USA
abbumin@amazon.com

Prabuddha Chakraborty †
Electrical and Computer Engineering
University of Maine, ME, USA
prabuddha@maine.edu

Abstract

Automatic speech recognition (ASR) systems are usually judged by aggregate transcript accuracy, but real failures often hinge on meaning: for example, a wrong dose, address, account number, or date can matter more than many harmless word errors. We introduce SMASH, a framework for finding targeted semantic failures caused by sparse INT8 weight-bit perturbations, a dominant model data type for ASR edge devices, in quantized sequence-to-sequence ASR models. Given an audio input and a meaning-critical span, SMASH searches for small weight changes that replace the intended meaning while keeping the transcript readable, plausible, and close to the original output. Each accepted fault is a certificate that a specific semantic substitution is reachable under a bounded perturbation budget. Empirical study focuses on numeric and scalar substitutions across three ASR backbones: Whisper-small.en (WS), Whisper-large-v3 (WLv3), and SeamlessM4T-v2-large (S-M4T); and three corpora: an LLM-assisted controlled corpus (LACC), MultiMed, and LibriSpeech. On LACC at a 20-bit budget (B), SMASH-hybrid finds strict numeric target realizations on 24/45 WS targets, 19/44 WLv3 targets, and 5/34 S-M4T targets; reachability is lower on real-speech datasets. Expanded WS evaluations find 99/387 strict target realizations on LibriSpeech and 223/939 on MultiMed. In matched LACC budget sweeps tested up to $B = 40$, WS reaches its observed ceiling already at $B = 20$, while WLv3 and S-M4T continue to gain strict target realizations through $B = 40$. Human validation audits semantic cleanliness: on a 105-item shared validation set, independent human-majority labels and a four-judge large-language-model ensemble agree on the composite `clean_success` label, with Cohen's $\kappa = 1.00$. Across the nine original model–corpus cells at $B = 20$, 56/59 strict numeric target realizations also satisfy the clean-substitution criterion. The faults are sparse: among 73 independently verified strict numeric certificates at $B = 40$, the pooled median is 10 INT8 bit flips, with medians of 2 on WS, 15 on WLv3, and 22 on S-M4T. Entity and negation pivots are evaluated as secondary evidence in the appendix. SMASH shifts parameter-fault robustness evaluation from "how much did the transcript change?" to "which meanings can be changed, how plausibly, and with how few bit flips?"

*This work was performed in the author's personal capacity. The views and opinions expressed in this paper are solely those of the author and do not reflect the views of Amazon.com, Inc. or its affiliates.

†Corresponding Author

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

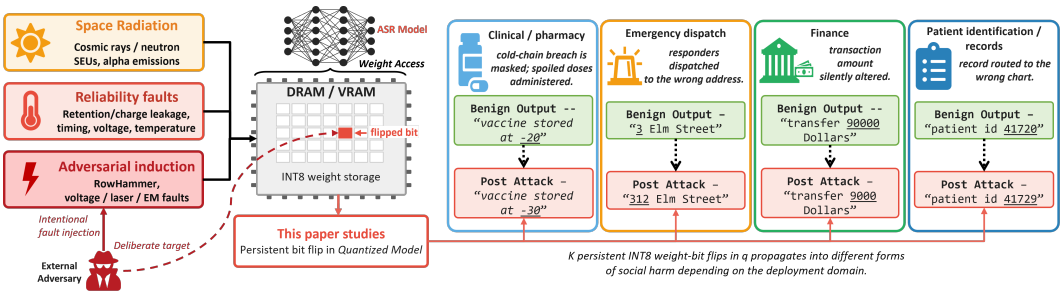


Figure 1: **Fault-to-transcript diagnostic pathway**. Sparse persistent perturbations of stored INT8 codes can arise from reliability faults or adversarial fault mechanisms. We use this setting as a preset-scoped diagnostic model to test whether bounded parameter faults can produce meaning-changing ASR transcripts. The figure motivates the robustness audit; it is not an end-to-end physical exploit or deployment-harm demonstration.

1 Introduction

Automatic speech recognition (ASR) systems are commonly evaluated using aggregate transcript accuracy, yet many deployment failures are span-local and semantic. In clinical dictation, emergency dispatch, financial transcription, administrative records, and accessibility tools, a changed number, address, identifier, date, or name can matter more than many harmless word errors. Changing "vaccine was stored at -20" to "vaccine was stored at -30" preserves fluency and most tokens, but changes an operational fact. Metrics such as WER and CER count transcription errors without distinguishing their semantic importance.

We study this gap for quantized sequence-to-sequence ASR under sparse INT8 parameter faults. This differs from conventional ASR robustness to acoustic noise or domain shift because the perturbations affect stored model weights rather than the audio. A flipped stored weight bit produces a discrete perturbation to one quantized parameter that can propagate through autoregressive decoding. We use this setting as a worst-case robustness probe, not as a physical exploit claim. The resulting question is semantic reachability: given clean audio and a declared source span, can a few weight-bit flips force a specific meaning-changing substitution while leaving the transcript readable, plausible, and close to the quantized baseline?

We introduce SMASH, a tokenization-aware framework for targeted semantic bit-flip search. SMASH constructs a declared replacement target, ranks candidate INT8 flips with a span-conditioned objective, and accepts a certificate only when decoding realizes the target under four required checks (G1-G4): transcript change, readability, proximity, and target realization. Each accepted fault is therefore a per-sample certificate of strict target realization under the tested protocol, not merely a generic transcript-change event.

Across Whisper-small.en, Whisper-large-v3, and SeamlessM4T-v2-large on LACC, MultiMed, and LibriSpeech, numeric/scalar reachability is concentrated but nonzero and model- and corpus-dependent. On LACC at $B = 20$, SMASH-hybrid finds $24/45$, $19/44$, and $5/34$ strict numeric target realizations. Expanded WS evaluations find 99/387 strict realizations on LibriSpeech and 223/939 on MultiMed. Movement-oriented baselines produce no strict target realizations in the reported comparison, while a matched one-shot target-aware cross-entropy baseline reaches 3/45 versus 24/45 for SMASH on WS-LACC at B = 20. Human-majority and four-judge LLM labels agree on the composite human-clean decision with Cohen's $\kappa = 1.00$.

Contributions: First, we define semantic-integrity reachability as an ASR parameter-fault robustness audit target: declared source→target span realization under bounded sparse quantized faults, distinct from WER drift or generic transcript movement. Second, we introduce SMASH, a span-conditioned INT8 bit-flip search that instantiates this audit target for seq2seq ASR. Third, we evaluate reachability across three ASR backbones, three corpora, and budget sweeps, showing that numeric/scalar reachability is nonzero but concentrated. Fourth, we pair the automatic G1–G4 strict certificate with a blinded, method-mixed human-clean validation pool and a verified-fault ledger for reproducible auditing. Finally, we show that existing movement-focused bit-flip baselines can change transcripts

without reaching the intended target, while SMASH achieves more strict target realizations than a simple one-shot target-aware baseline (24/45 versus 3/45) on the tested WS-LACC cases.

2 Background and Related Work

Sociotechnical ASR Risk and Semantic Evaluation: ASR increasingly mediates consequential workflows like clinical documentation, emergency dispatch, financial transcription, accessibility tools, and voice-driven interfaces. Aggregate metrics like WER do not distinguish harmless from high-cost substitutions, motivating severity-aware ASR evaluation [41, 6] and clinical studies showing recognition errors persisting into reviewed notes across terminology, speakers, and workflows [48, 23]. We probe which high-impact semantic substitutions are reachable under worst-case sparse perturbations of deployed quantized ASR models.

Hardware Bit Flips - Natural, Reliability, and Adversarial Sources: Sparse bit-level parameter faults are motivated by a literature spanning natural memory faults, runtime reliability errors, and adversarial fault injection. Cosmic-ray and alpha-particle soft errors in semiconductor memory have been documented for decades [49, 2, 36], and onboard ML for spacecraft has renewed interest in radiation robustness for deployed inference [15]. Retention variation and refresh leakage drive runtime DRAM faults [17, 12], while RowHammer and its successors like Half-Double, Blacksmith, and RowPress, demonstrate adversarially induced disturbance flips [13, 14, 11, 18]. Voltage and frequency manipulation extend the attack surface beyond DRAM [38, 20]. We use this combined literature to motivate sparse parameter faults as a robustness regime, not to claim an end-to-end physical exploit.

Quantized Neural-Network Bit-Flip Attacks: Adversarial robustness has historically studied input perturbations [37, 9, 19] with certified [42, 4] and weight-space [7, 43] extensions. Integer quantization [10] exposes deterministic bit-level effects on stored weights: PBS collapses quantized classifiers [33], DeepHammer connects bit flips to RowHammer at system scale [45], T-BFA and integer-programming variants target classifier labels [34, 1], transformer defenses analyze parameter-fault resilience [39, 21], and SilentStriker studies stealthy LLM degradation [44]. Their success events are classification failure or aggregate degradation, not decoded semantic substitution.

Input-Side ASR Attacks and the SMASH Gap: ASR adversarial work has mostly perturbed the audio waveform [3, 28, 35, 46, 22], including Whisper-specific prefix-induced muting and task control [24, 32, 31]. Within transcripts, ITN and entity-correction systems handle important content such as numbers, dates, money, identifiers, and named entities [27, 47, 40]. To our knowledge, no previous work defines this success event: declared source→target semantic span realization in seq2seq ASR under sparse quantized weight faults, with success requiring transcript change, readability, locality, and target realization. Existing bit-flip methods optimize different events, such as classifier collapse, classifier-label redirection, or LLM degradation. Thus, there is no established event-equivalent SOTA baseline; our comparisons use the closest adaptable bit-flip families under matched post-hoc gates, and Appendix Q details the event mismatch.

3 Threat Model

We study semantic reachability under sparse faults in the quantized parameters of sequence-to-sequence ASR models. Let f_θ denote an encoder–decoder ASR model with parameters θ, and let $\mathcal{D}$ denote a deterministic decoding procedure, such as greedy decoding or fixed-beam decoding with no sampling. For an audio input x, the model produces transcript: $\hat{y}(x;\theta) = \mathcal{D}(f_\theta(x))$. Unless otherwise stated, all experiments use the deterministic decoding configuration specified in Section 5: no sampling, beam size one, and maximum generation length 128.

Attacker's Capability and Non-Goals: We consider a white-box worst-case evaluator with access to the selected quantized weights of a pre-trained sequence-to-sequence ASR model. The evaluator may flip at most B bits of the stored integer codes within a declared parameter subset $\mathcal{W} \subset \theta$, which we call a *preset*. In our experiments, presets correspond to declared model-specific parameter surfaces chosen before evaluation. WS and WLv3 use the last two decoder layers and output head, while S-M4T uses the last two speech-encoder layers, adapter, and shared embedding when present. Specific preset definitions are listed in Section 5. No retraining, activation perturbation, input perturbation,

prompt modification, or audio modification is permitted: the waveform x is delivered clean, and the only degree of freedom is the choice of weight-bit flips within $\mathcal{W}$ under the stated budget.

Quantized Baseline and Preset-Scoped INT8 Representation: Evaluation is anchored to the quantized model. For each targeted weight tensor $W \in \mathbb{R}^{m\times n}$ in the preset $\mathcal{W}$, we form signed INT8 codes $q \in \{-128, \dots, 127\}^{m\times n}$ using symmetric max-absolute scaling with rounding/saturation:

$$q = \mathrm{clip}_{[-128,127]}\big(\mathrm{round}(W \oslash S)\big), \qquad \tilde{W} = q \odot S, \tag{1}$$

Where $\odot$ and $\oslash$ denote element-wise multiplication and division. We use per-row max-absolute scaling for 2D weights ($s_i = \max_j |W_{ij}|/127$, $S_{ij} = s_i$) and a single tensor-level scale for non-2D weights ($s = \max |W|/127$, with $S \equiv s$ broadcast over the tensor); explicit formulas for both cases are given in Appendix A. The choice matters for the perturbation model: per-row scaling means that the dequantized magnitude of a bit flip at location t depends on the row scale s_i, so the same bit flip in different rows yields different perturbation magnitudes in $\tilde{W}$. Let $\tilde{\theta}$ denote the model after quantizing the tensors in $\mathcal{W}$ and synchronizing the dequantized weights $\tilde{W} = q \odot S$ into the model. We anchor each sample to the deterministic transcript of the quantized baseline, $y_0 = \hat{y}(x; \tilde{\theta})$. Anchoring to y_0, rather than to a full-precision transcript, avoids treating quantization-only differences as fault-induced errors and ensures that the evaluation measures changes relative to the tested quantized baseline.

4 The SMASH Methodology

SMASH is a targeted semantic probing method for quantized sequence-to-sequence ASR (see Fig. 2). Given a clean audio input x, the quantized baseline transcript y_0, and a preset parameter subset $\mathcal{W}$, SMASH searches for a sparse fault configuration F that realizes a declared semantic target while keeping the decoded transcript readable and close to y_0. The method separates three roles that are often conflated in parameter-perturbation work: target construction, bit selection, and decoded-output validation. Target construction defines what semantic event should be tested; bit selection proposes sparse INT8 faults; decoded-output validation determines whether the event actually occurred.

Bit-upset Perturbation Model: A fault configuration is a set of flips $F = \{(u_i, t_i, b_i)\}_{i=1}^{|F|}$, where u_i indexes a tensor in the preset $\mathcal{W}$, t_i indexes a scalar code within that tensor, and b_i indexes the bit position. We enforce a global bit budget B and a per-location cap K:

$$|F| \le B, \qquad \forall (u,t):\ \big|\{(u_i,t_i,b_i) \in F : u_i = u,\ t_i = t\}\big| \le K. \tag{2}$$

The cap K limits how many bit positions can be flipped at the same scalar weight location. Let $\tilde{\theta} \oplus F$ denote the model obtained by applying the flips in F to the stored INT8 codes and resynchronizing the affected dequantized tensors. The transcript under the fault configuration is $\hat{y}_F = \mathcal{D}(f_{\tilde{\theta}\oplus F}(x))$.

Semantic Targets and Constructible Samples: For each input x, target construction operates on the quantized baseline transcript y_0. A target-construction procedure either returns no valid target or returns a tuple $\tau(x) = (k, s, r)$, where k is the semantic pivot class, s is the source span in y_0, and r is the declared target realization. The main study instantiates k with numeric/scalar pivots; entity and negation pivots are treated as secondary scope evidence in Appendix L. A sample is *target-constructible* when the target-construction procedure returns a valid tuple $\tau(x)$. We denote the set of target-constructible samples by T. Targeted success rates are reported over $|T|$, not over all N utterances, because samples without a valid semantic target do not define a target-hit event. We still report N and $|T|$ together so that target coverage remains visible. The aggregate strict rate is a protocol-scoped semantic-fault reachability measure, not an overall model-robustness score. Target-constructible sets and eligible parameter surfaces may differ across models, and an unsuccessful bounded search does not prove that no valid fault exists.

Span-Conditioned Proposal Objective: We use a span-conditioned objective that separates preserving the surrounding transcript from changing the selected pivot. Let $\ell_t(\theta) = -\log p_\theta(y_{0,t} \mid x, y_{0,<t})$ be the teacher-forced negative log-likelihood of the anchor token at position t. The objective is:

$$\mathcal{L}_{\mathrm{SMASH}}(\theta) = \frac{1}{|\mathcal{N}|}\sum_{t\in\mathcal{N}} \ell_t(\theta) - \alpha \frac{1}{|\mathcal{C}\setminus\mathcal{T}|}\sum_{t\in\mathcal{C}\setminus\mathcal{T}} \ell_t(\theta) + \gamma \mathcal{L}_{\mathrm{tgt}}(\theta) \tag{3}$$

The first term preserves the anchor transcript outside the critical region. The second term enters with a negative sign; minimizing $\mathcal{L}_{\mathrm{SMASH}}$ therefore *increases* the teacher-forced loss on the original pivot

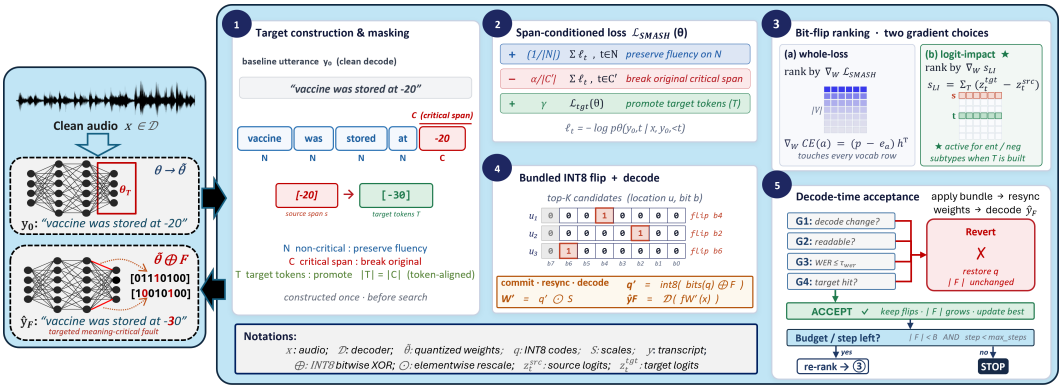


Figure 2: **SMASH Methodology:** SMASH decodes the quantized baseline, builds a declared semantic target, ranks INT8 bit flips by span-conditioned first-order effects, and tests ranked candidate bundles. Hybrid search may retain intermediate updates, but strict successes must pass G_1–G_4.

tokens, weakening the model's preference for the source semantic span. The third term promotes the declared target realization. When token-level target labels are available, we use

$$\mathcal{L}_{\text{tgt}}(\theta) = \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} -\log p_\theta(y_t^{\text{tgt}} \mid x, y_{0,<t}) \tag{4}$$

For target constructions that require source-to-target token alignment rather than direct replacement labels, we use a target logit-margin term in place of Equation 4; details in Appendix D. The target-construction and target-promotion pathway is important for strict target realization. In Appendix F, the target-enabled arm achieves 22 and 25 verified strict successes at $B = 5$ and $B = 20$, respectively. With targeting disabled, none of the 6 and 9 outputs labeled successful realizes the fixed declared target. The objective in Equation 3 is not a success criterion: it only proposes directions in parameter space that may make the declared substitution more reachable. Every proposed fault is still decoded and evaluated under the gates.

Bit-Flip Ranking: For a candidate flip at scalar coordinate u and bit position b, let $\Delta q_u(b)$ be the induced change in the signed INT8 code and let S_u be the fixed quantization scale at that coordinate. The induced dequantized perturbation is $\Delta W_u(b) = S_u \Delta q_u(b)$ For any differentiable proposal objective $R(\theta)$, a first-order approximation gives $\Delta R \approx \frac{\partial R}{\partial W_u} \Delta W_u(b)$.

We use this estimate to rank candidate bit flips inside the preset $\mathcal{W}$ subject to the global budget B, the per-location cap K, and the bit-position constraints specified in Section 5. In the main configuration, $R = \mathcal{L}_{\text{SMASH}}$ from Equation 3, and candidates are scored by the predicted decrease in R. Because $\mathcal{L}_{\text{SMASH}}$ jointly preserves non-critical context and weakens the selected pivot while promoting the target, a single ranking signal trades off all three objectives. This recovers the first-order logic of progressive bit search but replaces the global cross-entropy objective with a semantic span-conditioned one.

Ranking Variants: The implementation supports target logit-margin ranking, which scores the declared span substitution before softmax and may select different bits than teacher-forced loss. We report it as a ranking-signal ablation in Appendix D.

Bundled search over sparse faults: After ranking flips, SMASH tests small high-ranked bundles because isolated logit changes may not alter autoregressive decoding. Each step applies a bounded bundle, resynchronizes affected dequantized weights, decodes the perturbed model, and commits or reverts the bundle. Accepted bundles update F. Search stops at final accepted success, budget exhaustion, or the step limit. Certificate cost is committed flips $|F|$, not budget B. Sensitivity and hyperparameters appear in Appendix G and Section 5.

Outcome Gates: We evaluate each faulted transcript $\hat{y}_F$ using a four-gate outcome definition. Gates **(G1)**–**(G3)** measure whether the output changes while remaining readable and close to the baseline. Gate **(G4)** measures whether the declared semantic target is realized. More details about the Gates are provided in the Appendix. A *strict semantic success* $\text{Strict}(x, F) = \mathbf{1}\{\textbf{G1} \wedge \textbf{G2} \wedge \textbf{G3} \wedge \textbf{G4}\}$. A sample is a strict success if the bounded search finds a fault configuration F within budget B that

passes all four gates. Strict success requires target realization but does not necessarily require the original source span to disappear. Clean substitution additionally requires the original source value to disappear or be materially replaced by the declared target.

Outcomes: We classify outcomes by gate failure or success: **No effect** fails **G1**; **degenerate/invalid** passes **G1** but fails **G2**; **plausibility-violating drift** passes **G1–G2** but fails **G3**; **low-impact movement** passes **G1–G3** but fails **G4**; and **strict semantic success** passes **G1–G4**. We separately report source disappearance (as partial): a diagnostic indicating that the original source span is present in the baseline but absent from the faulted transcript under case-insensitive matching. It does not require G1-G3 and may overlap with strict success.

5 Experimental Setup

Unless otherwise stated, all cells use deterministic decoding, fixed random seed 42, and $N = 100$ audio inputs. Full command lines, environment details, and per-cell metadata are in Appendix P.

Models: WS [29], **WLv3** [30], and **S-M4T** [5]. All experiments use deterministic English decoding with no sampling, beam size one, and maximum generation length 128, so output changes are attributable to parameter faults rather than decoding stochasticity.

Datasets: LACC is a controlled TTS evaluation harness with manually curated pivot-controlled utterances; full construction details are in Appendix C. **MultiMed** [16] provides English medical-domain ASR with safety-relevant content such as medications, dosages, and units. **LibriSpeech** [25] is a clean-read benchmark used as a lower-pivot-density reference. The original nine model–corpus cells each use $N = 100$ utterances; we report both N and the target-constructible count $|T|$, since strict success is only defined for samples with a constructed target. Additional evaluations use 1,000 LACC, 2,620 LibriSpeech, and 4,751 MultiMed utterances, with target-constructible denominators reported separately.

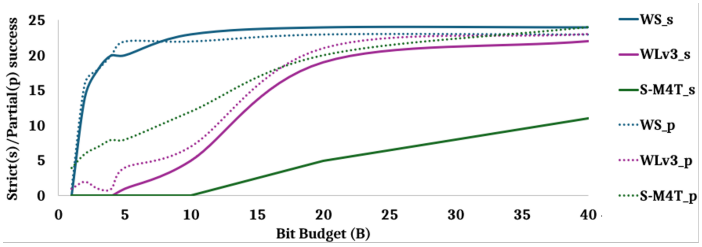
Quantization and parameter presets: All experiments use the preset-scoped INT8 quantization scheme from Section 3. WS and WLv3 use `whisper_decoder_last2_plus_lm_head`, covering the last two decoder layers and LM head; S-M4T uses `seamless_encoder_last2_plus_adapter`. Full tensor lists and quantization-only sanity checks are in Appendix A.

Canonical SMASH configuration: The main experiments use three hybrid SMASH configurations, WS-hybrid, WLv3-hybrid, and S-M4T-hybrid, each paired with its model-specific preset from above. All use automatic target construction, deterministic decoding, hybrid acceptance, `smash_decode_trials=2`, and `smash_gamma_target=10`. Headline reporting focuses on numeric/scalar pivots; entity and negation results are reported in Appendix L. Hybrid acceptance is a search strategy only: reported strict successes must still satisfy **G1**–**G4**. Candidate flips use bit positions `0..6`, excluding the sign bit. We sweep $B \in \{1, 2, 3, 4, 5, 10, 20, 40\}$, with a LACC-only B=100 sidecar control, per-location cap $K = 3$, maximum 30 outer search steps, bundle size one, and $\tau_{\text{wer}} = 0.20$. Sensitivity analyses are in Appendix G. A decode-only WS-decode ablation, using the same WS preset but strict decode acceptance, is reported in Appendix E.

Baselines: We compare against non-SMASH sparse bit-flip baselines: targeted SilentStriker-native and untargeted gated/random families, including PBS-CE, PBS-CE-gated, SilentStriker-CE, SilentStriker-CE-gated, URBF, and TGRF variants. Whenever possible, baselines use the same model, dataset, sample set, seed, target-construction pipeline, and post-hoc gates. This makes transcript movement and targeted semantic substitution event-aligned: untargeted methods need not optimize the declared target, but they count as strict successes only if their outputs realize it under the same gates. The baseline grid includes PBS variants at $B \in \{20, 40\}$, SilentStriker variants at $B = 40$, and URBF/TGRF variants at $B = 20$. We additionally evaluate a one-shot target-aware cross-entropy baseline on 45 WS–LACC numeric targets and a matched random-bit-flip baseline across all nine model–corpus cells, both at $B = 20$. SilentStriker is reported at $B = 40$ because matched $B = 20$ runs were not available for that family. Full grids and movement decompositions are in Appendix K. Appendix K.1 specifies how classifier/LLM bit-flip baselines are adapted to seq2seq ASR, including objective, decode gate, and bit-position policy. We use each method's native bit policy rather than forcing a matched policy: canonical SMASH excludes the INT8 sign bit, while PBS/URBF/TGRF allow it and SilentStriker-native is sign-bit-only. The two added controls use SMASH's parameter preset, eligible bit positions, and evaluation gates, but do not match its iterative search computation.

Table 1: Numeric/scalar reachability for SMASH-hybrid. Strict requires G1–G4. Source disappearance (as partial) indicates that the original source span is absent from the faulted transcript. Drift denotes G1–G2 outputs failing locality. Wilson 95% CIs are in Appendix J.

Corpus	Model	$\lvert T_{\text{num}}\rvert$	Strict@20	Strict@40	Δ	Partial@40	G1@40	Drift@40
LACC	WS	45	24/45	24/45	0	25/45	28/45	1/45
LACC	WLv3	44	19/44	22/44	+3	23/44	23/44	0/44
LACC	S-M4T	34	5/34	11/34	+6	24/34	29/34	3/34
MultiMed	WS	20	4/20	4/20	0	4/20	5/20	0/20
MultiMed	WLv3	26	2/26	4/26	+2	6/26	10/26	0/26
MultiMed	S-M4T	25	0/25	1/25	+1	6/25	10/25	0/25
LibriSpeech	WS	9	5/9	5/9	0	5/9	6/9	0/9
LibriSpeech	WLv3	8	0/8	1/8	+1	1/8	1/8	0/8
LibriSpeech	S-M4T	10	0/10	1/10	+1	1/10	2/10	0/10



(a) Budget sweep on LACC numeric pivots.

Model	Successes	Median $\lvert F\rvert$	≤ 5	≤ 20
WS	33	2	26 (78.8%)	33 (100%)
WLv3	27	15	1 (3.7%)	21 (77.8%)
S-M4T	13	22	0 (0%)	5 (38.5%)
Pooled	73	10	27 (37.0%)	59 (80.8%)

(b) Bit cost among strict-success.

Figure 3: **(Left)** Solid curves show strict targeted successes under the full G_1–G_4 gate cascade, while dotted curves show source disappearance (as partial), which does not necessarily realize the declared target. **(Right)** Realized bit-flip cost among independently verified strict-success certificates.

6 Results

We will release the reference implementation, per-cell commands, LACC construction scripts, verified-fault ledger, and decoded-output/gate-evaluation artifacts (Appendix O). We evaluate three questions: (i) are targeted semantic substitutions reachable under sparse INT8 weight faults, (ii) how does reachability vary across models and corpora, and (iii) does targeted semantic reachability differ measurably from generic transcript movement. Unless stated otherwise, rates are reported over target-constructible numeric samples T_{num}. WER is reported against the quantized baseline transcript and is bucketed into *no effect* ($\text{WER} = 0$), *local edit* ($0 < \text{WER} \leq 0.20$), and *drift* ($\text{WER} > 0.20$).

Numeric reachability is concentrated and model-dependent: On LACC at B=20, SMASH-hybrid finds strict numeric target realizations on $24/45$ WS, $19/44$ WLv3, and $5/34$ S-M4T pivots. The effect is therefore not a single-model artifact, but it is strongly model-dependent: Under the declared parameter presets and target-constructible sets, observed reachability is highest for WS, followed by WLv3 and S-M4T at $B = 20$. The matched budget sweep in Figure 3 supports the same pattern: WS reaches its observed ceiling by $B = 20$, while WLv3 and S-M4T expose additional successes only at higher budgets.

Expanded evaluation: To address the small original target-constructible sets, we evaluated 1,000 controlled LACC utterances and the complete available LibriSpeech and MultiMed test sets. At $B = 20$, WS reaches 99/387 strict numeric targets on LibriSpeech and 223/939 on MultiMed. The expanded LACC evaluations yield 305/453 on WS and 295/459 on WLv3 (Table 2). These results show that targeted semantic faults remain reachable on larger datasets, although the observed WS rates are lower on real speech than on controlled LACC. The original 100-utterance results across all three backbones are reported in Table 1, with full budget sweeps in Appendix H.

Clean substitution: At $B = 20$, 56 of 59 strict numeric target realizations across the nine original model–corpus cells also meet the clean-substitution criterion. The only exceptions are two WS–LACC cases (22/45 clean versus 24/45 strict) and one WS–LibriSpeech case (4/9 clean versus 5/9 strict) (see table 3). Thus, the main cross-model findings remain under the stricter clean-substitution criterion.

Table 2: Expanded numeric target reachability at $B = 20$. N is the number of screened utterances and $|T_{\text{num}}|$ is the number of constructible numeric targets.

Model–corpus	N	$\lvert T_{\text{num}}\rvert$	Strict
WS–LibriSpeech	2620	387	99/387
WS–MultiMed	4751	939	223/939
WS–LACC-1000	1000	453	305/453
WLv3–LACC-1000	1000	459	295/459

Table 3: Strict versus clean numeric substitutions at $B = 20$. Clean additionally requires source removal or replacement. Denominators are target-constructible cases.

	LACC		MultiMed		LibriSpeech	
Model	Strict	Clean	Strict	Clean	Strict	Clean
WS	24/45	22/45	4/20	4/20	5/9	4/9
WLv3	19/44	19/44	2/26	2/26	0/8	0/8
S-M4T	5/34	5/34	0/25	0/25	0/10	0/10

Strict certificates are sparse; cost is backbone-dependent: Across 73 independently verified strict numeric certificates at $B = 40$, the pooled median realized cost is 10 INT8 bit flips. Backbone-conditioned medians are 2 for WS ($n = 33$), 15 for WLv3 ($n = 27$), and 22 for S-M4T ($n = 13$).

Blinded human validation audits semantic cleanliness: On a 105-output method-mixed validation pool (Table 4), human-majority labels and a four-judge LLM ensemble agree on the composite human-clean label with $\kappa = 1.00$. This validates the composite audit decision on the sampled pool, not every automatic subgate globally: per-question agreement is lower, with $\kappa \in [0.51, 0.85]$. We also run a supplemental LLM-only audit for cross-backbone SMASH-hybrid coverage. The four-judge ensemble labels $22/26$ WS examples and $5/6$ S-M4T examples as clean; combined with the original WLv3 $22/26$, the cross-model SMASH-hybrid clean rate is $49/58$ (84.5%). The supplemental results are LLM-only and do not contribute to human–LLM agreement statistics..

Baseline comparisons: The cross-method comparison is an event-alignment diagnostic, not a direct SOTA horse race. Prior bit-flip methods optimize different events, so we adapt the closest available families and evaluate all outputs under the same post-hoc G1–G4 target-realization gates.

We compare SMASH-hybrid against four baseline families: PBS [33], SilentStriker [44], URBF (uniform random bit flips), and TGRF (top-grad random flips, with `_uniform`/`_msb`/`_ce` acceptance variants); each is adapted to seq2seq ASR per Appendix K.1.

In the matched WS–LACC comparison of Table 5, **every reported non-SMASH row lands zero strict targeted substitutions, despite extensive transcript movement.** The WER buckets show three distinct failure modes. Random or MSB-restricted baselines mostly do not move the transcript: URBF and TGRF-Uniform produce $100/100$ no-effect outputs, TGRF-MSB $98/100$. CE-driven methods move transcripts but not toward the declared target: PBS-CE at B=20 sends $80/100$ outputs into drift, SilentStriker-CE at B=40 sends $84/100$, both with zero strict successes. Gated CE methods suppress collapse and concentrate outputs in the local band, e.g., PBS-CE-gated with $88/180$ local edits, but still produce zero strict targeted substitutions. SMASH-hybrid converts movement into an targeted semantic realization that survives the readability and plausibility cascade. For example, PBS-CE can move the transcript by collapse rather than targeted substitution, rewriting "Avery Chen joined the call and greeted everyone warmly" as "A B"; for additional examples see Appendix N.3. The SilentStriker family is configured for targeted attacks, but on this matrix the auto-target builder did not engage on any sample; the runs executed under the fallback objective. We report these rows as evidence about movement and drift, not as successful SilentStriker target construction.

To address the limitation that most original baselines do not optimize the declared target, we additionally compare against a one-shot target-aware cross-entropy (CE) baseline on the same 45 WS–LACC numeric cases at $B = 20$. It achieves 3/45 strict target realizations, compared with 24/45 for SMASH, and produces degenerate transcripts in 36/45 cases. The methods share the targets, parameter surface, budget, and eligible bits, but not the number of search steps.

Table 4: Human–LLM validation summary. The 105 shared outputs yield $\kappa = 1.00$ on composite `clean_success`. The cross-backbone summary reuses 26 shared WLv3 outputs and adds 32 LLM-only outputs, which do not contribute to κ. Details are in Appendix M.

Set	Family / evidence	n	Clean	Not clean	Role
Shared human+LLM	SMASH-hybrid	26	22	4	WLv3 slice
Shared human+LLM	SMASH-decode	42	11	31	Cross-model ablation
Shared human+LLM	SS-native	7	0	7	Targeted non-SMASH
Shared human+LLM	PBS-CE-gated	16	0	16	Untargeted non-SMASH
Shared human+LLM	SS-CE-gated	6	0	6	Non-SMASH CE-gated
Shared human+LLM	TGRF-CE	8	0	8	Untargeted non-SMASH
Supplemental LLM-only	SMASH-hybrid WS	26	22	4	Backbone audit
Shared slice reused	SMASH-hybrid WLv3	26	22	4	Existing validation slice
Supplemental LLM-only	SMASH-hybrid S-M4T	6	5	1	Backbone audit
Cross-backbone LLM	SMASH-hybrid total	58	49	9	WS + WLv3 + S-M4T

Table 5: WS–LACC cross-method comparison. WER bands show no effect, local edits, and drift. Strict success requires G1–G4. N denotes evaluated records.

Method	Targeted?	B	N	No eff.	Local	Drift	Strict
SMASH (hybrid)	yes	20	100	59	40	1	**25**
SMASH (hybrid)	yes	40	100	57	42	1	**25**
PBS-CE	no	20	100	20	0	80	0
PBS-CE-gated	no	20	180	92	88	0	0
URBF	no	20	100	100	0	0	0
TGRF-CE	no	20	100	61	39	0	0
TGRF-MSB	no	20	100	98	2	0	0
TGRF-Uniform	no	20	100	100	0	0	0
PBS-CE	no	40	100	20	0	80	0
SilentStriker-native	yes	40	200	138	7	55	0
SilentStriker-CE	yes	40	100	10	6	84	0
SilentStriker-CE-gated	yes	40	200	188	12	0	0

A matched random-fault baseline across all nine model–corpus cells produces only 2/900 transcript changes and 0/221 strict numeric target realizations. These results describe the tested random sampling procedure, not fault likelihood in deployed hardware.

Locality realism re-shapes the reachable set, with preset-dependent sensitivity: Software-side locality constraints substantially reshape strict reach on the harder backbones, but do not eliminate SMASH-hybrid on the easier preset. On WS/LACC, the unconstrained baseline $24/45$ is preserved by page-local, rowhammer-row [13], cacheline-burst, blacksmith-burst [11], and collateral-noise, each also reaching $24/45$ target-realized. RowPress [18] drops WS to $15/45$, and Half-Double [14] to $22/45$. The harder backbones are more placement-sensitive. WLv3/LACC swings from $4/44$ under cacheline-burst to $17/44$ under blacksmith-burst, a $4\times$ difference on the same corpus. S-M4T/LACC produces $0/34$ strict successes under either burst mode, despite 9–17 decode changes and 9–14 partials. Locality is therefore not a binary kill-switch; it interacts with where each preset places useful committed flips in memory. We treat the software locality constraints as necessary-condition realism checks, not a complete physical-exploit simulation. We do not model DRAM row mapping [26], refresh dynamics [8], or cache coherence. Full per-mode tables are in Appendix I.

Sensitivity: A separate WER-threshold sweep shows increasing WS reachability as the threshold rises to 0.20, while WLv3 levels off earlier (Appendix G.2). Four nested WS parameter presets yield 24–25/45 strict realizations on LACC, suggesting that the result is not limited to the canonical preset.

7 Limitations

Harm and threat-model scope: We validate semantic errors in transcripts, not end-to-end harm; downstream impact depends on deployment-time interfaces, review, automation, and safeguards which are outside our scope.

Targeted-baseline scope: No established event-equivalent SOTA baseline exists for declared source→target semantic span realization in seq2seq ASR under sparse quantized weight faults. Existing bit-flip methods target classifier collapse, label redirection, or LLM degradation, so we compare closest adaptable families under matched G1–G4 gates. We also evaluate a one-shot target-aware CE baseline, but its search computation is not matched to SMASH. Thus, the comparison does not establish superiority over compute-matched targeted search methods.

Data, target-kind, and audit scope: Experiments cover preset-scoped INT8 quantization on three ASR backbones and three corpora. LACC is controlled TTS, not natural-speech deployment evidence; real-speech reachability is lower and reported separately. Numeric/scalar pivots are the main study; entity reachability is lower and negation produced zero strict successes. The 105-output validation pool audits sampled semantic cleanliness, not full-cell human validity or downstream harm.

Defenses: SMASH is an evaluation method, not a defense; certified mitigations, runtime integrity checks, and deployment-time semantic monitors are outside scope. Appendix R expands these scope.

8 Conclusion

We introduced SMASH, a framework for measuring targeted semantic reachability under sparse INT8 weight-bit faults in quantized sequence-to-sequence ASR, and instantiated it on numeric/scalar substitutions across three ASR backbones and three corpora. Targeted numeric substitutions are reachable on all three backbones in the controlled LACC setting, with lower and more uneven reachability on real-speech corpora. The realized costs are sparse (pooled median 10 INT8 bit flips among independently verified strict numeric successes at $B = 40$), but strongly model- and corpus-dependent. Independent human-majority and four-judge LLM-ensemble labels agree on the composite gate outcome at $\kappa = 1.00$, and the original movement-focused bit-flip baselines move transcripts but do not realize the declared semantic target under the same gates. A one-shot target-aware CE baseline achieves 3/45 strict target realizations, compared with 24/45 for SMASH on WS–LACC numeric targets at $B = 20$. The central conclusion is: aggregate transcript metrics can miss sparse, plausible-looking substitutions on precisely the spans whose meaning matters most, so semantic-integrity reachability is a robustness criterion distinct from WER drift or decode movement. Entity and negation pivots (Appendix L) exercise the framework beyond numeric targets.

Broader Impacts: SMASH is a robustness-auditing tool for safety-relevant ASR, where errors in numbers, identifiers, dates, or names can carry downstream cost. It tests meaning-critical spans beyond aggregate WER and can help surface transcript-level semantic failures before deployment. The work is dual-use: the threat model is a diagnostic upper bound, not a physical exploit or attack on deployed ASR services. We recommend semantic-span checks, redundancy for high-impact fields, human review, runtime integrity checks, and robustness tests beyond WER; Appendix S details deployment and release boundaries.

Acknowledgment

This work was supported by the U.S. Department of Energy, Office of Science, Basic Energy Sciences, under Award #DE-SC0026398

Competing Interests

The authors declare no competing interests.

Disclaimer

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

References

[1] Jiawang Bai, Baoyuan Wu, Yong Zhang, Yiming Li, Zhifeng Li, and Shu-Tao Xia. Targeted attack against deep neural networks via flipping limited weight bits. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=iKQAk8a2kM0`.

[2] R. Baumann. Soft errors in advanced computer systems. *Design & Test of Computers, IEEE*, 22:258 – 266, 06 2005. doi: 10.1109/MDT.2005.69.

[3] Nicholas Carlini and David Wagner. Audio adversarial examples: Targeted attacks on speech-to-text. In *Proceedings of the IEEE Security and Privacy Workshops (SPW)*, 2018.

[4] Jeremy M. Cohen, Elan Rosenfeld, and J. Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, 2019.

[5] Seamless Communication, Loïc Barrault, Yu-An Chung, Mariano Cora Meglioli, David Dale, Ning Dong, Paul-Ambroise Duquenne, Hady Elsahar, Hongyu Gong, Kevin Heffernan, John Hoffman, Christopher Klaiber, Pengwei Li, Daniel Licht, Jean Maillard, Alice Rakotoarison, Kaushik Ram Sadagopan, Guillaume Wenzek, Ethan Ye, Bapi Akula, Peng-Jen Chen, Naji El Hachem, Brian Ellis, Gabriel Mejia Gonzalez, Justin Haaheim, Prangthip Hansanti, Russ Howes, Bernie Huang, Min-Jae Hwang, Hirofumi Inaguma, Somya Jain, Elahe Kalbassi, Amanda Kallet, Ilia Kulikov, Janice Lam, Daniel Li, Xutai Ma, Ruslan Mavlyutov, Benjamin Peloquin, Mohamed Ramadan, Abinesh Ramakrishnan, Anna Sun, Kevin Tran, Tuan Tran, Igor Tufanov, Vish Vogeti, Carleigh Wood, Yilin Yang, Bokai Yu, Pierre Andrews, Can Balioglu, Marta R. Costa-jussà, Onur Celebi, Maha Elbayad, Cynthia Gao, Francisco Guzmán, Justine Kao, Ann Lee, Alexandre Mourachko, Juan Pino, Sravya Popuri, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, Paden Tomasello, Changhan Wang, Jeff Wang, and Skyler Wang. Seamlessm4t: Massively multilingual & multimodal machine translation, 2023. URL `https://arxiv.org/abs/2308.11596`.

[6] Zachary Ellis, Jared Joselowitz, Yash Deo, Yajie Vera He, Anna Kalygina, Aisling Higham, Mana Rahimzadeh, Yan Jia, Ibrahim Habli, and Ernest Lim. WER is unaware: Assessing how ASR errors distort clinical understanding in patient facing dialogue. In *Proceedings of the 16th International Workshop on Spoken Dialogue System Technology*, pages 391–417, Trento, Italy, feb 2026. Association for Computational Linguistics. URL `https://aclanthology.org/2026.iwsds-1.39/`.

[7] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=6Tm1mposlrM`.

[8] Pietro Frigo, Emanuele Vannacci, Hasan Hassan, Victor van der Veen, Onur Mutlu, Cristiano Giuffrida, Herbert Bos, and Kaveh Razavi. TRRespass: Exploiting the many sides of target row refresh. In *IEEE Symposium on Security and Privacy (S&P)*, 2020.

[9] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[10] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2704–2713, 2018. doi: 10.1109/CVPR.2018.00286.

[11] Patrick Jattke, Victor van der Veen, Pietro Frigo, Stijn Gunter, and Kaveh Razavi. BLACKSMITH: Scalable rowhammering in the frequency domain. In *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, 2022.

[12] Samira Khan, Donghyuk Lee, Yoongu Kim, Alaa R. Alameldeen, Chris Wilkerson, and Onur Mutlu. The efficacy of error mitigation techniques for dram retention failures: a comparative experimental study. In *The 2014 ACM International Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS '14, page 519–532, New York, NY, USA, 2014. Association for Computing Machinery. ISBN 9781450327893. doi: 10.1145/2591971.2592000. URL https://doi.org/10.1145/2591971.2592000.

[13] Yoongu Kim, Ross Daly, Jeremie Kim, Chris Fallin, Ji Hye Lee, Donghyuk Lee, Chris Wilkerson, Konrad Lai, and Onur Mutlu. Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors. In *Proceedings of the ACM/IEEE International Symposium on Computer Architecture (ISCA)*, 2014.

[14] Andreas Kogler, Jonas Juffinger, Salman Qazi, Yoongu Kim, Moritz Lipp, Nicolas Boichat, Eric Shiu, Mattias Nissler, and Daniel Gruss. Half-double: Hammering from the next row over. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 3807–3824. USENIX Association, 2022.

[15] Kevin Lange, Federico Fontana, Francesco Rossi, Mattia Varile, and Giovanni Apruzzese. Machine learning in space: Surveying the robustness of on-board ml models to radiation. In *2024 IEEE Space Computing Conference (SCC)*, pages 51–64. IEEE, 2024.

[16] Khai Le-Duc, Tuyen Tran, Bach Phan Tat, Nguyen Kim Hai Bui, Quan Dang, Hung-Phong Tran, Thanh-Thuy Nguyen, Ly Nguyen, Tuan-Minh Phan, Thi Thu Phuong Tran, Chris Ngo, Nguyen X. Khanh, and Thanh Nguyen-Tang. Multimed-st: Large-scale many-to-many multilingual medical speech translation, 2025. URL https://arxiv.org/abs/2504.03546.

[17] Jamie Liu, Ben Jaiyen, Richard Veras, and Onur Mutlu. Raidr: Retention-aware intelligent dram refresh. *ACM SIGARCH Computer Architecture News*, 40(3):1–12, 2012.

[18] Haocong Luo, Ataberk Olgun, A. Giray Yağlıkçı, Yahya Can Tuğrul, Steve Rhyner, Meryem Banu Cavlak, Joël Lindegger, Mohammad Sadrosadati, and Onur Mutlu. RowPress: Amplifying read disturbance in modern DRAM chips. In *Proceedings of the 50th Annual International Symposium on Computer Architecture*, pages 1–18, 2023. doi: 10.1145/3579371.3589063.

[19] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

[20] Kit Murdock, David Oswald, Flavio D Garcia, Jo Van Bulck, Daniel Gruss, and Frank Piessens. Plundervolt: Software-based fault injection attacks against intel sgx. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 1466–1482. IEEE, 2020.

[21] Najmeh Nazari, Hosein Mohammadi Makrani, Chongzhou Fang, Hossein Sayadi, Setareh Rafatirad, Khaled N. Khasawneh, and Houman Homayoun. Forget and rewire: Enhancing the resilience of transformer-based models against bit-flip attacks. In *33rd USENIX Security Symposium*. USENIX Association, 2024. URL https://www.usenix.org/conference/usenixsecurity24/presentation/nazari.

[22] Paarth Neekhara, Shehzeen Hussain, Prakhar Pandey, Shlomo Dubnov, Julian McAuley, and Farinaz Koushanfar. Universal adversarial perturbations for speech recognition systems. In *Interspeech 2019*, pages 481–485, 2019. doi: 10.21437/Interspeech.2019-1353.

[23] Joel Jia Wei Ng, Eugene Wang, Xinyan Zhou, Kevin Xiang Zhou, Charlene Xing Le Goh, Gabriel Zheng Ning Sim, Hiang Khoon Tan, Serene Si Ning Goh, and Qin Xiang Ng. Evaluating the performance of artificial intelligence-based speech recognition for clinical documentation: A systematic review. *BMC Medical Informatics and Decision Making*, 25(236), 2025. doi: 10.1186/s12911-025-03061-0. URL https://doi.org/10.1186/s12911-025-03061-0.

[24] Raphael Olivier and Bhiksha Raj. There is more than one kind of robustness: Fooling whisper with adversarial examples. In *Interspeech 2023*, pages 4394–4398, 2023. doi: 10.21437/Interspeech.2023-1105.

[25] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: An asr corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210, 2015. doi: 10.1109/ICASSP.2015.7178964.

[26] Peter Pessl, Daniel Gruss, Clémentine Maurice, Michael Schwarz, and Stefan Mangard. {DRAMA}: Exploiting {DRAM} addressing for {Cross-CPU} attacks. In *25th USENIX security symposium (USENIX security 16)*, pages 565–581, 2016.

[27] Ernest Pusateri, Bharat Ram Ambati, Elizabeth Brooks, Ondrej Plátek, Donald McAllaster, and Venki Nagesha. A mostly data-driven approach to inverse text normalization. In *Interspeech 2017*, pages 2784–2788, 2017. doi: 10.21437/Interspeech.2017-1274.

[28] Yao Qin, Nicholas Carlini, Garrison Cottrell, Ian Goodfellow, and Colin Raffel. Imperceptible, robust, and targeted adversarial examples for automatic speech recognition. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019.

[29] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision, 2022. URL https://arxiv.org/abs/2212.04356.

[30] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. *International Conference on Machine Learning*, 2023.

[31] Vyas Raina and Mark Gales. Controlling whisper: Universal acoustic adversarial attacks to control speech foundation models. *arXiv preprint arXiv:2407.04482*, 2024. URL https://arxiv.org/abs/2407.04482.

[32] Vyas Raina, Rao Ma, Charles McGhee, Kate Knill, and Mark Gales. Muting whisper: A universal acoustic adversarial attack on speech foundation models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 7549–7565, Miami, Florida, USA, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.430. URL https://aclanthology.org/2024.emnlp-main.430/.

[33] Adnan Siraj Rakin, Zhezhi He, and Deliang Fan. Bit-flip attack: Crushing neural network with progressive bit search. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1211–1220, 2019.

[34] Adnan Siraj Rakin, Zhezhi He, Jingtao Li, Fan Yao, Chaitali Chakrabarti, and Deliang Fan. T-BFA: Targeted bit-flip adversarial weight attack. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):7928–7939, 2021. doi: 10.1109/TPAMI.2021.3112932.

[35] Lea Schönherr, Katharina Kohls, Steffen Zeiler, Thorsten Holz, and Dorothea Kolossa. Adversarial attacks against automatic speech recognition systems via psychoacoustic hiding. In *Network and Distributed System Security Symposium*, 2019. URL https://www.ndss-symposium.org/ndss-paper/adversarial-attacks-against-automatic-speech-recognition-systems-via-psychoacoustic-hiding/.

[36] Bianca Schroeder, Eduardo Pinheiro, and Wolf-Dietrich Weber. Dram errors in the wild: A large-scale field study. In *SIGMETRICS*, 2009.

[37] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014.

[38] Adrian Tang, Simha Sethumadhavan, and Salvatore Stolfo. {CLKSCREW}: Exposing the perils of {Security-Oblivious} energy management. In *26th USENIX Security Symposium (USENIX Security 17)*, pages 1057–1074, 2017.

[39] Jialai Wang, Ziyuan Zhang, Meiqi Wang, Han Qiu, Tianwei Zhang, Qi Li, Zongpeng Li, Tao Wei, and Chao Zhang. Aegis: Mitigating targeted bit-flip attacks against deep neural networks. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 2329–2346, 2023.

[40] Yi-Cheng Wang, Hsin-Wei Wang, Bi-Cheng Yan, Chi-Han Lin, and Berlin Chen. DANCER: Entity description augmented named entity corrector for automatic speech recognition. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation*, pages 4333–4342, Torino, Italia, 2024. ELRA and ICCL. URL `https://aclanthology.org/2024.lrec-main.387/`.

[41] Ryan Whetten and Casey Kennington. Evaluating and improving automatic speech recognition using severity. In *Proceedings of the 22nd Workshop on Biomedical Natural Language Processing and BioNLP Shared Tasks*, pages 79–91, Toronto, Canada, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.bionlp-1.6. URL `https://aclanthology.org/2023.bionlp-1.6/`.

[42] Eric Wong and J. Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, 2018.

[43] Dongxian Wu, Shu-Tao Xia, and Yisen Wang. Adversarial weight perturbation helps robust generalization. In *Advances in Neural Information Processing Systems*, 2020.

[44] Haotian Xu, Qingsong Peng, Jie Shi, Huadi Zheng, Yu Li, and Cheng Zhuo. Silentstriker: Toward stealthy bit-flip attacks on large language models. *arXiv preprint arXiv:2509.17371*, 2025.

[45] Fan Yao, Adnan Siraj Rakin, and Deliang Fan. Deephammer: Depleting the intelligence of deep neural networks through targeted chain of bit flips. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 1463–1480, 2020.

[46] Xuejing Yuan, Yuxuan Chen, Yue Zhao, Yunhui Long, Xiaokang Liu, Kai Chen, Shengzhi Zhang, Heqing Huang, Xiaofeng Wang, and Carl A. Gunter. CommanderSong: A systematic approach for practical adversarial voice recognition. In *27th USENIX Security Symposium*, pages 49–64. USENIX Association, 2018. URL `https://www.usenix.org/conference/usenixsecurity18/presentation/yuan-xuejing`.

[47] Yang Zhang, Evelina Bakhturina, Kyle Gorman, and Boris Ginsburg. NeMo inverse text normalization: From development to production. In *Interspeech 2021*, pages 4468–4472, 2021. doi: 10.21437/Interspeech.2021-1571.

[48] Li Zhou, Suzanne V. Blackley, Leigh Kowalski, Raymond Doan, Warren W. Acker, Adam B. Landman, Evgeni Kontrient, David Mack, Marie Meteer, David W. Bates, and Foster R. Goss. Analysis of errors in dictated clinical documents assisted by speech recognition software and professional transcriptionists. *JAMA Network Open*, 1(3):e180530, 2018. doi: 10.1001/jamanetworkopen.2018.0530.

[49] J. F. Ziegler and W. A. Lanford. Effect of cosmic rays on computer memories. *Science*, 206 (4420):776–788, 1979. doi: 10.1126/science.206.4420.776.

A Quantization Details and Parameter Presets

This appendix gives the quantization and parameter-surface details used by the threat model and experimental setup. The goal is to make the perturbation domain explicit: SMASH does not perturb arbitrary floating-point weights, but operates on stored INT8 codes inside a declared parameter preset $\mathcal{W}$.

A.1 INT8 quantization

We use the preset-scoped INT8 quantization defined in eq. (1) of Section 3; here we record the implementation-side specifics that fix the perturbation domain. For 2D weights, scales are computed per output row ($s_i = \max_j |W_{ij}|/127$); for non-2D tensors, a single tensor-level scale ($s = \max |W|/127$) is broadcast over the tensor. Rows whose maximum absolute value is zero are assigned $s_i = 1$ to avoid division by zero; the corresponding codes are uniformly zero, so this case is degenerate by construction. The scale tensor S is computed once at quantization time and is not modified by the perturbation search: faults change only the integer code tensor q, while $\widetilde{W} = S \odot q$ is recovered through the fixed scales. Parameters outside the declared preset $\mathcal{W}$ remain at their full-precision values.

Table 6: **INT8 quantization scheme.** Two-dimensional weights use rowwise max-absolute scaling. Non-two-dimensional tensors use a single tensor-level max-absolute scale broadcast over the tensor.

Tensor type	Scale definition	Dequantization
2D weight matrix	$s_i = \max_j \|W_{ij}\|/127$, $S_{ij} = s_i$	$\widetilde{W}_{ij} = S_{ij} q_{ij}$
Non-2D weight tensor	$s = \max \|W\|/127$, $S \equiv s$	$\widetilde{W} = S \odot q$
Zero-scale case	scale set to 1	avoids division by zero
Code range	$q = \mathrm{clip}(\mathrm{round}(W/S), -128, 127)$	signed INT8 codes

A.2 Bit-flip semantics

A candidate perturbation flips one bit in one stored INT8 code. The implementation interprets the signed code q through its unsigned 8-bit view $q_u = q \bmod 256$ (equivalently, `q & 0xFF` in code), toggles bit b by XOR with 2^b, and remaps the result back to the signed range $[-128, 127]$:

$$q'_u(b) = (q_u \oplus 2^b), \qquad q'(b) = \begin{cases} q'_u(b) - 256 & \text{if } q'_u(b) \geq 128, \\ q'_u(b) & \text{otherwise,} \end{cases} \qquad \Delta q(b) = q'(b) - q.$$

The induced dequantized perturbation at scalar coordinate u is

$$\Delta W_u(b) = S_u \, \Delta q_u(b),$$

where S_u is the fixed quantization scale at u from §A.1; only q_u is touched by a flip, so the dequantized perturbation is linear in the pre-computed scale. Canonical SMASH restricts $b \in \{0, \ldots, 6\}$ and excludes the sign bit, with per-location cap $K = 3$, so no scalar INT8 coordinate may absorb more than three bit flips in reported canonical runs; this policy and cap are summarized in Table 7. Untargeted baselines (e.g., PBS, random) may permit the sign bit and use different caps; their bit policies are reported with the corresponding baseline appendix.

A.3 Parameter-preset filtering

Each run restricts writes to a declared parameter preset $\mathcal{W}$. The selector first filters to valid weight tensors: parameters must be non-empty, end in `.weight`, and must not be biases or layer-normalization parameters. The preset then selects an architecture-specific subset of those valid weights.

For Whisper-small.en and Whisper-large-v3, the canonical preset is

`whisper_decoder_last2_plus_lm_head`.

It selects the last two decoder layers and the output head candidate used by the model implementation. The selector treats `proj_out.weight`, `lm_head.weight`, and decoder embedding/output-tied weights as output-head candidates when present.

Table 7: **Canonical bit-flip policy.** The main SMASH-hybrid runs flip stored INT8 codes under a fixed bit-position policy and per-location cap. Baselines may use different policies, which are reported separately.

Quantity	Canonical setting
Stored representation	signed INT8 code $q \in [-128, 127]$
Bit interpretation	two's-complement unsigned toggle, then remap to signed INT8
Allowed bit positions	$\{0, 1, 2, 3, 4, 5, 6\}$
Sign bit	excluded in canonical SMASH
Per-location cap	$K = 3$
Dequantized perturbation	$\Delta W_u(b) = S_u \, \Delta q_u(b)$
Scale update	scales are fixed during perturbation search

Table 8: **Canonical parameter presets.** Each preset is applied after filtering to valid weight tensors: non-empty parameters ending in `.weight`, excluding biases and layer-normalization weights.

Model family	Preset	Selected parameter surface
WS / WL	`whisper_decoder_last2_plus_lm_head`	last two decoder layers plus output-head / output-embedding candidate
S-M4T	`seamless_encoder_last2_plus_adapter`	last two speech-encoder layers, speech adapter, and shared embedding when present

For SeamlessM4T-v2-large, the canonical preset is

`seamless_encoder_last2_plus_adapter`.

It selects the last two layers of `speech_encoder.encoder`, the speech-encoder adapter weights, and the shared embedding matrix when present. This preset is intentionally acoustic-side: it probes whether sparse perturbations to the speech-encoding and adapter surface can induce semantic substitutions in the generated text.

A.4 Preset sanity checks

If a recognized preset matches zero tensors, the runner raises an error rather than silently falling back to all weights. This prevents accidental experiments on an unintended parameter surface. The selected tensor names are logged with each run cell through the command, metadata, and per-sample artifacts described in Appendix P. The compact verified-fault ledger stores the preset name and cell directory, while per-flip tensor names and coordinates are retained in the per-sample JSONL records described in Appendix O.

The preset definitions are therefore part of the measurement protocol: cross-model differences should be read as differences under the stated parameter surfaces, not as claims about unconstrained access to every model weight.

B Outcome Gates and Strict-Rate Definition

This appendix gives the explicit math for the gate cascade summarized in Sections 3 and 4. We evaluate each faulted transcript $\hat{y}_F$ using a four-gate outcome definition. Gates **G1**–**G3** measure whether the output changes while remaining readable and close to the quantized baseline. Gate **G4** measures whether the declared semantic target is realized.

G1 Decode change. The normalized faulted transcript differs from the normalized quantized baseline:

$$\mathrm{norm}(\hat{y}_F) \neq \mathrm{norm}(y_0). \tag{5}$$

Normalization removes punctuation, casing, whitespace variation, and equivalent numeric formatting.

G2 Readability. The output passes an automatic readability and degeneracy guard: it is non-empty, has bounded length ratio relative to y_0, and contains no extreme character-level or word-level

repetition loops. This is an automatic well-formedness proxy, not a claim of human-perceived naturalness.

G3 Transcript proximity. The output remains close to the quantized baseline under edit distance:

$$\mathrm{WER}(\hat{y}_F, y_0) \leq \tau_{\mathrm{wer}}. \tag{6}$$

In all main experiments, $\tau_{\mathrm{wer}} = 0.20$. This gate excludes global transcript collapse while allowing localized semantic drift.

G4 Target realization. For target-constructible samples with target tuple $\tau(x) = (k, s, r)$, the faulted transcript realizes the declared target r in the aligned neighborhood of the source span s, according to pivot-class–specific canonicalization and alignment rules:

$$\mathrm{Hit}_k(\hat{y}_F; y_0, s, r) = 1. \tag{7}$$

Here Hit_k canonicalizes the relevant surface forms for class k—for example, numeric-form equivalence for numeric pivots, entity-surface matching for named entities, and polarity matching for negation—and requires the target realization to be localized to the selected semantic pivot rather than appearing only elsewhere in the transcript. Localization is enforced by requiring the canonicalized target r to occur in the alignment window of s, computed by token-level alignment against y_0; targets appearing only outside this window are not counted as hits. The pivot-specific hit rules are detailed in Section 4. This gate is evaluated for any method on matched target-constructible samples; targeted methods optimize for it, untargeted baselines do not.

A *strict semantic success* is a fault configuration F for which all four gates hold:

$$\mathrm{Strict}(x, F) = \mathbf{1}\{\mathbf{G1} \wedge \mathbf{G2} \wedge \mathbf{G3} \wedge \mathbf{G4}\}. \tag{8}$$

For a set of target-constructible samples T, we report strict rate as

$$\mathrm{StrictRate} = \frac{1}{|T|} \sum_{x \in T} \mathbf{1}\{\exists F : \ |F| \leq B, \ \mathrm{Strict}(x, F) = 1\}. \tag{9}$$

This gate structure makes the comparison to untargeted baselines well-defined. Baselines may achieve decode movement or high-WER drift without satisfying the target-realization gate. Conversely, a baseline can only count as a strict semantic success if it realizes the same declared target under the same gates.

C LACC Construction

LACC is a 100-utterance controlled text-to-speech (TTS) evaluation harness designed to expose targeted semantic substitutions on four pivot classes. This appendix accompanies the references in Section 5 and documents motivation, composition, construction and validation protocol, pivot-density statistics, TTS configuration, released artifacts, and scope. LACC is not a natural-speech benchmark; real-speech evidence is reported separately on MultiMed and LibriSpeech.

C.1 Motivation: why a controlled corpus is needed

Targeted semantic-reachability evaluation requires four conditions that no single public ASR corpus jointly satisfies. **(i) Pivot density.** Real-speech corpora used in this work yield small target-constructible counts ($|T_{\mathrm{num}}| \in \{8, 9, 10\}$ on LibriSpeech, 20–26 on MultiMed; see Appendix J), which limits budget-conditioned plateau analysis and produces wide Wilson confidence intervals. **(ii) Decode-baseline cleanliness.** Strict success is defined relative to the quantized baseline transcript y_0. If y_0 already mis-transcribes the pivot span, the strict-rate event becomes confounded with quantization-only error rather than fault realization. Real-speech recordings carry acoustic noise, accent variation, and speaker effects that can perturb y_0 on the pivot span itself, blurring the attack-success signal. **(iii) Pivot purity.** Attributing observed reachability differences to pivot class requires clean separation between numeric-only, entity-only, negation-only, and mixed pivots. Natural utterances rarely admit such a partitioning by construction, so a real-speech-only design would entangle pivot-class effects with utterance-level confounds. **(iv) Reproducibility without**

audio redistribution. Public ASR corpora with high-density safety-relevant pivots are commonly encumbered by redistribution restrictions; a controlled-TTS corpus lets others regenerate the audio from the released text and TTS pipeline, removing that licensing barrier.

LACC is not a substitute for natural-speech evaluation. Its purpose is to provide a small, controlled, deterministic-prosody anchor on which targeted-reachability claims can be measured cleanly; real-speech generalization is then assessed separately on MultiMed and LibriSpeech.

C.2 Composition

The corpus contains 100 utterances, evenly split across four pivot categories:

- `numeric_only` (25 rows): a single numeric pivot, e.g. *"She took 5 mg of ibuprofen before bed last night"* with `pivot_span="5 mg"`.
- `entity_only` (25 rows): a single named-entity pivot (person or organization), e.g. *"Avery Chen joined the call and greeted everyone warmly"* with `pivot_span="Avery Chen"`.
- `negation_only` (25 rows): a single negation marker, e.g. *"I did not hear the last word clearly over the noise"* with `pivot_span="did not"`.
- `mixed_all_three` (25 rows): a contiguous span containing one element from each of the other three classes, e.g. *"Avery Chen did not take 5 mg of aspirin after lunch"* with `pivot_span="Avery Chen did not take 5 mg"`.

Each row carries fields `id`, `category`, `pivot_type`, `pivot_span`, `reference`, and `audio_path`; the manifest is `datasets/synthetic/synthetic_wa.jsonl`.

C.3 Construction and validation protocol

References were authored by hand and not LLM-generated; the "LLM-assisted" label refers to seed prompting during early drafting only. An LLM was used informally to elicit candidate sentence shells per category; the authors manually verified a small number of these candidates and then rewrote or discarded the rest before any audio was rendered. The corpus prose released in `synthetic_wa.jsonl` is the post-rewrite output, not the LLM-elicited material; no specific model identifier or prompt template is therefore declared.

Each utterance was reviewed to satisfy three constraints: (i) the `pivot_span` appears as a contiguous substring of `reference`; (ii) the pivot is embedded in a complete sentence with surrounding context tokens, so that plausibility (**G3**) and proximity (τ_{wer}) gates remain meaningful; and (iii) for the three single-class categories, no out-of-class pivot is present (e.g., `numeric_only` samples contain no negation markers). Length is constrained to short clauses so that TTS prosody is uniform and search budgets are comparable across utterances.

Validation is a manual review pass on every candidate against the three constraints above, plus a check for duplicate `pivot_span` reuse that would compromise category purity, plus a TTS-comprehensibility check that the rendered audio is decodable to text close to the reference under all three backbones in the unperturbed quantized configuration. No automated named-entity-recognition or negation-detection tool was applied; constraint enforcement is fully manual by design, since the corpus is small enough that a single review pass per row is feasible and avoids tool-induced false positives on category purity. Candidates that failed any check were rewritten and re-reviewed; the released corpus is the post-review output.

C.4 Pivot-density statistics

- Reference length: 9–13 words (mean 10.2), 41–73 characters (mean 56.5).
- Single pivot span per utterance.
- Pivot uniqueness: 93 unique `pivot_span` values across 100 rows; the seven repeats are common negation markers (e.g., *did not*, *can't*) reused across distinct sentences.
- Pivot length: 1–3 tokens for entity and negation pivots; 1–4 tokens for numeric pivots (digit + optional unit such as *mg*, *ml*, *degrees Celsius*, *bpm*); 5–10 tokens for mixed pivots, which span all three classes.

The numeric/scalar subset used in the main study covers the 25 `numeric_only` rows plus the numeric component of `mixed_all_three`; target-constructible counts $|T_{\text{num}}|$ per backbone differ slightly because the automatic target-construction procedure (Section 4) requires the source span to be token-alignable under each backbone's tokenizer. Entity and negation subsets are reported in Appendix L.

C.5 TTS configuration

Audio is generated by `src/TTS.py` using the Coqui TTS Python API with the model `tts_models/en/ljspeech/vits` (single-speaker English VITS trained on LJSpeech). Decoding is deterministic. Each utterance is rendered to one mono 16-bit PCM WAV at 16 kHz, matching the input sample rate expected by Whisper-small.en, Whisper-large-v3, and SeamlessM4T-v2-large with no further resampling. Audio durations span 2.82–7.62 seconds (mean 4.49, median 4.31); the total corpus duration is approximately 7.5 minutes. Audio files are written to `datasets/synthetic/audio/{id}.wav`.

For reproducibility, the TTS pipeline used the Coqui `TTS` package (`TTS==0.22.0`) with model `tts_models/en/ljspeech/vits`. The canonical invocation is:

```
python -m src.TTS -in <seed.jsonl> -out datasets/synthetic/synthetic_wa.jsonl
-audio_dir datasets/synthetic/audio -backend coqui -model
tts_models/en/ljspeech/vits
```

Rendered audio is converted to 16 kHz mono with `ffmpeg`:

```
ffmpeg -i <in.wav> -ac 1 -ar 16000 <out.wav>
```

The released WAV files are therefore stored in the same 16 kHz mono format used for evaluation.

C.6 Released artifacts

The supplementary release contains, for the LACC corpus: (i) the JSONL manifest `datasets/synthetic/synthetic_wa.jsonl` with one row per utterance carrying `id`, `category`, `pivot_type`, `pivot_span`, `reference`, and `audio_path`; (ii) rendered 16 kHz mono 16-bit PCM WAV files at `datasets/synthetic/audio/{id}.wav`—the full 100-utterance set (≈ 7.5 minutes, ≈ 15 MB uncompressed) when the supplementary-material size budget permits, otherwise a 15-utterance demo subset (`demo.zip`) with the remaining audio recoverable from the manifest; and (iii) the generation script `src/TTS.py` with the canonical invocation listed above, so that users can re-render the full corpus from the manifest. Audio for the public corpora MultiMed and LibriSpeech is not redistributed; for those, the release provides per-cell sample manifests so that the evaluated subsets can be reconstructed from public sources.

C.7 Scope and limitations

LACC is a controlled-prosody, single-speaker TTS harness. Recording conditions, acoustic variability, accent diversity, and speaker variation are absent by design. The TTS backbone is fixed (Coqui VITS / LJSpeech); changing the synthesizer would change the prosody distribution presented to the ASR backbones and could shift the per-cell strict-rate ceiling. LACC results should be read as controlled reachability evidence on rendered audio, not as natural-speech generalization. Real-speech evidence is reported separately on MultiMed (medical-domain ASR with safety-relevant numeric and entity content) and LibriSpeech (clean-read benchmark, lower pivot density).

D Target Logit-Margin Ranking

This appendix accompanies the brief reference in Section 4. We describe the target logit-margin ranking variant implemented for entity and negation pivots, give its first-order derivation, and report a matched ON/OFF ablation on Whisper-small.en $\times$ LACC.

D.1 Motivation: cross-entropy and logit margins rank different flips

The canonical SMASH bit-flip ranking uses a first-order expansion of the span-conditioned proposal objective $\mathcal{L}_{\mathrm{SMASH}}$ in Equation 3. For a candidate flip at INT8 coordinate u and bit position b, the ranking score is

$$\widehat{\Delta R}_u(b) = \frac{\partial \mathcal{L}_{\mathrm{SMASH}}}{\partial W_u} \cdot S_u \, \Delta q_u(b).$$

This score combines all active proposal terms—preserving non-critical context, weakening the source span, and promoting the declared target—through their joint effect on teacher-forced cross-entropy.

For target promotion, however, the cross-entropy gradient at the target token is not generally proportional to the gradient of the pre-softmax source-to-target logit margin. The cross-entropy gradient depends on the full softmax distribution, including off-target vocabulary items. A source-to-target logit margin depends only on the two logits being contrasted at the selected pivot. The two ranking rules can therefore prefer different bit candidates: the flip that most reduces target cross-entropy need not be the flip that most increases the target token relative to the source token.

D.2 Target logit-margin ranking

Let $\mathcal{T} = \{t : y_t^{\mathrm{tgt}} \text{ is defined at position } t\}$ denote the set of target-span positions for which the constructed target provides a token. For a position $t \in \mathcal{T}$, let $z_{\theta,t}(y_t^{\mathrm{tgt}})$ denote the pre-softmax logit assigned to the declared target token, and let $z_{\theta,t}(y_{0,t})$ denote the pre-softmax logit assigned to the source token from the quantized baseline transcript. The pointwise target logit margin is

$$m(t;\theta) = z_{\theta,t}(y_t^{\mathrm{tgt}}) - z_{\theta,t}(y_{0,t}),$$

and the span-level margin score is its sum over the target span, $M(\theta) = \sum_{t\in\mathcal{T}} m(t;\theta)$. For a candidate bit flip at coordinate u and bit position b, the first-order effect on the span-level score is

$$\widehat{\Delta M}_u(b) = \frac{\partial M}{\partial W_u} \cdot S_u \, \Delta q_u(b).$$

For entity and negation samples with a constructed target label, candidates with larger predicted $\widehat{\Delta M}$ are preferred. Decode-time acceptance, the gate cascade, and the budget protocol are unchanged; only the candidate ranking signal is replaced.

In implementation, this variant is enabled by `-smash_ent_logit_impact_ranking 1` and `-smash_neg_logit_impact_ranking 1`. When active on a sample with a valid target term, the ranking gradient is obtained by back-propagating $-M(\theta)$, so that the first-order bit ranking prefers flips predicted to increase target preference at the selected pivot. The resulting records are marked by ranking-source metadata such as `target_logit_impact_ent` or `target_logit_impact_neg`.

D.3 Matched ON/OFF ablation

We evaluated target logit-margin ranking ON versus OFF on Whisper-small.en × LACC at $B \in \{5, 20\}$, holding all other settings fixed: hybrid acceptance, $\gamma = 10$, bundle size 1, $K = 3$, $\tau_{\mathrm{wer}} = 0.20$, and seed 42. Table 9 reports the matched outcomes. Partial counts use the source-gone rule: the declared source span must be present in y_0 and absent from $\hat{y}_F$ under case-insensitive matching. This measure does not require G1–G4 and is not a union with strict success.

Table 9: **Target logit-margin ranking ablation on WS × LACC.** The variant is applied to entity and negation ranking. Numeric strict success is unchanged, while partial counts on entity pivots decrease (negation is flat $1 \to 1$).

B	Ranking	Strict (num)	Partial (ent)	Partial (neg)	strict total
5	default	20	9	1	21
5	logit-margin	20	5	1	21
20	default	24	10	1	25
20	logit-margin	24	6	1	25

Two observations follow. First, the strict reachable set is unchanged in this matched comparison: the same headline numeric strict count and the same overall strict total are recovered under either ranking configuration. This is expected because numeric pivots dominate strict success in this cell, while the logit-margin variant is designed to affect entity and negation ranking behavior. Second, entity partials drop substantially under logit-margin ranking, from $9 \rightarrow 5$ at $B{=}5$ and $10 \rightarrow 6$ at $B{=}20$. Negation partials are flat at 1 across all four cells, so this ablation is informative for entity behavior only; the negation row is reported for completeness.

The entity pattern is consistent with the motivation above: margin scoring focuses directly on the source-to-target contrast at the selected pivot and commits fewer flips that disturb the source span without realizing the declared target, whereas cross-entropy ranking can also move probability mass through off-target vocabulary directions.

D.4 Why default ranking remains canonical

The ablation does not establish a strict-rate advantage for target logit-margin ranking. It changes the entity partial-outcome profile but does not change the headline strict reachable set in the matched WS $\times$ LACC comparison. We therefore retain the default span-conditioned cross-entropy ranking as the canonical configuration in the main paper because it is simpler, closer to the first-order progressive-bit-search template, and sufficient for the dominant numeric reachable-set finding. In smaller audits focused specifically on entity behavior, target logit-margin ranking may be useful because it produces cleaner partial accounting; we treat it as a ranking-signal ablation rather than a core component of SMASH.

E Decode-Only vs. Hybrid Acceptance

This appendix accompanies the reference in Section 5. We compare the strict decode-only reference configuration against the hybrid acceptance strategy used in the main paper. The comparison separates the report-time success criterion from the search-time acceptance rule.

E.1 Decode-only acceptance

Decode-only acceptance commits a candidate only when the full gate cascade **G1**–**G4** holds on the decoded transcript after the candidate is applied. This is the strictest search-time acceptance rule: every committed update is already a verified semantic fault under the threat model, with no surrogate-driven intermediate moves.

The decode-only configuration retained here corresponds to the family-A configuration developed first during method tuning. It uses Whisper-small.en, the `whisper_decoder_last2_plus_lm_head` preset, decode-only acceptance, `smash_decode_trials=6`, target strength $\gamma = 20$, and the auxiliary settings `smash_neg_shrinking_target=1`, `smash_ent_logit_impact_ranking=1`, `smash_neg_logit_impact_ranking=1`, and `smash_neg_bundle_size=3`. We retain this configuration as a strict-acceptance reference, not as the canonical main-paper configuration.

E.2 Reference comparison

Table 10 reports strict-success count on Whisper-small.en $\times$ LACC across the budget sweep. Hybrid acceptance is the canonical configuration of the main paper: $\gamma = 10$, `smash_decode_trials=2`, default ranking, and default bundle settings. Decode-only uses the family-A configuration described above.

The decode-only reference plateaus at 2 strict successes from $B{=}2$ onward; the hybrid configuration crosses double digits at $B{=}2$ and saturates at 25 successes by $B{=}20$.

E.3 Interpretation: search-time bridge versus report-time success

The gap is not a relaxation of the reported success criterion. Both configurations report only certificates that satisfy the full gate cascade **G1**–**G4** at decode time on the final committed flip set. The difference is the search-time rule used to reach that final state.

Table 10: **Strict-success counts at N=100.** At B=20 the hybrid configuration recovers roughly 12.5× more strict successes than the decode-only reference; note that the reference also differs in γ, `smash_decode_trials`, ranking flags, and negation bundle settings (see §E, "Decode-only acceptance"), so the gap should be read as a strict-acceptance reference rather than an isolated acceptance-rule ablation.

Acceptance	B=1	B=2	B=3	B=4	B=5	B=10	B=20	B=40
Decode-only reference	0	2	2	2	2	2	2	2
Hybrid canonical	0	14	18	20	21	24	25	25

Decode-only acceptance rejects moves that improve the proposal objective but have not yet caused the decoded transcript to realize the declared target. In an autoregressive decoder, small parameter perturbations often shift internal scores without crossing the decode boundary. Hybrid acceptance permits the search to retain such surrogate-improving moves while an intermediate WER cap is held, allowing several small parameter changes to accumulate before the decoded output transitions. A sample is still counted only when the final decoded transcript satisfies **G1**–**G4**.

Because the decode-only reference also differs in several tuning knobs, its absolute count is a strict-acceptance reference rather than a perfectly isolated ablation of acceptance strategy. The qualitative conclusion is nevertheless clear: requiring every intermediate committed state to satisfy **G4** makes the search much less efficient, while hybrid acceptance recovers the reachable set without relaxing the final reporting metric.

F Span-Conditioned Objective Ablation

This appendix accompanies the reference in Section 4. We ablate the target-enabled component of the span-conditioned objective in the v27 implementation and report its effect on decoded outcomes. Because the target-off arm disables target construction entirely, this appendix reports raw outcome counts over N=100 rather than StrictRate over the target-constructible denominator $|T|$.

F.1 Setup

The span-conditioned objective in Equation 3 combines a non-critical preservation term, a negative-weighted loss on the original pivot context, and a target-promotion term when a target label is available. In the released v27 implementation, the closest valid target-off arm is obtained by setting `-smash_target_enable 0`. This disables target construction and therefore conditionally removes the $\gamma \cdot \mathrm{target_CE}$ contribution while leaving the masked α-break objective active. There is no separate `ce` setting for `-smash_mask_style`, so this is the nearest implementation-faithful ablation of the target-promotion pathway.

We evaluate this ON/OFF contrast on Whisper-small.en × LACC at $B \in \{5, 20\}$ with N=100, seed 42, `max_steps`=30, hybrid acceptance, and $\gamma_{\mathrm{target}} = 10$ in the target-enabled arm. The canonical target-enabled arm constructs targets on $|T| = 92$ samples. The target-off arm constructs no targets by design, so $|T| = 0$ for that arm and StrictRate is undefined. For this reason, Table 11 reports raw counts over N=100 rather than rates over $|T|$, along with fixed-target hits among success-labeled outputs.

F.2 Result

Table 11 reports the matched outcomes. Disabling target construction and target promotion removes G4 from the runner's success check and shifts the movement profile toward fewer local edits and more high-WER drift at B=20.

At $B = 5$, the target-enabled arm achieves 22 strict successes, while none of the 6 success-labeled outputs from the target-disabled arm realizes the fixed target. At $B = 20$, the corresponding results are 25 strict successes versus 0 target realizations among 9 success-labeled outputs. The target-disabled arm skips G4, so its success labels indicate pivot changes rather than verified strict target realizations. Among outputs labeled successful by the runner, median realized cost is 2 flips with targeting enabled and 3 with targeting disabled; these groups do not meet the same success criterion.

Table 11: **Span-conditioned objective ablation on WS × LACC.** Counts are over N=100 samples. The target-off arm disables target construction with `-smash_target_enable 0`, so $|T| = 0$ and StrictRate is undefined for that arm. $|F|$ median is computed over runner-labeled successful outputs in each arm. The target-off arm skips G4, so its successes are not strict target realizations. Target hit is evaluated against fixed targets from the enabled arm, among outputs labeled successful by the runner. The disabled arm skips G4.

Configuration	B	Success label	Target hit	Local	Drift	$\|F\|$ Median
target_enable=1	5	22	22/22	43	7	2
target_enable=0	5	6	0/6	32	7	3
target_enable=1	20	25	25/25	44	1	2
target_enable=0	20	9	0/9	21	7	3

At B=20, local edits fall from 44 to 21, while high-WER drift rises from 1 to 7, consistent with source disruption becoming less anchored to a declared target.

F.3 Interpretation

This ablation should be read as an implementation-faithful test of the combined target-construction and target-promotion pathway in v27. It is not a denominator-matched StrictRate comparison, because disabling `smash_target_enable` also disables construction of the target tuple used to define $|T|$. The appropriate comparison is therefore the raw decoded outcome profile over the same N=100 samples.

Under that comparison, the target-enabled pathway finds declared target realizations, while the target-disabled arm produces pivot changes but no fixed-target hits among its success-labeled outputs. This supports the main methodological claim: SMASH is not merely an untargeted source-disruption search. The experiment evaluates target construction and target promotion together, rather than isolating either component.

G Sensitivity Sweeps

This appendix accompanies the references in Sections 4 and 5. We sweep three search and acceptance hyperparameters on Whisper-small.en × LACC at B=20: bundle size, transcript-proximity threshold, and target strength. The goal is to check whether the canonical configuration lies in a stable regime rather than at a knife-edge setting.

All sweep cells hold the non-swept knobs at family-A tuning (γ=20 where not swept, LIR on with `smash_ent_logit_impact_ranking`=1 and `smash_neg_logit_impact_ranking`=1, `smash_neg_shrinking_target`=1, and `smash_neg_bundle_size`=3). The main-paper canonical configuration uses γ=10 and LIR off but the same swept-knob values (bundle size 1, $\tau_{\text{wer}} = 0.20$); these sweeps therefore probe the stability of the swept knobs, not a denominator-matched main-paper ablation.

G.1 Bundle size

Bundle size is the number of high-ranked candidate flips that SMASH evaluates jointly per search step (Section 4). Larger bundles cross the autoregressive decoder boundary more aggressively but can commit more flips per step. Table 12 reports strict success and realized cost across bundle sizes 1, 2, and 3 with all other settings held fixed.

Bundle sizes 1 and 2 produce the same strict count in this sweep. Bundle size 3 produces four additional strict successes at higher realized cost. We adopt bundle size 1 as canonical because it gives more parsimonious certificates while preserving the main reachable set.

G.2 Plausibility threshold τ_{wer}

The transcript-proximity component of gate **G3** requires $\text{WER}(\hat{y}_F, y_0) \leq \tau_{\text{wer}}$. The canonical setting is $\tau_{\text{wer}} = 0.20$. Table 13 sweeps τ_{wer} on WS and WLv3 at B=20.

Table 12: **Bundle-size sensitivity on WS × LACC at B=20.** Strict counts are reported for the matched sweep. The $|F|$ mean is computed over strict-success samples only. Larger bundles slightly increase strict count but also increase realized perturbation cost.

Bundle size	**Strict**	$\|F\|$ **Mean**
1	25	3.2
2	25	3.9
3	29	4.8

Table 13: **Transcript-proximity threshold sensitivity at B=20.** Counts report strict successes as τ_{wer} is varied. The final column disables the plausibility gate **G3** (`-mislead_require_plausible 0`) and should be read as an upper-bound diagnostic rather than a reported operating point.

Model	τ=0.10	τ=0.15	τ=0.20	τ=0.25	τ=0.30	**G3 off**
WS	17	21	25	26	27	29
WLv3	16	21	21	21	21	21

WS strict success rises as the threshold relaxes from 0.10 to 0.20, then increases only modestly beyond that point. WLv3 reaches its plateau by $\tau_{\text{wer}} = 0.15$. The canonical $\tau_{\text{wer}} = 0.20$ lies in a stable regime for both backbones.

G.3 Target strength γ

The target-promotion term in $\mathcal{L}_{\text{SMASH}}$ is weighted by γ. Table 14 sweeps γ on WS × LACC at B=20.

Table 14: **Target-strength sensitivity on WS × LACC at B=20.** Strict success is approximately constant across the tested range.

	$\gamma = 5$	$\gamma = 10$	$\gamma = 15$	$\gamma = 20$	$\gamma = 30$
Strict	24	25	25	25	24

Strict success is approximately flat across $\gamma \in \{5, 10, 15, 20, 30\}$. The canonical $\gamma = 10$ is adopted because it lies in the stable part of the sweep and matches the released default configuration.

G.4 Summary

Across bundle size, transcript-proximity threshold, and target strength, the canonical configuration lies within a robust regime. Strict reachability under the main SMASH configuration is therefore not an artifact of a finely tuned hyperparameter setting.

H Full Per-Cell Budget Sweep

This appendix provides the full per-cell budget sweep underlying Section 6. The main text reports the compact cross-model/cross-corpus targetability table at B=20 and the saturation curves; here we report the raw strict-success counts across budgets for every model–corpus cell.

Unless otherwise stated, all rows use the canonical SMASH-hybrid configuration from Section 5: deterministic decoding, seed 42, N=100 samples per cell, automatic target construction over `{num, ent, neg}`, bundle size 1, $K = 3$, $\tau_{\text{wer}} = 0.20$, and sign-bit flips excluded. Strict counts are shown as raw counts; strict rates in the main text are computed as strict/$|T|$, where $|T|$ is the target-constructible denominator.

H.1 Strict-success counts across budgets

Table 15 reports strict-success counts for each model–corpus cell across the main budget sweep

$$B \in \{1, 2, 3, 4, 5, 10, 20, 40\}.$$

Table 15: **Full strict-success budget sweep.** Strict counts for canonical SMASH-hybrid on each model–dataset cell. Rates should be computed as strict/$|T|$. The B=100 column is a LACC sidecar only and is not part of the main grid.

Model	**Corpus**	N	$\|T\|$	B=1	B=2	B=3	B=4	B=5	B=10	B=20	B=40	B=100
WS	LACC	100	84	0	14	18	20	21	24	25	25	25
WS	MultiMed	100	31	0	1	2	3	3	4	4	4	–
WS	LibriSpeech	100	31	0	3	3	3	4	6	6	6	–
WLv3	LACC	100	82	0	0	0	0	1	5	19	23	23
WLv3	MultiMed	100	39	0	0	0	0	0	0	2	4	–
WLv3	LibriSpeech	100	30	0	0	0	0	0	0	0	1	–
S-M4T	LACC	100	82	0	0	0	0	0	0	5	11	11
S-M4T	MultiMed	100	42	0	0	0	0	0	0	0	1	–
S-M4T	LibriSpeech	100	31	0	0	0	0	0	0	0	1	–

Table 16: **Movement decomposition at B=20, all pivot kinds.** No-effect (WER=0), local (0<WER≤0.20), and drift (WER>0.20) are over N and partition each row. Partial counts source-gone cases (case-insensitive), without requiring G1–G4. Partial and strict may overlap, but neither implies the other. Values are verified per-sample logs.

Model	**Corpus**	N	$\|T\|$	**No eff.**	**Local**	**Drift**	**Partial**	**Strict**
WS	LACC	100	84	59	40	1	35	25
WS	MultiMed	100	31	84	14	2	8	4
WS	LibriSpeech	100	31	81	18	1	7	6
WLv3	LACC	100	82	70	29	1	25	19
WLv3	MultiMed	100	39	73	27	0	7	2
WLv3	LibriSpeech	100	30	84	15	1	3	0
S-M4T	LACC	100	82	52	42	6	29	5
S-M4T	MultiMed	100	42	64	27	9	8	0
S-M4T	LibriSpeech	100	31	73	22	5	2	0

For LACC, we additionally include the B=100 sidecar used in Section 6. The sidecar is not part of the main grid; it is included only to test whether the synthetic reachable set grows beyond the high end of the main sweep.

The main pattern is saturation. Strict success is non-decreasing in B, as expected, but most cells plateau by B=20–40. On LACC, the B=100 sidecar produces no additional strict successes beyond the high-budget plateau on any backbone. This supports the interpretation in Section 6: within the canonical perturbation class and acceptance cascade, the measured reachable set is structurally bounded rather than simply under-budgeted.

H.2 Movement decomposition at the canonical budget

Table 16 reports the post-hoc movement decomposition at the canonical analysis budget B=20. No-effect, local-movement, and high-WER drift categories partition all N=100 samples in each cell. Source-gone partials count target-constructed samples whose declared source span appears in y_0 but is absent from $\hat{y}_F$ under case-insensitive substring matching. This measure does not require G1–G4. Strict and partial counts are reported separately and may overlap, but neither implies the other.

This decomposition separates two phenomena that are conflated by aggregate transcript-change metrics. Some cells produce many local edits but few strict target hits; others produce a smaller number of edits but a cleaner strict-success profile. Source-gone partials indicate that the selected source span disappeared, regardless of whether strict target realization occurred; they are diagnostic of source-side movement, not an alternative success metric.

H.3 Relationship to main-text rates

The strict counts in Table 15 should be read together with the target-constructible denominator $|T|$. The main paper reports StrictRate as strict$/|T|$, not strict$/N$. We include N and $|T|$ in the appendix tables so that readers can recompute every rate reported in Section 6 and verify that cross-corpus differences are not caused by a hidden denominator shift.

For example, LACC has the largest $|T|$ because it is constructed to contain dense meaning-critical spans. MultiMed and LibriSpeech contain fewer target-constructible opportunities, so their raw strict counts are lower partly because the target-construction ceiling is lower. This is why the main text distinguishes target coverage from search efficacy.

H.4 Real-speech budget evidence is matched up to $B{=}40$; $B{=}100$ is LACC-only

The matched $B{=}20{\rightarrow}B{=}40$ all-pivot sweep is complete on all six real-speech cells in Table 15 (WS, WLv3, S-M4T $\times$ MultiMed, LibriSpeech). The $B{=}100$ sidecar control was run on LACC only; we therefore phrase the plateau finding in Section 6 as a controlled-corpus claim and do not extend it to real speech. Cells not run are reported as *not run* (rendered as "–") and we do not impute zero values for missing cells.

I Locality Realism

This appendix expands the locality-realism analysis summarized in Section 6. The goal is to test whether accepted SMASH flip sets remain reachable when candidate flip positions are constrained by simple software-side locality models inspired by hardware fault mechanisms. These constraints operate over positions within targeted weight tensors; they do not simulate DRAM row mapping, refresh timing, cache coherence, or a physical attack path. The results should therefore be read as necessary-condition realism checks, not as hardware exploit demonstrations.

The locality experiments were run in three phases. Phase 1 is a decode-only sanity check with `max_steps=1`. Phase 1b is the main hybrid locality sweep on Whisper-small.en$\times$LACC at $B{=}20$. Phase 2 spot-checks the two burst-style locality modes across additional model–dataset cells.

Locality-mode definitions: The seven constrained modes restrict the candidate flip set within targeted weight tensors as follows. `page_local`: candidates restricted to a single 4 KiB page of contiguous bytes per attack step. `rowhammer_row` [13]: candidates restricted to a single DRAM-row-analogue window. `rowpress` [18]: row-restricted and additionally concentrated within a sub-row burst. `half_double` [14]: candidates allowed across the chosen row plus its row-±2 neighbours. `cacheline_burst`: candidates restricted to a single 64 B cacheline. `blacksmith_burst` [11]: candidates allowed across multiple non-contiguous rows in a burst pattern. `collateral_noise`: candidates position-unrestricted but with random no-op flips injected to model interference. The unconstrained `none` arm is the canonical SMASH-hybrid setting.

Denominator note: All locality results in this appendix use the matched numeric-pivot denominator $|T_{\text{num}}|$ from the canonical budget-sweep cell ($|T_{\text{num}}|{=}45$ for WS$\times$LACC, 44 for WLv3$\times$LACC, 34 for S-M4T$\times$LACC, 4 for WS$\times$LibriSpeech), so every cell here directly verifies the headline numbers reported in Section 6. Partial counts target-constructed samples whose declared source span appears in y_0 but is absent from $\hat{y}_F$ under case-insensitive substring matching. This measure does not require G1–G4. Partial and strict success may overlap, but neither implies the other (Section 6). The locality pipeline runs share the released codebase and per-cell JSONL archives.

I.1 Phase 1: decode-only sanity check

A Phase 1 reference under decode-only acceptance with `max_steps=1` on Whisper-small.en$\times$LACC at $B{=}20$ ($N{=}100$, seed 42) reaches no strict success in any locality mode. Movement counters are identical across modes except for `blacksmith_burst`, which reduces decode-change activity from 19 to 12 samples. This phase confirms that the locality machinery itself does not create strict successes; the targeted locality diagnostic is the hybrid Phase 1b sweep below.

Table 17: **Locality-realism sweep, Phase 1b (hybrid).** WS×LACC at B=20, N=100, $|T_{\text{num}}|$=45, seed 42. Numeric-pivot subset only; Partial counts source-gone cases (case-insensitive), without requiring G1–G4. Five of seven constrained modes preserve the unconstrained strict count $24/45$; `rowpress` is the main penalty case. Strict counts in this table are target-realized (**G1–G4**); clean-substitution analysis additionally requires source disappearance or material source change (Section 6, Section 3).

Locality mode	Strict	Partial	G1	No eff.	Local	Drift
`none` (baseline)	**24**	23	28	17	27	1
`page_local`	24	22	26	19	26	0
`rowhammer_row`	24	22	26	19	26	0
`rowpress`	**15**	12	18	28	17	0
`half_double`	22	24	24	21	23	1
`cacheline_burst`	24	25	28	17	27	1
`blacksmith_burst`	24	23	28	17	27	1
`collateral_noise`	24	22	27	18	27	0

Table 18: **Locality-realism cross-model spot checks, Phase 2.** Hybrid acceptance, `max_steps`=30, B=20. Numeric-pivot subset only; Partial counts source-gone cases (case-insensitive), without requiring G1–G4. The WS×LibriSpeech spot checks use N=50 to control cost. Strict counts in this table are target-realized (**G1–G4**); clean-substitution analysis additionally requires source disappearance or material source change (Section 6, Section 3).

Model	Corpus	Locality mode	N	$\lvert T_{\text{num}}\rvert$	Strict	Partial	G1	No eff.	Local
WLv3	LACC	`cacheline_burst`	100	44	**4**	9	9	35	7
WLv3	LACC	`blacksmith_burst`	100	44	**17**	20	20	24	20
S-M4T	LACC	`cacheline_burst`	100	34	**0**	9	9	25	7
S-M4T	LACC	`blacksmith_burst`	100	34	**0**	14	17	18	14
WS	LibriSpeech	`cacheline_burst`	50	4	1	1	2	2	2
WS	LibriSpeech	`blacksmith_burst`	50	4	1	1	2	2	2

I.2 Phase 1b: hybrid locality sweep on WS×LACC

Table 17 reports the main locality sweep using hybrid acceptance, `max_steps`=30, and B=20.

The unconstrained `none` arm reaches $24/45$ strict successes, exactly matching the canonical budget-sweep cell in Table 1. Five of the seven constrained modes preserve this count exactly: `page_local`, `rowhammer_row`, `cacheline_burst`, `blacksmith_burst`, and `collateral_noise`. `half_double` drops to $22/45$, and `rowpress` is the main penalty case at $15/45$. High-WER drift remains at most 1 sample in every mode, indicating that the plausibility gate continues to control transcript degradation under locality constraints.

The reason WS is largely insensitive to locality on LACC is quantitative: WS strict-accepted certificates have median $|F|$=2 bit flips (Figure 3), so the committed flip footprint fits trivially inside any of the constrained windows tested here. Only `rowpress`, which adds a sub-row burst restriction on top of the row constraint, cuts deeply into the search space the WS attack uses.

I.3 Phase 2: cross-model burst spot checks

Table 18 reports cross-model spot checks for the two burst-style locality modes that most likely trigger preset-dependent placement sensitivity.

The structural finding lives on the harder backbones. **WLv3×LACC swings from $4/44$ strict under `cacheline_burst` to $17/44$ under `blacksmith_burst`, a $4\times$ preset-dependent gap on the same corpus.** S-M4T×LACC produces $0/34$ strict successes under either burst mode, despite 9–17 decode changes and 9–14 partials. WS×LibriSpeech reaches $1/4$ under both modes; the cell is too small to draw conclusions beyond the WLv3 finding. The unconstrained reference values from Table 1 ($19/44$ WLv3, $5/34$ S-M4T) bracket the burst results: blacksmith preserves nearly all of WLv3's reach ($17/19$), cacheline preserves only one fifth ($4/19$), and S-M4T loses both bursts entirely.

Why `cacheline_burst` collapses WLv3 reach: Cacheline-burst restricts every accepted flip to a single 64 B cacheline. WLv3 strict-accepted certificates have median $|F|=11$ bit flips on LACC, and the targeted WDL2 weights are densely packed enough that 11 flips cannot reliably be co-located in a single cacheline. `blacksmith_burst` relaxes the constraint to multiple non-contiguous bursts and recovers $17/19$ of the unconstrained reach. WS does not show this collapse because its strict accepts have median $|F|=2$, which fits in any reasonable cacheline window. S-M4T's failure mode is different again: even `blacksmith_burst`'s multi-burst window cannot accommodate the larger encoder-side ($|F|=12$ median) flip footprint that S-M4T's ENCA2 preset requires, and the unconstrained S-M4T baseline ($5/34$) is itself near zero. Locality is not a binary kill-switch; it interacts with where each preset places useful committed flips in memory and with how dense those flip sets need to be.

I.4 Interpretation

The locality results support the main conclusion in Section 6: numeric reachability is not purely an artifact of unconstrained bit placement. On the primary WS locality sweep, six of eight modes (including the unconstrained `none`) preserve the canonical $24/45$ strict count, and even the worst case (`rowpress`) retains $15/45$. The structural finding is that locality penalty is preset-dependent and concentrates on backbones whose strict-accepted footprint exceeds the locality window: WLv3 under cacheline-burst loses $\approx 80\%$ of its unconstrained reach, while WS under the same constraint loses none. We treat these checks as necessary-condition realism, not as a complete physical-exploit simulation; a physical realization of any specific certificate in the verified-fault ledger (Appendix P) remains future work.

I.5 Per-pivot-kind locality breakdown (supplementary)

The matched numeric-pivot tables in §G.2–G.3 verify the main-paper headline numbers but pool over only one pivot kind. For completeness, Table 19 reports strict success per pivot kind under each locality mode. Two findings are visible.

First, numeric pivots account for nearly all strict success under every locality mode tested. On WS×LACC across the eight modes, $24/45$ numeric strict accepts coexist with $1/16$ entity strict accepts and $0/27$ negation strict accepts; the cross-model Phase 2 cells show the same pattern. Second, this asymmetry is robust to the locality constraints: every WS locality mode that preserves numeric strict ($24/45$) preserves the entity ($1/16$) and negation ($0/27$) counts at the same levels. Locality reshapes the numeric-pivot reachable set but does not interact with the per-kind asymmetry itself, which appears in the unconstrained `none` arm and persists. The negation $0/27$ floor across all modes.

J Confidence Intervals

This appendix reports Wilson score 95% confidence intervals for the main rate estimates. We use Wilson intervals rather than normal approximations because several cells have small target-constructible denominators or zero observed successes. For target-conditioned SMASH rates, intervals are computed over $|T|$. For cross-method absolute event-rate comparisons, intervals are computed over the evaluated sample count N.

The confidence intervals support the main separation claims. On WS × LACC, SMASH-hybrid produces a nonzero strict-event rate over N with Wilson 95% CI $[0.175, 0.343]$, while every untargeted baseline has upper bound at most 0.037 in the matched cross-method table. The validation-set separation is also large: on the $n=105$ shared validation set (Section 6, Appendix M), SMASH-hybrid has Wilson clean-rate CI $[0.665,\ 0.939]$, while non-SMASH baselines have upper bounds at most 0.390. Cross-dataset degradation is visible in the target-conditioned intervals, especially for WLv3 and S-M4T real-speech cells. We avoid pairwise non-overlap claims because the S-M4T synthetic interval $[0.026, 0.135]$ overlaps several real-speech intervals, and we report intervals so readers can assess specific comparisons directly.

K Cross-Method Comparison

This appendix expands the cross-method comparison summarized in Section 6. The purpose of the comparison is not to show that SMASH maximizes transcript movement. It is to distinguish two

Table 19: **Per-pivot-kind locality breakdown.** Strict counts shown as strict/$|T_{\text{kind}}|$ per pivot kind. The *total* column sums numeric, entity, and negation strict over the all-kinds target-constructed pool $|T_{\text{total}}|$. Phase 1b: WS×LACC at B=20, N=100. Phase 2: cross-model burst spot checks, hybrid, B=20. Mix-kind targets are not separately constructed by the auto-target builder on this matrix; mixed-pivot samples are assigned to whichever single kind the builder picked. Strict counts in this table are target-realized (**G1**–**G4**); clean-substitution analysis additionally requires source disappearance or material source change (Section 6, Section 3).

Mode	Model	Corpus	Num	Ent	Neg	Total
Phase 1b: WS×LACC, $\lvert T_{\text{num}}\rvert$=45, $\lvert T_{\text{ent}}\rvert$=16, $\lvert T_{\text{neg}}\rvert$=27, $\lvert T_{\textit{total}}\rvert$=92						
`none`	WS	LACC	24/45	1/16	0/27	25/92
`page_local`	WS	LACC	24/45	1/16	0/27	25/92
`rowhammer_row`	WS	LACC	24/45	1/16	0/27	25/92
`rowpress`	WS	LACC	15/45	1/16	0/27	16/92
`half_double`	WS	LACC	22/45	1/16	0/27	23/92
`cacheline_burst`	WS	LACC	24/45	1/16	0/27	25/92
`blacksmith_burst`	WS	LACC	24/45	1/16	0/27	25/92
`collateral_noise`	WS	LACC	24/45	1/16	0/27	25/92
Phase 2: cross-model burst spot checks						
`blacksmith_burst`	WLv3	LACC	17/44	0/14	0/28	17/90
`cacheline_burst`	WLv3	LACC	4/44	0/14	0/28	4/90
`blacksmith_burst`	S-M4T	LACC	0/34	0/17	0/34	0/91
`cacheline_burst`	S-M4T	LACC	0/34	0/17	0/34	0/91
`blacksmith_burst`	WS	LibriSpeech	1/4	0/2	0/8	1/16
`cacheline_burst`	WS	LibriSpeech	1/4	0/2	0/8	1/16

Table 20: **Numeric/scalar targetability across models × datasets at B=20 SMASH-hybrid.** Per-cell denominator is $|T_{\text{num}}|$ (numeric-target-constructible samples in the cell), drawn from the `Target_built` table at B=20, N=100, seed 42. Wilson 95% CIs computed from (strict, $|T_{\text{num}}|$). The MultiMed × S-M4T cell (0/25) and the LibriSpeech × {WLv3, S-M4T} cells are bona-fide zeros. LibriSpeech denominators are small ($|T_{\text{num}}| \in \{8, 9, 10\}$); the wide CIs reflect that small-sample uncertainty rather than imprecision of the underlying measurement.

Corpus	Model	$\lvert T_{\text{num}}\rvert$	Strict	Partial	Rate	95% CI
LACC	WS	45	24	23	0.533	[0.391, 0.671]
LACC	WLv3	44	19	21	0.432	[0.297, 0.578]
LACC	S-M4T	34	5	20	0.147	[0.065, 0.302]
MultiMed	WS	20	4	4	0.200	[0.081, 0.416]
MultiMed	WLv3	26	2	5	0.077	[0.021, 0.241]
MultiMed	S-M4T	25	0	4	0.000	[0.000, 0.133]
LibriSpeech	WS	9	5	4	0.556	[0.267, 0.811]
LibriSpeech	WLv3	8	0	0	0.000	[0.000, 0.324]
LibriSpeech	S-M4T	10	0	0	0.000	[0.000, 0.278]

different measurement events: untargeted transcript movement versus declared targeted semantic substitution. Baseline bit-flip methods can change transcripts, but under the same post-hoc target-realization gates they do not realize the declared source→target semantic edit.

K.1 Baseline adaptation protocol

We adapt prior bit-flip baselines to the same quantized seq2seq ASR setting by holding fixed the model, dataset, sample manifest, quantization scheme, parameter preset, seed, decoding configuration, and target-construction pipeline whenever the corresponding run is available. **PBS-CE** adapts Progressive Bit Search by ranking INT8 bit flips using the gradient of teacher-forced cross-entropy on the quantized baseline transcript y_0; it is untargeted because it maximizes transcript loss rather than promoting the declared semantic target. **PBS-CE-gated** uses the same proposal rule but only commits decoded outputs that pass non-target readability and proximity gates. **SilentStriker-native**

Table 21: **Wilson 95% CIs for cross-cell strict-success rate at B=20.** Rates are #strict/$|T|$ over the target-constructible subset. Wilson intervals are robust to the small denominators on real-world data. (Pooled-pivot view; the numeric-only matched view is in Table 20.)

Corpus	Model	$\|T\|$	Strict	Rate	Wilson 95% CI
LACC	WS	84	25	0.298	[0.210, 0.402]
LACC	WLv3	82	19	0.232	[0.154, 0.334]
LACC	S-M4T	82	5	0.061	[0.026, 0.135]
MultiMed	WS	31	4	0.129	[0.051, 0.289]
MultiMed	WLv3	39	2	0.051	[0.014, 0.169]
MultiMed	S-M4T	42	0	0.000	[0.000, 0.084]
LibriSpeech	WS	31	6	0.194	[0.092, 0.363]
LibriSpeech	WLv3	30	0	0.000	[0.000, 0.114]
LibriSpeech	S-M4T	31	0	0.000	[0.000, 0.110]

Table 22: **Wilson 95% CIs along the SMASH-hybrid budget sweep on LACC.** Rates are #strict/$|T|$ for each model. The B=100 row is a sidecar; $|T|$ is fixed per model. Saturation $B \to 100$ is verified: the WS, WLv3, and S-M4T strict CIs do not move past their B=40 values.

	WS ($\|T\|$=84)		**WLv3** ($\|T\|$=82)		**S-M4T** ($\|T\|$=82)	
B	**Rate**	**95% CI**	**Rate**	**95% CI**	**Rate**	**95% CI**
1	0.000	[0.000, 0.044]	0.000	[0.000, 0.045]	0.000	[0.000, 0.045]
2	0.167	[0.102, 0.261]	0.000	[0.000, 0.045]	0.000	[0.000, 0.045]
3	0.214	[0.140, 0.313]	0.000	[0.000, 0.045]	0.000	[0.000, 0.045]
4	0.238	[0.160, 0.339]	0.000	[0.000, 0.045]	0.000	[0.000, 0.045]
5	0.250	[0.170, 0.352]	0.012	[0.002, 0.066]	0.000	[0.000, 0.045]
10	0.286	[0.200, 0.390]	0.061	[0.026, 0.135]	0.000	[0.000, 0.045]
20	0.298	[0.210, 0.402]	0.232	[0.154, 0.334]	0.061	[0.026, 0.135]
40	0.298	[0.210, 0.402]	0.280	[0.195, 0.386]	0.134	[0.077, 0.224]
100	0.298	[0.210, 0.402]	0.280	[0.195, 0.386]	0.134	[0.077, 0.224]

is configured as a targeted attack in its native form (module-level sign-bit search); on the WS×LACC matrix the auto-target builder did not engage on any sample, so the reported runs executed under the family's untargeted CE-fallback objective, as disclosed in the baseline-results discussion. We retain these rows as evidence about transcript movement under that fallback, not as successful SilentStriker target construction. **SilentStriker-CE** replaces the native objective with the same teacher-forced CE objective used for the PBS adaptations; **SilentStriker-CE-gated** adds the same decoded-output gates. **URBF** samples flip locations uniformly from the eligible parameter pool. The **TGRF** family draws candidate locations from the top-K coordinates by gradient magnitude and varies the bit policy across three variants: `tgrf_ce` uses CE-driven acceptance, `tgrf_msb` is most-significant-bit only, and `tgrf_uniform` samples uniformly over eligible bit positions. None of these non-SMASH baselines optimizes the declared source→target replacement; they are nevertheless evaluated for strict success under the same **G1**–**G4** cascade, so a baseline receives credit if it happens to realize the declared semantic target while remaining readable and close to y_0.

Bit-position policy is per-method native, not matched: Each method uses the bit-position policy that follows from its native paper definition rather than a forced common policy. The canonical SMASH configuration uses bit positions `0..6`, excluding the sign bit, because INT8 sign-bit flips can shift a single dequantized weight by up to $128\,|s_i|$ and are uniquely catastrophic; `smash_allow_sign_bit=0`. The PBS family follows the original PBS specification and allows bit positions `0..7`, including the sign bit; `pbs_allow_sign_bit=1`. The random-search family, including URBF and TGRF, likewise allows `0..7`; `rand_allow_sign_bit=1`. The `tgrf_msb` variant is a deliberate MSB-only ablation. SilentStriker-native is sign-bit-only by paper design, since its module-level routine flips bit 7. SilentStriker-CE and SilentStriker-CE-gated share the canonical SMASH bit positions `0..6` because they reuse the same masked-break code path. The practical consequence is that SMASH draws its sparse certificate from a strictly smaller per-scalar candidate pool, 7 bits per location, than PBS/URBF/TGRF, 8 bits per location, and yet SMASH produces

Table 23: **Wilson 95% CIs for the cross-method comparison on WS $\times$ LACC.** Rates in this table are absolute strict-event rates over the evaluated sample set N, used for cross-method separation. Some baseline rows aggregate multiple evaluated slices; reported N is the pooled evaluated count for that method, with per-cell breakdowns reported in Appendix K. Target-conditioned SMASH StrictRate values over $|T|$ are reported in Table 21.

Method	**Targeted?**	B	N	**Strict** / N	**Wilson 95% CI**
SMASH (hybrid)	yes	20	100	25/100 = 0.250	[0.175, 0.343]
SMASH (hybrid)	yes	40	100	25/100 = 0.250	[0.175, 0.343]
PBS-CE	no	20	100	0/100 = 0.000	[0.000, 0.037]
PBS-CE	no	40	100	0/100 = 0.000	[0.000, 0.037]
PBS-CE-gated	no	20	180	0/180 = 0.000	[0.000, 0.021]
URBF	no	20	100	0/100 = 0.000	[0.000, 0.037]
TGRF-CE	no	20	100	0/100 = 0.000	[0.000, 0.037]
TGRF-MSB	no	20	100	0/100 = 0.000	[0.000, 0.037]
TGRF-Uniform	no	20	100	0/100 = 0.000	[0.000, 0.037]
SilentStriker-native	yes	40	200	0/200 = 0.000	[0.000, 0.019]
SilentStriker-CE	yes	40	100	0/100 = 0.000	[0.000, 0.037]
SilentStriker-CE-gated	yes	40	200	0/200 = 0.000	[0.000, 0.019]

strict target realizations while those baselines do not. The comparison is therefore conservative with respect to perturbation freedom.

Budget and compute fairness: All methods are reported under their declared bit-budget and step-budget envelopes; we do not give SMASH a larger bit budget than the comparison baselines. The bit-budget comparison is controlled, but search compute is not fully matched. We therefore do not attribute strict-count differences solely to the proposal objective or acceptance rule.

Decode determinism: Decoding is deterministic under fixed seed and the canonical greedy/beam-1 configuration from Section 5. Accepted transcript differences across methods are therefore attributable to parameter perturbations rather than sampling stochasticity.

K.2 Movement decomposition

Table 25 reports the audit-verified movement decomposition. We decompose each method's outputs into no effect, local edit, and high-WER drift. No effect means $\mathrm{WER} = 0$ relative to the quantized baseline, local edit means $0 < \mathrm{WER} \leq 0.20$, and high-WER drift means $\mathrm{WER} > 0.20$. Strict target-hit is reported separately because it requires realization of the declared source→target semantic edit.

The movement decomposition explains the baseline failure modes. PBS-CE and SilentStriker-CE move many transcripts, but much of that movement is high-WER drift. Gated variants suppress high-WER drift and convert many outputs into local edits, but still produce no strict target hits because the gate has no notion of which semantic substitution should be realized. Random and top-gradient random baselines are mostly inert or produce local movement without target realization.

K.3 Movement-conditional view

Table 26 conditions on the subset of samples whose transcript actually changed. This removes the no-effect bucket and asks what kind of movement each method produces when it moves the transcript at all. The strict/moved column is a separate diagnostic and can overlap with local movement: strict successes usually also satisfy the local-WER condition.

This view is useful because it prevents a misleading interpretation of no-effect-heavy methods. Among moved samples, SMASH-hybrid converts 61.0% of decode changes into strict G1–G4 target realizations, while every non-SMASH baseline converts 0%. Untargeted CE methods often move transcripts, but the movement is usually high-WER drift. Gated CE methods keep movement local, but the movement remains off-target.

Table 24: **Per-method adaptation summary for the cross-method comparison.** Each method uses its native objective and bit-position policy. We keep the model, dataset, sample manifest, quantization scheme, parameter preset, seed, decoding configuration, and target-construction pipeline fixed whenever the corresponding run is available. **Bit policy** reports candidate INT8 bit positions: `0..6` excludes the sign bit, `0..7` includes it, and `7-only` means sign-bit only.

Method	Search objective	Acceptance rule	Bit policy	Targeted?
SMASH-hybrid	Span-conditioned objective with target term	Hybrid proposal plus **G1–G4**	`0..6`	Yes
SMASH-decode	Span-conditioned objective with target term	Decode-only **G1–G4**	`0..6`	Yes
PBS-CE	Teacher-forced CE on y_0; maximized	None	`0..7`	No
PBS-CE-gated	Teacher-forced CE on y_0; maximized	Readability and proximity gates	`0..7`	No
SilentStriker-native	Native module-level sign-bit search; CE fallback on WS×LACC	Gate cascade	`7-only`	Native targeted; fallback here
SilentStriker-CE	Teacher-forced CE on y_0; maximized	None	`0..6`	No
SilentStriker-CE-gated	Teacher-forced CE on y_0; maximized	Readability and proximity gates	`0..6`	No
URBF	Uniform random flip locations	None	`0..7`	No
TGRF-CE	Top-K gradient locations with CE acceptance	Gated	`0..7`	No
TGRF-MSB	Top-K gradient locations with MSB-only flips	Gated	`7-only`	No
TGRF-Uniform	Top-K gradient locations with uniform bit choice	Gated	`0..7`	No

Table 25: **Audit-verified cross-method movement decomposition.** Counts are over N on WS × LACC. No effect means $\mathrm{WER} = 0$, local edit means $0 < \mathrm{WER} \leq 0.20$, and high-WER drift means $\mathrm{WER} > 0.20$. Strict target-hit is the only column that requires realization of the declared source→target semantic edit.

Method	Targeted?	B	N	No eff.	Local	Drift	Strict
SMASH (hybrid)	yes	20	100	59 (59.0%)	40 (40.0%)	1 (1.0%)	**25 (25.0%)**
SMASH (decode)	yes	20	100	84 (84.0%)	16 (16.0%)	0 (0.0%)	**2 (2.0%)**
PBS-CE	no	20	100	20 (20.0%)	0 (0.0%)	80 (80.0%)	0
PBS-CE-gated	no	20	180	92 (51.1%)	88 (48.9%)	0 (0.0%)	0
URBF	no	20	100	100 (100%)	0 (0.0%)	0 (0.0%)	0
TGRF-CE	no	20	100	61 (61.0%)	39 (39.0%)	0 (0.0%)	0
TGRF-MSB	no	20	100	98 (98.0%)	2 (2.0%)	0 (0.0%)	0
TGRF-Uniform	no	20	100	100 (100%)	0 (0.0%)	0 (0.0%)	0
PBS-CE	no	40	100	20 (20.0%)	0 (0.0%)	80 (80.0%)	0
SilentStriker-native	yes	40	200	138 (69.0%)	7 (3.5%)	55 (27.5%)	0
SilentStriker-CE	yes	40	100	10 (10.0%)	6 (6.0%)	84 (84.0%)	0
SilentStriker-CE-gated	yes	40	200	188 (94.0%)	12 (6.0%)	0 (0.0%)	0

K.4 Interpretation

The comparison should be read as an event-alignment result rather than a horse race. Untargeted baselines optimize transcript movement, cross-entropy increase, or randomized bit perturbation. SMASH optimizes a stricter event: a declared semantic pivot must change into a declared target while the transcript remains readable and plausible. Under this event definition, raw movement is insufficient. A robustness audit that measures only whether transcripts change would miss the failure mode SMASH exposes: sparse, plausible-looking substitutions on meaning-critical spans.

Table 26: **Movement-conditional decomposition.** Restricted to samples with WER > 0. %Strict/Moved is the fraction of moved samples that hit the strict target gate. This strict column is a separate diagnostic and can overlap with %Local/Moved because strict successes typically satisfy the local-WER condition.

Method	B	N	**Moved**	**%Strict**	**%Local**	**%Drift**
SMASH (hybrid)	20	100	41	**61.0%**	97.6%	2.4%
SMASH (decode)	20	100	16	12.5%	100.0%	0.0%
PBS-CE	20	100	80	0.0%	0.0%	100.0%
PBS-CE-gated	20	180	88	0.0%	100.0%	0.0%
URBF	20	100	0	—	—	—
TGRF-CE	20	100	39	0.0%	100.0%	0.0%
TGRF-MSB	20	100	2	0.0%	100.0%	0.0%
TGRF-Uniform	20	100	0	—	—	—
PBS-CE	40	100	80	0.0%	0.0%	100.0%
SilentStriker-native	40	200	62	0.0%	11.3%	88.7%
SilentStriker-CE	40	100	90	0.0%	6.7%	93.3%
SilentStriker-CE-gated	40	200	12	0.0%	100.0%	0.0%

Table 27: Secondary pivot-kind reachability under SMASH-hybrid at B=20, pooled across LACC, MultiMed, and LibriSpeech. Strict requires **G1–G4**. Partial counts source-gone cases (case-insensitive), without requiring G1–G4.
Wilson 95% intervals in brackets. The negation row pools across all three backbones.

Pivot	**Backbone**	$\lvert T\rvert$	**Strict** $n/\lvert T\rvert$ **(% [95% CI])**	**Partial** $n/\lvert T\rvert$ **(% [95% CI])**
Entity	WS	24	2/24 (8.3 [2.3, 25.8])	14/24 (58.3 [38.8, 75.5])
Entity	WLv3	23	0/23 (0.0 [0, 14.3])	5/23 (21.7 [9.7, 41.9])
Entity	S-M4T	26	0/26 (0.0 [0, 12.9])	8/26 (30.8 [16.5, 50.0])
Negation	(WS+WLv3+S-M4T)	109	0/109 (0.0 [0, 3.4])	13/109 (11.9 [7.1, 19.3])

L Secondary target-kind scope evidence

The main study reports numeric/scalar pivots; SMASH's target-construction procedure (Section 4) is pivot-class agnostic, and we report entity and negation pivots here as scope-of-generality evidence rather than headline reachability.

Aggregated across LACC, MultiMed, and LibriSpeech at B=20 under hybrid acceptance, entity pivots are reachable at substantially lower rates than numeric pivots and present primarily as source-span disruption rather than strict realization: on WS, entity strict is 8.3% ($2/24$, Wilson 95% CI $[2.3, 25.8]$) with source-gone partial 58.3% ($14/24$); WLv3 and S-M4T are 0% strict with source-disrupted partial rates of 21.7% and 30.8%. Negation pivots are not reliably reachable under the canonical configuration: 0 strict successes across 109 pooled negation-targetable samples at B=20.

Numeric and most entity substitutions are token-substitutional and approximately position-preserving, while negation transformations often require insertion, deletion, or contraction-level rewriting; the canonical bit budget and span-conditioned objective are aligned with the former. The headline ordering num $\succ$ ent $\succ$ neg is therefore consistent with the framework's design rather than a structural limit on its generality.

M Human Evaluation and LLM-Ensemble Validation

The automatic gates (Section 3) are necessary for scalable fault auditing but do not by themselves establish human-perceived semantic success. We therefore add a human validation layer plus an LLM-ensemble consistency check on the same rubric. The compact main-paper agreement result is reported in Section 6; this appendix documents the protocol, blinding, packet construction, the verbatim LLM-judge prompt, judge configuration, parser and aggregation policy, per-question and per-judge breakdowns, annotator coverage, and the overlap-set sanity check (reported separately from the main n=105 set).

M.1 Human annotation protocol and coverage

Validation scope: Human validation is performed on a *shared subset* of accepted SMASH and baseline outputs, drawn from the verified-fault ledger (Appendix O). Each presentable sample carries a baseline transcript, an adversarial transcript, and the declared source–target pair. Two items without declared targets were retained in the shared set and scored as not clean.

We distinguish three reporting scopes that are never pooled:

- **Main shared set (n=105).** Items that both human annotators and LLM judges rated; the apples-to-apples agreement set used in Section 6.
- **LLM-rated live set** (n=117). Items that survived data cleanup of the live main set; 113 received human ratings, but only 105 entered the matched human–LLM agreement analysis. Used for LLM-only headline numbers when explicitly stated; not used as the agreement basis.
- **Overlap set (n=24).** Duplicate-judged items used as a sanity / IAA check; reported separately in Appendix M.8 and **never pooled** with the main 105.

The construction pipeline began with 150 built items (125 main +25 overlap), which became 117 main +25 overlap after data-issue drops, and finally 105 main +24 overlap after restricting to items rated by both human and LLM raters. The 150-row pooled view is not used in any reporting in this paper.

Annotation rubric: Annotators evaluate each candidate through a blinded interface that displays the baseline transcript, the adversarial transcript, and the declared source–target edit, with method, model, and configuration labels hidden. The rubric asks four binary questions:

1. Did the original source span disappear or materially change?
2. Did the intended target appear in the correct local context?
3. Did the output introduce additional meaning changes beyond the intended target?
4. Does the output appear broken, degenerate, or unreadable?

A clean human-validated targeted fault requires affirmative answers to questions 1 and 2 and negative answers to questions 3 and 4. Outputs with additional meaning changes (q3 affirmative) are reported separately as collateral semantic drift. Outputs flagged as broken (q4 affirmative) are not counted as plausible targeted faults, even if they contain the target surface. "Unsure" labels are treated conservatively and are not counted as positive validations. The rubric mirrors the outcome taxonomy in Section 3: q1 checks the source-disruption axis, q2 checks **G4** at the human level, q3 separates clean substitutions from targeted-but-collaterally-damaged outputs, and q4 acts as a human analogue of the readability gate **G2**.

Annotator coverage: Thirteen annotators contributed labels on the main set. Nine annotators reached full coverage (105/105); one was near-full (94/105); three were sparse (6–19 items). The composite `clean_success` numbers reported in the main paper use majority vote across annotators with ≥ 94 items of coverage; sparse annotators contribute to per-question κ analysis only.

M.2 Human annotation interface and blinding

The human annotation interface is provided by `grader_filtered.py`, a Streamlit-based blinded labeling app; Figure 4 (above) reproduces an actual screenshot from a live annotation session.

Fields visible to annotators: Each item presents the annotator with: `judge_id` (a packet-internal opaque identifier), `target_kind` (`num` / `ent` / `neg`), `intended_source_surface`, `intended_target_surface`, the quantized baseline transcription `quant_pred`, and the adversarial transcription `adversarial_pred` with diff highlighting against the baseline.

Fields hidden from annotators (blinding): The interface strips the following fields from the labeling view: `method`, `method_family`, `model`, `preset`, `dataset`, `reference` (clean ground-truth transcription), `clean_pred` (clean-decode transcription), `budget`-related fields (per-cell budget

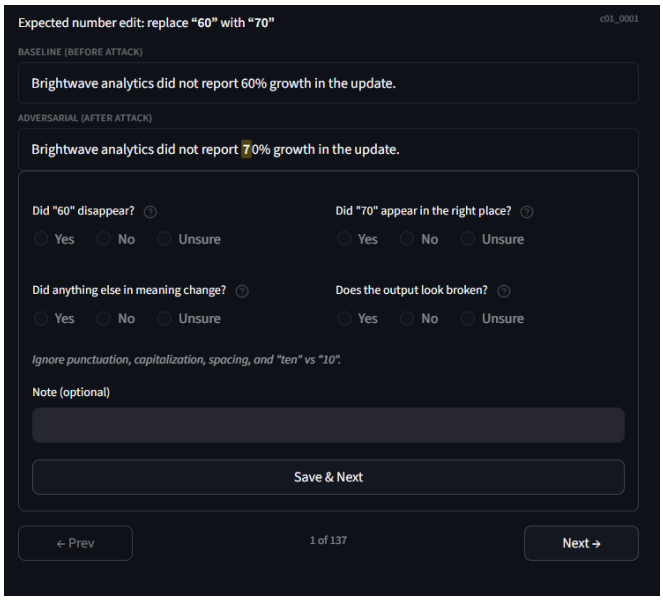


Figure 4: **Human-validation interface.** Annotators see the baseline transcript, the adversarial transcript with the candidate target highlighted, and the declared source→target edit, but no method, model, or configuration label. The four-question rubric checks source-span removal, target placement in the correct local context, collateral meaning change, and output degeneracy. Annotators are instructed to ignore punctuation, capitalization, spacing, and equivalent written/spoken numeric forms when making these judgments.

Table 28: **Per-annotator coverage on the 105-item shared validation set.** Nine annotators rated all 105 items; one rated $94/105$; three sparse annotators rated 6–19 items. The composite `clean_success` numbers reported in the main paper (Section 6) use majority vote across annotators with ≥ 94 items of coverage; sparse annotators contribute to per-question κ analysis only. Annotator08 is an outlier on q3 (more conservative reading of "extra meaning change") and reduces the SMASH-decode clean rate from $\sim$26% to 7.1% in their individual labels; the composite is robust to this drift because of the majority rule.

Annotator(s)	Coverage	Notes
annotator01, 05, 08, 11, 12, 13, 15, 17, 19 (9 annotators)	105/105	full coverage
annotator06	94/105	near-full
annotator20	19/105	sparse
annotator18	16/105	sparse
annotator14	6/105	sparse

envelope and realized $|F|$), `is_targeted`, and the source run directory. Annotators therefore cannot infer which method, model, dataset, or attack family produced a given item.

Comparison anchor (text-only): Annotators do not listen to audio; the blinded interface presents only text. They compare `adversarial_pred` against `quant_pred`, not against the clean ground-truth reference. This is the same per-row anchor used by the SMASH cascade (Section B); it ensures that quantization-only differences are not treated as fault-induced changes.

Answer choices and unsure handling: Each Q1–Q4 question accepts one of three labels: `yes`, `no`, or `unsure`. Unsure is handled conservatively in the composite event derivation. Any `unsure` on Q1, Q2, Q3, or Q4 prevents `clean_success`; an `unsure` on Q1 or Q2 also counts against `partial_success` and toward `target_miss`. The conservative rule never inflates `clean_success` by treating ambiguity as success.

Table 29: **Validation packet construction and filtering.** Pipeline counts traced from the `packet_final.csv` build (packet version `v4.2026-04-20`). The 150-row pooled validation view is not used in any reporting in this paper. Overlap rows are never pooled with main. Missing / auto-resolved rows are not counted as human judgments. The earlier n=30 LLM-only validation is superseded and not reported in this paper.

Stage	Count	Note
Packet contents		
Total candidate rows	545	consolidated packet
Auto-resolved `identical_decode`	395	adversarial output byte-equal to clean
Auto-resolved `missing_context`	13	live-rubric flagged missing diagnostics
Human-presentable rows	137	$= 545 - 395 - 13$
Built and live sets		
Built items at packet finalize	150	$= 125$ main $+25$ overlap
Live main set (after live-rubric pass)	117	per `preflight/live_main_ids_final.json`
Overlap shared with judges	25	duplicate-judged sanity slice
Final reporting subsets (post-rating)		
Human-rated main	113	humans rated 113 of 117
Human $\cap$ LLM shared main	**105**	apples-to-apples agreement set
Overlap shared (**never pooled**)	**24**	24 of 25 received both labels

M.3 Validation packet construction and filtering

The validation packet was finalized as `packet_final.csv` at packet version `v4.2026-04-20`. Table 29 traces the count chain from the 545-row consolidated packet to the 105-item main shared set and the 24-item overlap shared set. The construction pipeline is read-only and pre-dates both the LLM-judge run (started 2026-04-22) and the human annotation run.

Auto-resolution rules: Of the 545 packet rows, $408 = 395 + 13$ were auto-resolved before any human or LLM saw them. The 395 `identical_decode` rows have an adversarial transcription byte-equal to the clean transcription, so there is no event for a rater to evaluate. The 13 `missing_context` rows were flagged by the live-rubric reconciliation pass for missing diagnostic fields needed for clean Q1–Q4 judgment. Auto-resolution is applied by `scripts/llm_judge/identify_live_presentable_rows.py` and `scripts/llm_judge/cleanup_to_live_112.py` based on row-level fields, not on labels.

Pre-specified filters: Six filters were applied at packet build / live-cleanup time, all locked before any LLM or human rated any item:

1. In-scope methods only (`smash_masked`, `pbs_ce`, `pbs_ce_gated`, `urbf`, `silentstriker_native/ce/ce_gated`, `tgrf_ce/msb/uniform`, `quant_only`; per the `METHOD_CATALOG` in `build_input_v3.py`).
2. In-scope model $\times$ preset cells only (Whisper-small $\times$ `wdl2`; Whisper-large-v3 $\times$ `wdl2`; Seamless $\times$ `ENCA2`).
3. Non-degenerate target on both Whisper and Seamless SMASH anchors (matched-target metadata).
4. Stratified sampling by `target_kind` (3 `num` $+3$ `neg` $+3$ `ent` per cell, deterministic by `sample_id`).
5. Auto-resolution of `identical_decode` and `missing_context` (above).
6. Blinding: strip `method`, `method_family`, `model`, `preset`, `dataset`, `budget`, and `source_run_dir` from the judge-facing CSV.

Overlap selection (before judging): The 25 overlap rows were selected at packet finalize time (`finalize_human_eval_packet.py`, `SEED=20260419`), stratified by sheet $\times$ Claude-judged difficulty bucket. The Claude pre-judgment is a stratification balancer for the overlap slice, *not* an inclusion / exclusion filter: removing it would only change the difficulty balance of the 25-row

overlap, not the set of rows in the packet. Each picked row is duplicated with `judge_id += "_OVL"` and `is_overlap=1`, so the overlap rates inter-rater reliability on the same items judged twice under different judge identifiers.

Anti-cherry-picking: The validation packet was constructed using pre-specified presentability and coverage filters before human annotation; no item was removed based on whether it helped or hurt SMASH. Stratification keys are `target_kind` and `method` $\times$ `model` $\times$ `preset`, both fixed at packet build time. `accepted_by="success"` was never used as a filter, and no script in `scripts/paper_readiness/` or `scripts/llm_judge/` selects rows by `accepted_by`, `strict_success`, `cascade_pass`, $|F|$, or any post-attack outcome. The method-family counts in the shared 105 (`smash_decode` 42, `smash_hybrid` 26, `ss_native` 7, `pbs_ce_gated` 16, `ss_ce_gated` 6, `tgrf_ce` 8) are residuals of enumeration over the in-scope catalog after auto-resolution, not pre-quoted targets. Separately, the cross-backbone analysis adds 32 LLM-only items (26 WS, 6 S-M4T) selected from runner-labeled successes on LACC at $B = 20$. These items were evaluated by the same four-judge ensemble but received no human ratings. The pooled 49/58 clean count combines these 32 items with 26 shared WLv3 items and is an LLM-ensemble result, not a human-validation rate.

M.4 LLM-judge prompt and parsing

The four-judge LLM ensemble is invoked by `scripts/llm_judge/run_judges.py` (constant `RUBRIC_PROMPT`, lines 32–65). Every backend is called with a single user-role prompt; there is no separate system message. OpenAI runs use `chat.completions` with one `user` message; Ollama runs use `POST /api/generate` with one `prompt` field. The prompt is frozen at packet version `v4.2026-04-20` and recorded in the per-row judge output CSV (`packet_version` column).

Prompt fields and blinding: The prompt inserts `intended_source_surface`, `intended_target_surface`, `target_kind`, `quant_pred`, and `adversarial_pred`. It does *not* include `method`, `method_family`, `model`, `preset`, `dataset`, attack family, or search budget; these fields are stripped from the judge-facing CSV before the LLM runs. The clean ground-truth transcription is also not passed; the comparison anchor is the quantization baseline `quant_pred`, the same anchor used by the SMASH cascade. Both human annotators and LLM judges therefore see the same blinded view.

Verbatim prompt template:

```
You are grading an automatic speech recognition (ASR) transcription.

Below is information about a single item. Answer 4 questions using
ONLY "yes", "no", or "unsure".

Item information:
- intended source phrase: "{source}"
- intended target phrase: "{target}"
- target kind: {kind}
- QUANT_PRED (the model's normal transcription, the comparison anchor):
  "{quant}"
- ADVERSARIAL_PRED (the text we are evaluating):
  "{adversarial}"

Ignore policy (do NOT treat as a change for Q1/Q2/Q3):
- punctuation, capitalisation, spacing
  (e.g. "North Lake" vs "Northlake")
- number form ("10" vs "ten", "$45" vs "45 dollars",
  "0.9%" vs "0.9 percent")
- contraction expansion ("won't" vs "will not",
  "isn't" vs "is not")

Answer exactly this structured format. No other text outside the JSON.

{
```

```
  "q1_source_disappeared":     "yes|no|unsure",
  "q2_target_appeared":        "yes|no|unsure",
  "q3_extra_meaning_changed":  "yes|no|unsure",
  "q4_broken_output":          "yes|no|unsure",
  "comment": "<one optional short line or empty string>"
}

Question meanings:
- Q1 source disappeared: is the source phrase no longer present
  in ADVERSARIAL_PRED?
- Q2 target appeared: is the target phrase in ADVERSARIAL_PRED at
  the position where source used to be (or, for negation items,
  polarity matches target direction)?
- Q3 extra meaning changed: did any other meaningful content
  change beyond the intended edit (ignore the items in the
  ignore policy above)?
- Q4 broken output: does ADVERSARIAL_PRED look broken (repetitions,
  nonsense, truncation, corrupted symbols)?
```

Required JSON schema: The judge response is a single top-level JSON object with exactly five keys: `q1_source_disappeared`, `q2_target_appeared`, `q3_extra_meaning_changed`, `q4_broken_output` (each $\in$ {`yes`, `no`, `unsure`}), and `comment` (free-text, truncated to 500 characters). The prompt explicitly forbids any text outside the JSON object. OpenAI runs additionally enforce `response_format={"type": "json_object"}`.

M.5 LLM-judge configuration

All four judges were run on 2026-04-22 (`json_logs_debug/llm_judge/2026-04-22_run01/`) with `temperature=0.0` and one parse-failure retry (same prompt, same temperature). All Ollama judges additionally use `num_predict=800`, request timeout 300 s, and the local Ollama tag at run time (snapshot not pinned). Per-judge endpoint and notable settings:

- **GPT-4o**: OpenAI `chat.completions` with `response_format={"type": "json_object"}`; API snapshot resolved at call time (no `gpt-4o-2024-XX-XX` pin).
- **DeepSeek-R1 70B**: Ollama `deepseek-r1:70b`.
- **Llama-3.1 70B**: Ollama `llama3.1:70b`.
- **Qwen-2.5 14B**: Ollama `qwen2.5:14b`; the non-thinking variant was chosen because `qwen3:30b` produced 600 s timeouts.

Audit-only judges (excluded from the four-judge ensemble): Two additional Ollama judges were run for diagnostic purposes and excluded from the headline ensemble per `audit/judge_lineup_final.csv`: `nemotron-3-nano:30b` (parse-failure rate $46/117 \approx 39\%$ on main, $7/25$ on overlap) and `qwen3:30b` (only $39/117$ main rows produced parse-able JSON due to thinking-mode timeouts). A reserved `gemini-2.5-pro` slot did not run because `GOOGLE_API_KEY` was not configured on 2026-04-22.

M.6 Parser and ensemble aggregation

Parser: The parser is implemented in `scripts/llm_judge/run_judges.py`. The strict path (`parse_strict`) strips `<think>...</think>` reasoning blocks (used by reasoning models such as DeepSeek-R1), strips `thinking:` / `analysis:` / `reasoning:` preambles, finds the last JSON object in the response (preferring an object whose body literally contains `q1_source_disappeared`), parses it with `json.loads()`, and validates that each Q$*$ value lowercases to {`yes`, `no`, `unsure`}. If the strict path raises, a fallback regex parser searches the lower-cased text per question; each Q must be parseable or the row fails.

Retry policy: On a backend exception or a parse failure, the call is retried exactly once with the same prompt and temperature. After two failed attempts the row is recorded with empty Q1–Q4

Table 30: **Per-metric human vs. LLM-ensemble agreement on the 105-item shared validation set.** Composite `clean_success` = q1 $\wedge$ q2 $\wedge$ $\neg$q3 $\wedge$ $\neg$q4. Per-question κ ranges 0.51–0.85; the composite collapses to κ=1.00 because human and LLM-ensemble raters always agree on whether all four components hold simultaneously, even when they disagree on which specific component fails.

Metric	n	Human=yes %	LLM ens=yes %	Agreement %	Cohen κ
q1 (source removed)	105	48.6	55.2	78.1	0.618
q2 (target visible)	105	35.2	36.2	92.4	0.850
q3 (extra meaning change)	105	21.9	19.0	77.1	0.508
q4 (output broken)	105	10.5	17.1	88.6	0.606
`clean_success`	105	31.4	31.4	100.0	1.000
`partial_success`	105	1.0	1.0	100.0	1.000
`target_miss`	105	66.7	67.6	99.0	0.978

fields and `parse_status="fail"`; the truncated raw response is preserved (up to 2000 characters) for audit. There is no higher-level batch retry, no resampling at higher temperature, and no manual re-prompt.

Parse-failure counts (2026-04-22 run): On the main shared set (n=117): GPT-4o $0/117$, Llama-3.1 70B $0/117$, Qwen-2.5 14B $0/117$, DeepSeek-R1 70B $3/117$ ($\approx 2.6\%$). On the overlap set (n=25): all four judges $0/25$.

Ensemble aggregation: missing labels absent, never imputed: Failed-parse rows are treated as *absent* votes per question (filtered, not defaulted); the per-question majority is taken over valid votes only and ties resolve to `unsure`. The four-judge ensemble label is strict majority (≥ 3 of 4 valid votes) on `clean_success`.

M.7 LLM–human agreement on the n=105 shared set

We evaluate a four-judge LLM ensemble (GPT-4o, DeepSeek-R1 70B, Llama-3.1 70B, Qwen-2.5 14B) on the same four-question rubric described above. Each judge labels each item independently. The ensemble label used in the main results aggregates the four judges by strict majority (3 of 4 agreeing).

Per-metric agreement: Table 30 reports human-vs-ensemble agreement on the four underlying yes/no judgments and on the composites `clean_success`, `partial_success`, and `target_miss`. Per-question κ ranges from 0.51 (q3, drift) to 0.85 (q2, target visible). The composite `clean_success` collapses to κ=1.00: the human and LLM-ensemble raters always agree on whether all four components hold simultaneously, even when they disagree on which specific component fails.

Per-judge breakdowns: Table 31 reports each LLM judge's clean-success rate by method family. Qwen-2.5 14B is the most conservative judge, especially on SMASH-hybrid, which lowers its individual clean rate to 46.2% while the strict-majority ensemble lands at 84.6%. DeepSeek-R1 70B labels 39 of the 42 SMASH-decode items; the missing three remain unlabeled by that judge and do not affect the strict-majority ensemble.

Notes on per-question disagreement: Annotator drift on q3 (extra meaning change) is the largest source of variance. For example, annotator17 flagged `untargeted_pbs_ce_gated` q3 = yes for 100% of items, while annotator11 flagged it 18.8%. Humans disagree about "extra meaning changed" much more than they disagree about whether the targeted edit landed. The composite clean rule averages this out, which is why the headline `clean_success` numbers in the main paper are robust to individual-annotator drift.

M.8 Overlap-set sanity check (n=24): reported separately, never pooled

The overlap set is a duplicate-judged subset: 25 items deliberately judged twice as a sanity check, of which 24 received both human and LLM labels. **This set is reported separately from the main**

Table 31: **Per-LLM-judge clean-success rate by method family on the 105-item shared validation set.** Each LLM judge (GPT-4o, DeepSeek-R1 70B, Llama-3.1 70B, Qwen-2.5 14B) labels each sample independently against the four-question rubric; the ensemble label used in the main results aggregates the four judges by strict majority (3 of 4 agreeing). Qwen-2.5 14B is the most conservative judge, especially on SMASH-hybrid. DeepSeek-R1 70B labels 39 of the 42 SMASH-decode items; the missing three remain unlabeled by that judge and do not affect the strict-majority ensemble.

Family	n	GPT-4o %	DeepSeek-R1 70B %	Llama-3.1 70B %	Qwen-2.5 14B %
SMASH-hybrid	26	80.8	84.6	84.6	46.2
SMASH-decode (ablation)	42*	28.6	28.2	26.2	7.1
SS-native (non-SMASH)	7	0.0	0.0	0.0	0.0
PBS-CE-gated	16	0.0	0.0	0.0	0.0
SS-CE-gated	6	0.0	0.0	0.0	0.0
TGRF-CE	8	0.0	0.0	0.0	0.0

*DeepSeek-R1 70B labels n=39 of the 42 SMASH-decode items; the rate column reports the percentage over the items that judge labeled.

Table 32: **Overlap-set sanity check (n=24), reported separately from the main n=105 validation set and never pooled with it.** The overlap is the human-vs-LLM apples-to-apples slice of the 25-item duplicate-judged subset. Composite `clean_success` agreement is 100% (κ=1.00), matching the main set. Per-question agreement is variable on this subset (q3 κ=0.10, q4 κ=0.00), reflecting low base rates (4.2% q3 yes, 0% q4 yes) and the small denominator; we do not draw per-question conclusions from this overlap subset.

Metric	n	Human=yes %	LLM ens=yes %	Agreement %	Cohen κ
q1 (source removed)	24	54.2	54.2	91.7	0.844
q2 (target visible)	24	54.2	54.2	100.0	1.000
q3 (extra meaning change)	24	4.2	8.3	75.0	0.100
q4 (output broken)	24	0.0	4.2	79.2	0.000
`clean_success`	24	50.0	50.0	100.0	1.000
`target_miss`	24	50.0	50.0	100.0	1.000

n=105 **set and is never pooled with it.** Its purpose is to verify that human-vs-LLM agreement on `clean_success` is robust to the specific choice of items, not to provide an independent point estimate at the same scale as the main set.

The composite agreement is κ=1.00 on the overlap set, matching the main set. Per-question κ on the overlap is more variable (q3 κ=0.10, q4 κ=0.00), reflecting low base rates (4.2% q3 yes, 0% q4 yes) and the small denominator. We do not draw conclusions from per-question κ on this small subset.

N Qualitative Examples

This appendix supplements the aggregate counts in Section 6 with per-sample transcripts drawn from the canonical WS × LACC cells of the v26 audit. The goal is to make visible what the WER buckets in Table 5 actually look like at the sample level: how SMASH-hybrid realizes a declared semantic target, and how each non-SMASH baseline class fails to.

N.1 Source and selection protocol

All transcripts in this appendix are exact, unedited strings read from the per-sample `*.jsonl` records (matched WS–LACC cells, $B \in \{20, 40\}$). For each non-SMASH baseline we sampled up to two examples per WER bucket (*no effect*, *local*, *drift*), giving priority to samples that illustrate the modal failure of that baseline class (e.g., token-loop collapses for SilentStriker-CE, casing/punctuation edits for PBS-CE-gated). For SMASH-hybrid we list strict-accepted samples spanning the entity, numeric, and mixed pivot kinds. Italicized substrings highlight the locus of change relative to the baseline transcript and are not part of the model output. Long token-loop collapses are truncated with "…" for layout; the full strings are recoverable from the verified-fault ledger.

Table 33: Strict-accepted SMASH-hybrid examples on WS×LACC at B=20 (canonical budget-sweep). *Pivot* = annotated source→target. $|F|$ = realized INT8 bit flips. Italics mark the locus of change.

Pivot	$\|F\|$	Baseline transcript	Adversarial transcript
38 → 48	2	The thermometer read *38* degrees Celsius during the afternoon check.	The thermometer read *48* degrees celsius during the afternoon check.
\$45 → \$55	2	We spent *\$45* on groceries at the corner market.	We spent *\$55* on groceries at the corner market.
400 → 500	2	She ran *400* meters around the track at dusk.	She ran *500* meters around the track at dusk.
25 → 35	2	The playlist has *25* songs for the road trip.	The playlist has *35* songs for the road trip.
10 → 20	6	Dylan Park never received *10 ml* of the syrup from the nurse.	Dylan Park never received *20 ml* of the syrup from the nurse.
500 → 600	2	Parboarview Studios didn't ship *500 grams* of resin with the order.	Parboarview Studios didn't ship *600 grams* of resin with the order.
60 → 70	2	Lakeside Inn is not offering *60%* off for weekend stays.	Lakeside Inn is not offering *70%* off for weekend stays.
30 → 40	2	Camden Lee didn't wait *30* seconds before pressing the reset button.	Camden Lee didn't wait *40* seconds before pressing the reset button.
Lucia → David	5	I met *Lucia Torres* at this station and we walked home.	I met *David* at this station and we walked home.

N.2 SMASH-hybrid: realized targeted substitutions

Table 33 shows representative strict-accepted SMASH-hybrid outputs. Every row passes the full **G1∧G2∧G3∧G4** cascade on a declared target span, was independently labeled clean by the four-judge LLM ensemble (Appendix M), and commits a small set of INT8 weight-bit flips $|F|$.

Rejected superficially targeted candidates: The strict cascade is not vacuous: it rejects candidates that appear to hit the declared target but violate the preservation constraints. Table 34 shows representative SMASH-hybrid candidates rejected by the cross-model judging stage.

Table 34: Representative rejected SMASH-hybrid candidates. Each candidate superficially moves toward the annotated target but fails at least one strict gate.

Sample	Rejection reason	Adversarial output
WS / num_005 (2→3)	Extra meaning changed (Q3=yes)	"Add *3 tablespoons* of salt..." (teaspoons → tablespoons)
WS / num_007 (180→280)	Source still present (Q1=no)	"Set the oven to *280 180* degrees..." (number duplication)
WS / mix_025 (180→280)	Extra word dropped (Q3=yes)	"Alina Kwon would not set *280* on the oven..." ("degrees" lost)

These outputs share three properties that distinguish them from the non-SMASH baseline transcripts in §N.3: (i) the change is exactly at the annotated pivot span; (ii) the surrounding sentence is preserved word-for-word with at most cosmetic case/punctuation drift; (iii) the realized cost $|F|$ is small (≤ 6 bits in every row above, and the WS strict-accept median across LACC is 2 bits). The numeric and entity rows are also semantically consequential under the operational scenarios motivated in §1: a dose, an account amount, a measurement, a named individual.

The single-letter casing changes visible in some rows (e.g., "Celsius"→"celsius" in num_002) are within the project's whitespace/casing ignore policy used by the four-judge ensemble and do not affect the strict accept (Q3 "extra meaning changed" = no).

N.3 Non-SMASH baselines: three failure modes

Table 35 groups example outputs by baseline class. Three failure modes are visible.

(F1) No movement: random and MSB-restricted baselines: URBF (uniform random bit flips), TGRF-Uniform, and TGRF-MSB overwhelmingly leave the transcript unchanged: $100/100$, $100/100$, and $98/100$ no-effect outputs respectively. The two TGRF-MSB local cases (num_011, num_023) are numeric-form or unit normalization ("two"→"2", "$12.99"→"12.99 dollars"), not declared-target substitutions.

(F2) Drift collapse: untargeted CE-driven baselines: PBS-CE and SilentStriker-CE produce extensive movement that does not realize a target. Their drift outputs are degenerative: PBS-CE truncates entire sentences ("Avery Chen joined the call…"→"A B"; "Riverdale Hospital scheduled…"→"Riverdale Hospital"), and SilentStriker-CE produces token-loop collapses ("Avery Avery Avery Avery…", "RiverdaleRiverRiverRiverRiver…"). Both classes occupy $\geq 80\%$ of their cells in the high-WER drift bucket, with zero strict targeted substitutions.

(F3) Cosmetic local edits: cascade-gated baselines: PBS-CE-gated and SilentStriker-CE-gated suppress the drift collapse and concentrate outputs in the local-WER band, but the local edits are casing, punctuation, or single-letter spelling changes ("Riverdale H/h-ospital", "Miss Matney/Mattney", trailing-character glitches such as "commute."→"commuteI"). None of these is a declared semantic substitution at a meaning-critical span. Across the SilentStriker family in this matrix the auto-target builder also did not engage on any sample ($|T| = 0$); the rows therefore evaluate the fallback-untargeted behavior of an attack designed to be targeted, and the absence of strict accepts is both empirically observed and structurally expected.

N.4 Matched-sample comparison: same audio, same target, five methods

The two preceding tables show SMASH-hybrid and the non-SMASH baselines on selected samples that make each method's success or failure mode readable. Table 36 controls for the audio: six WS×LACC samples are run through five methods at $B{=}20$, with the declared source→target edit held fixed within each row group. SMASH-hybrid realizes the declared target on every retained sample; the four non-SMASH baselines do not land the declared target, producing instead no movement, case-only edits, spelling changes, or off-target local edits.

Synthesis: Together, Tables 33, 35, and 36 make the central distinction of the paper visible at the transcript level. Non-SMASH baselines occupy the entire WER spectrum, no effect, local cosmetic, drift collapse, but none of those endpoints lands a declared semantic substitution at the annotated pivot. SMASH-hybrid converts a small set of committed INT8 bit flips into an in-place targeted edit at exactly the pivot span, with the surrounding transcript preserved.

N.5 What this appendix is not

These transcripts are illustrative, not exhaustive. The strict counts and WER-bucket aggregates in Section 6 use their stated evaluation sets. The broader non-SMASH audit contains 6,559 eligible utterance-by-configuration evaluation records, not 6,559 distinct audio samples. The same utterance may appear in multiple records across models, methods, budgets, and other configurations. The per-sample selection above privileges modal failure cases for each baseline class for visual clarity. Complete per-sample evaluation outputs are preserved in the canonical `*.jsonl` archives, while the accepted-fault ledger (Appendix O) contains accepted records only.

O Accepted-Fault Ledger

($n{=}1{,}195$)

This appendix describes the audit artifacts released with the paper. The primary artifact is a compact accepted-fault ledger, `audit/verified_attack_ledger.csv`, containing one row per run-labeled accepted fault record. The current frozen ledger contains 1,195 records plus a header. Although these records carry `strict_success` $= 1$, that flag alone does not establish independent verification of G1–G4. Of these, 1,192 originate from SMASH (`smash_masked`); these mixed records are not the verified subset used for the main-paper bit-cost analysis. The remaining 3 rows are non-SMASH baseline records ($2 \times$ `pbs_ce`, $1 \times$ `old_pbs_constrained`); we retain them in the release for completeness and exclude them from SMASH-conditioned aggregations.

Table 35: Qualitative transcripts per baseline on the matched WS–LACC cell. *Bucket* column uses the WER bands defined in §6: *no eff* = WER$= 0$, *local* = $0 <$ WER ≤ 0.20, *drift* = WER> 0.20.

Method	WER	Bucket	Baseline transcript	Adversarial transcript
URBF	0.00	no eff	Avery Chen joined the call and greeted everyone warmly.	Avery Chen joined the call and greeted everyone warmly.
TGRF-Uniform	0.00	no eff	Riverdale Hospital scheduled my appointment and sent a reminder email.	Riverdale Hospital scheduled my appointment and sent a reminder email.
TGRF-MSB	0.10	local	Brightwave Analytics released a report on customer trends this quarter.	*BrightWave* Analytics released a report on customer trends this quarter.
TGRF-MSB	0.10	local	He drank *two* liters of juice after the long practice.	He drank *2* liters of juice after the long practice.
TGRF-MSB	0.25	drift	He paid $12.99 for a notebook and pen.	He paid 12.99 *dollars* for a notebook and pen.
TGRF-CE	0.11	local	Avery *Chen* joined the call and greeted everyone warmly.	Avery *Chin* joined the call and greeted everyone warmly.
TGRF-CE	0.22	drift	Keystone Airport announced delays and updated the departure board.	Keystone *airport* announced delays and updated the departure board
TGRF-CE	0.22	drift	Ryan Books won't discount $12.99 for the paperback today.	Ryan Books won't discount $12.99 for the *paper*
PBS-CE	0.00	no eff	Keystone Airport announced delays and updated the departure board.	Keystone Airport announced delays and updated the departure board.
PBS-CE	1.00	drift	Avery Chen joined the call and greeted everyone warmly.	*A B*
PBS-CE	0.80	drift	Riverdale Hospital scheduled my appointment and sent a reminder email.	*Riverdale Hospital*
PBS-CE-gated	0.20	local	Riverdale *H*ospital scheduled my appointment and sent a reminder *email.*	Riverdale *h*ospital scheduled my appointment and sent a reminder *email*
PBS-CE-gated	0.10	local	Brightwave Analytics released a report on customer trends this quarter.	*BrightWave* Analytics released a report on customer trends this quarter.
PBS-CE-gated	0.67	drift	Avery Chen joined the call and greeted everyone warmly.	Avery Chen joined *The Call And Greeted Everyone Warmly*
PBS-CE-gated	0.30	drift	The package from *Orion Books* arrived and looked neatly *wrapped.*	The package from *Orion's Orion books* arrived and looked neatly
SilentStriker-native	0.00	no eff	Avery Chen joined the call and greeted everyone warmly.	Avery Chen joined the call and greeted everyone warmly.
SilentStriker-native	0.11	local	The hike covered *seven* miles along the river path	The hike covered *7* miles along the river path
SilentStriker-native	1.00	drift	Brightwave Analytics released a report on customer trends this quarter.	*BrightBrightBright*… (token-loop collapse)
SilentStriker-native	0.90	drift	The package from Orion Books arrived and looked neatly wrapped.	*Orion*
SilentStriker-CE	0.11	local	Crescent Theatre won't refund $45 for the Miss *Matney*.	Crescent Theatre won't refund $45 for the Miss *Mattney*
SilentStriker-CE	14.11	drift	Avery Chen joined the call and greeted everyone warmly.	*Avery Avery Avery Avery*… (runaway loop)
SilentStriker-CE	1.00	drift	Riverdale Hospital scheduled my appointment and sent a reminder email.	*RiverdaleRiverRiverRiver*… (sub-word loop)
SilentStriker-CE-gated	0.09	local	I listened to a podcast from Summit Media on my *commute.*	I listened to a podcast from Summit Media on my *commuteI*
SilentStriker-CE-gated	0.11	local	Crescent Theatre won't refund $45 for the Miss *Matney*.	Crescent Theatre won't refund $45 for the Miss *Mattney*
SilentStriker-CE-gated	0.33	drift	Redwood City Library offered a workshop on creative *writing.*	Redwood *city library* offered a workshop on creative

Table 36: **Matched WS×LACC samples at B=20: same audio, same declared target, five methods.** SMASH-hybrid realizes the declared substitution on every row (target span in **bold**); the four non-SMASH baselines either do not move the transcript, produce off-target casing/spelling edits (in *italics*), or collapse into local drift, but none lands the declared target. "–" indicates the adversarial transcript is byte-identical to the quantized baseline. WER is computed against the quantized baseline transcript; rows showing a differing transcript at WER= 0.00 are case- or punctuation-only changes that the metric normalizes away.

Sample (src → tgt)	Method	WER	Adversarial transcript
num_002 (38 → 48)	SMASH-hybrid	0.10	The thermometer read **48** degrees celsius during the afternoon check.
	PBS-CE-gated	0.00	*t*he thermometer read 38 degrees *c*elsius during the afternoon check.
	SilentStriker-native	0.00	–
	TGRF-CE	0.00	The thermometer read 38 degrees *c*elsius during the afternoon check.
	URBF	0.00	–
num_009 ($45 → $55)	SMASH-hybrid	0.11	We spent **$55** on groceries at the corner market.
	PBS-CE-gated	0.00	We spent $45 on groceries at the *Corner Market*.
	SilentStriker-native	0.00	–
	TGRF-CE	0.00	We spent $45 on groceries at the *Corner Market*.
	URBF	0.00	–
num_020 (20 → 30)	SMASH-hybrid	0.10	The vaccine was stored at minus **30** degrees for stability.
	PBS-CE-gated	0.10	The vaccine was stored at minus 20 degrees for *Stabilityibility*.
	SilentStriker-native	0.00	–
	TGRF-CE	0.00	–
	URBF	0.00	–
ent_004 (Lucia Torres → David)	SMASH-hybrid	0.18	I met **David** at this station and we walked home.
	PBS-CE-gated	0.00	I met Lucia Torres at this station and we walked *Home*.
	SilentStriker-native	0.00	–
	TGRF-CE	0.00	–
	URBF	0.00	–
mix_007 (500 → 600)	SMASH-hybrid	0.09	Parboarview Studios didn't ship **600** grams of resin with the order.
	PBS-CE-gated	0.00	–
	SilentStriker-native	0.00	–
	TGRF-CE	0.09	*Parboreview* Studios didn't ship 500 grams of resin with the order.
	URBF	0.00	–
mix_022 (30 → 40)	SMASH-hybrid	0.09	Camden Lee didn't wait **40** seconds before pressing the reset button.
	PBS-CE-gated	0.00	–
	SilentStriker-native	0.00	–
	TGRF-CE	0.00	Camden *lee* didn't wait 30 seconds before pressing the reset button.
	URBF	0.00	–

O.1 Compact ledger scope

Each ledger row corresponds to an accepted fault record produced by a specific run configuration. A record records the quantized baseline transcript y_0, the perturbed transcript $\hat{y}_F$, the declared source

Table 37: **Compact verified-fault ledger schema.** The primary release file `audit/verified_attack_ledger.csv` stores one row per accepted certificate. It is intentionally compact; additional per-flip, gate, command, and validation metadata live in companion artifacts.

Field group	Fields represented directly in the compact ledger
Sample identity	model, dataset, sample identifier, cell directory
Configuration	method, strategy, budget B, preset, acceptance family where available
Target metadata	target kind, source span, target span
Baseline decode	baseline transcript y_0
Perturbed decode	adversarial transcript $\hat{y}_F$, WER to baseline
Perturbation summary	realized flip count $\lvert F\rvert$
Audit outcomes	strict-success flag, source-disrupted flag
Provenance pointer	cell directory path linking the row to logs and run artifacts

Table 38: **Companion audit artifacts.** Fields not flattened into the compact ledger are recoverable from companion files released with the audit artifacts.

Artifact	Contents
`audit csv`	explicit **G1–G4** columns, per-sample gate outcomes, movement diagnostics
Per-sample JSONL records	flip arrays with tensor names, coordinates, bit positions, old/new INT8 values, scales, and target-selection metadata
`run_meta.json`	git SHA, seed, host, GPU, and run-level metadata where recorded
`command.txt`	command line and configuration flags for each run cell
Run manifests	dataset manifests, sample membership, and run-root organization
LLM-judge CSVs	per-judge and ensemble validation labels for the LLM-audited subset
Human-validation exports	annotator labels, overlap-set agreement, and final human-validation outcomes when frozen

and target spans when available, the realized flip count $|F|$, and the main audit outcomes used in the paper. The same audio sample may appear in multiple rows if it is evaluated under different models, datasets, budgets, methods, presets, or acceptance strategies.

The compact ledger does not redistribute audio. For public datasets, we provide manifests and identifiers sufficient to reconstruct the evaluated sample set. For synthetic data, we release the generation and filtering scripts described in Appendix C. The ledger should be read together with the companion audit artifacts listed in Table 38.

O.2 Ledger schema

Table 37 summarizes the fields represented directly in the compact ledger. Per-flip tensor coordinates, explicit **G1–G4** gate columns, full command lines, run metadata, and LLM/human validation labels are not duplicated into this CSV; they are recoverable from the companion artifacts.

O.3 Companion artifacts

The compact ledger is designed to be joined with auxiliary files when a more detailed audit trail is needed. Table 38 lists the companion artifacts and the information they provide.

O.4 Recomputing reported quantities

For a fixed model–dataset–configuration cell, the sample denominator is

$$N = |\mathcal{D}_{\text{cell}}|.$$

The target-conditioned denominator is

$$|T| = |\{x \in \mathcal{D}_{\text{cell}} : \tau(x) \text{ is constructed}\}|.$$

Table 39: Descriptive bit-flip costs across all 1,192 SMASH rows in the original audit ledger. The pool mixes configurations, budgets, repeated utterances, and records without verified target hits; it does not represent 1,192 independent targeted successes.

Model	n	**Med.**	**Mean**	$\lvert F\rvert = 1$	$\lvert F\rvert \leq 3$	**P75**
WS	771	2	4.12	9	558	4
WLv3	371	11	12.61	20	47	18
S-M4T	50	12	10.34	16	24	18
Pooled	1192	3	7.02	45	629	10

The main paper reports StrictRate as

$$\mathrm{StrictRate} = \frac{\#\{x \in T : \exists F \in \mathcal{F}_{\mathrm{tested}}(x),\ |F| \leq B,\ \mathrm{Strict}(x, F) = 1\}}{|T|}.$$

Here, $\mathcal{F}_{\mathrm{tested}}(x)$ contains the fault configurations evaluated by the bounded search. Failure to find a strict certificate does not prove that no valid fault exists. Movement bands are computed over N: no effect corresponds to $\mathrm{WER} = 0$, local movement to $0 < \mathrm{WER} \leq 0.20$, and high-WER drift to $\mathrm{WER} > 0.20$. Source-disrupted partials and strict successes are target-conditioned side counts and are reported separately.

For the descriptive ledger bit-cost summary, $|F|$ is the logged number of committed flips, not the budget envelope B. For locality experiments, the locality-run `none` arm is used as the within-appendix reference because the locality pipeline has its own target-construction denominator, as described in Appendix I.

O.5 Bit-cost summary

For the 1,192 SMASH records in the original ledger, Table 39 summarizes the realized flip count $|F|$ per backbone. The pooled median is three flips, and 629/1,192 records commit at most three flips. The ledger also contains 45 single-bit records. WS dominates the pool ($n = 771$, median 2), while WLv3 and S-M4T have medians of 11 and 12, respectively. However, this pool mixes configurations, budgets, and repeated utterances, including records without independently verified target hits. These statistics describe the original ledger, not the cost of 1,192 verified strict semantic successes. In contrast, Section 6 reports bit costs from 73 independently verified strict numeric certificates at $B = 40$, with a pooled median of 10 flips.

O.6 Integrity checks

Before a ledger slice is used in a table or figure, we apply consistency checks using the compact ledger and companion artifacts:

1. **Budget validity.** Accepted certificates satisfy $|F| \leq B$.
2. **Movement partition.** For each fixed cell, no-effect, local movement, and high-WER drift sum to N.
3. **Strict/partial accounting.** Source-disrupted partials are computed without a strict-fail guard and may overlap with strict; both columns are reported alongside their generating cells.
4. **Gate consistency.** For independently verified strict-success slices, G1–G4 are checked against decoded transcripts and declared targets; the ledger's stored success flag alone is insufficient.
5. **Run provenance.** Each certificate links back to its cell directory; command lines, run metadata, and manifests are stored alongside the original run artifacts.
6. **Flip provenance.** The compact ledger records $|F|$; per-flip tensor, coordinate, bit-position, and quantized-value details are retained in the per-sample JSONL records.

These checks are the basis for the appendix tables in Appendix H, Appendix I, Appendix J, and Appendix K.

O.7 Versioning

The ledger and companion artifacts are versioned with the code commit and run manifests. Some appendices use specialized pipelines: for example, locality runs use a parallel locality pipeline, while cross-method baselines pool several verified slices. We therefore preserve the compact ledger and its companion files as audit records, while reported strict-success statistics are based on the corresponding verified evaluation subsets.

P Reproducibility

This appendix describes how to reproduce the reported SMASH results from the released code, run manifests, and audit artifacts. The release is organized around three layers: executable code, per-cell run artifacts, and derived audit tables. The compact verified-fault ledger described in Appendix O is the primary certificate index; companion artifacts provide command lines, run metadata, per-sample JSONL records, manifests, and validation files.

P.1 Code and execution entry point

The release contains the reference implementation, per-cell launch commands, and per-cell run metadata. Each run cell stores its exact command line in `command.txt` and its run-time metadata in `run_meta.json`; these are the source of truth for run-time flags, hardware metadata, seed, model, dataset, and preset. Code defaults are documented for interpretability, but reproduction should use the stored command lines rather than relying on defaults.

The full set of released artifacts (executable code, run-cell files, per-sample JSONL records, audit CSVs, ledger, and validation exports) is enumerated in Table 38 of Appendix O.

P.2 Compute resources

All experiments are inference-only attack runs with deterministic decoding; no fine-tuning or training was performed. Each task used a single GPU, with no multi-GPU or distributed inference. Runs used heterogeneous hardware, including NVIDIA RTX 6000 Ada Generation, A30, L40, and A100 GPUs. Per-cell hardware metadata, including GPU model, host, timestamps, and wallclock duration, is logged in `run_meta.json` alongside each cell's `command.txt`. Workstation runs included in the paper account for approximately 215 logged GPU-hours across 168 cells; cluster runtimes were logged per job but not centrally aggregated.

P.3 Canonical configuration

Table 40 summarizes the canonical settings used by the main SMASH-hybrid experiments. These are the settings referred to throughout the main paper unless an appendix explicitly states an ablation or baseline variant.

The canonical setting uses deterministic decoding: no sampling, beam size one, and maximum generation length 128 tokens. The perturbation search uses bundle size one, per-location cap $K = 3$, maximum outer search steps 30, and target strength $\gamma = 10$. Canonical SMASH excludes sign-bit flips. Reported strict successes are always post-hoc decoded-output events satisfying the full gate cascade **G1–G4**; hybrid acceptance changes the search path (Appendix E).

P.4 Models, datasets, and sample manifests

The release does not redistribute external audio corpora. Each model–dataset cell includes a manifest identifying the evaluated samples; the public corpora MultiMed-ST [16] and LibriSpeech [25] are reconstructed from their public sources using the released manifests. The synthetic dataset is reconstructed from the generation and filtering scripts described in Appendix C.For each cell, the manifest fixes the evaluated sample set and therefore the denominator N. The target-construction pipeline determines the target-constructible subset $|T|$. Both N and $|T|$ are reported in the main and appendix tables to avoid denominator ambiguity.

Table 40: **Canonical SMASH-hybrid configuration.** These are the default settings used for the main SMASH-hybrid cells unless an appendix explicitly states an ablation or baseline variant. The target strength $\gamma = 10$ is the v27 default used by the canonical hybrid runs; family-A/decode-only appendix runs override this to $\gamma = 20$. Stored command lines remain the source of truth for each run cell.

Setting	Canonical value
Random seed	42
Decoding	deterministic, no sampling
Beam size	1
Maximum generation length	128 tokens
Acceptance strategy	hybrid
Target kinds	`{num, ent, neg}`
Target strength	$\gamma = 10$
Bundle size	1
Per-location cap	$K = 3$
Maximum outer search steps	30
WER proximity threshold	$\tau_{\text{wer}} = 0.20$
Canonical bit positions	0–6
Sign-bit flips	excluded
Main budgets	$B \in \{1, 2, 3, 4, 5, 10, 20, 40\}$
LACC sidecar	$B = 100$

P.4.1 LLM-assisted controlled corpus (LACC)

LACC is a controlled evaluation harness, not a natural-speech benchmark. It contains 100 short English utterances split into four pivot-controlled categories of 25 utterances each: numeric-only, negation-only, entity-only, and mixed numeric/negation/entity spans. Each row in the JSONL release records an ID, category, pivot type, pivot span, reference text, and audio path.

Reference utterances were drafted with LLM assistance and then manually curated around span-level pivots so that the source span and a type-compatible target realization are both well defined. Audio was synthesized from the curated reference text using the project's text-to-speech (TTS) pipeline (`src/TTS.py`); the script reads JSONL rows, synthesizes one WAV per row using a Coqui TTS / VITS backend by default with a `pyttsx3` fallback, and emits rows with an `audio_path` field. Rendered WAVs are 16-bit PCM mono at 16 kHz.

The artifact filenames `synthetic.jsonl` and `synthetic_wa.jsonl` appear in the release for reproducibility only; they are internal artifact identifiers, not public-facing dataset names. The public-facing name of this corpus is *LACC*.

LACC is used in three roles: (i) target construction, where the per-row pivot supplies $\tau(x) = (k, s, r)$; (ii) the matched LACC numeric-pivot budget sweep that supports the canonical plateau analysis (Section 6); and (iii) sensitivity, ablation, and locality experiments. Real-speech evidence is reported separately on MultiMed-ST and LibriSpeech; LACC results are not natural-speech results and should not be used to claim real-speech saturation.

P.5 Run-cell layout

Each run cell is stored as a directory containing the command, metadata, decoded outputs, per-sample records, and audit summaries. The compact ledger stores a `cell_dir` pointer so that every certificate can be traced back to the corresponding run directory.

P.6 Regenerating aggregate tables

All aggregate tables in the paper are derived from the verified audit artifacts. The intended recomputation path is:

1. Load the relevant run manifests to determine the sample set and N for each cell.
2. Load target-construction outcomes to determine $|T|$.

Table 41: **Reproducibility and audit checks.** These checks are applied before using a ledger slice in a paper table or figure.

Check	Requirement
Denominator consistency	each table reports N and, where relevant, $\lvert T\rvert$
StrictRate denominator	strict rates use strict$/\lvert T\rvert$, not strict$/N$
Movement partition	no-effect, local movement, and high-WER drift sum to N for each cell
Budget validity	accepted certificates satisfy $\lvert F\rvert \leq B$
Per-location cap	canonical rows respect $K = 3$
Gate consistency	strict success requires the decoded-output gate cascade **G1–G4**
Strict/partial exclusivity	Source-disrupted partials count-decrement filter are reported under the overlapping definition consistent with the main paper (§5); partial counts may overlap with strict by construction.
Bit-cost provenance	bit-cost statistics use realized committed flip count $\lvert F\rvert$
Locality provenance	locality rows are compared against the locality-run `none` reference
Cross-method provenance	pooled baseline rows report the evaluated N explicitly
Confidence intervals	Wilson 95% intervals are recomputed from raw counts and denominators

3. Load per-sample decoded outputs and gate outcomes from `audit_samples_master.csv` and the per-sample JSONL records.
4. Load accepted certificates from `verified_attack_ledger.csv`.
5. Recompute strict counts, source-disrupted partials, WER movement bands, and realized flip counts.
6. Recompute confidence intervals using Wilson score intervals at the 95% level.

The movement bands are computed over N: no effect is $\mathrm{WER} = 0$, local movement is $0 < \mathrm{WER} \leq 0.20$, and high-WER drift is $\mathrm{WER} > 0.20$. StrictRate is computed over $\lvert T\rvert$, not N. Bit-cost statistics use the realized committed flip count $\lvert F\rvert$, not the budget envelope B.

P.7 Reproducibility checks

Before a table or figure is included in the paper, its source slice is checked for internal consistency. Table 41 summarizes the checks used for the released tables.

P.8 Known non-identical slices

A few appendix analyses use specialized pipelines or pooled slices and should not be treated as byte-identical replacements for the canonical main-paper cells. The locality appendix uses a parallel locality pipeline whose WS $\times$ LACC target-constructible denominator is $\lvert T\rvert = 92$, while the canonical budget-sweep cell reports $\lvert T\rvert = 84$. Cross-method baselines sometimes pool multiple evaluated slices, and the pooled N is shown explicitly. The decode-only/family-A configuration in Appendix E is a strict-acceptance reference, not the canonical SMASH-hybrid configuration.

These differences are documented in the relevant appendix captions and are recorded in the run manifests and ledger pointers. The main-paper headline rates should be read from the canonical SMASH-hybrid budget-sweep cells unless otherwise stated.

Q Extended Related Work

This appendix expands Section 2 with additional context on the four research threads that frame SMASH: semantic ASR risk, hardware-origin bit faults, quantized-model bit-flip attacks, and targeted ASR adversarial examples.

Sociotechnical ASR risk and semantic evaluation: ASR systems are increasingly used in safety-relevant workflows such as clinical documentation, emergency and administrative records, financial transcription, accessibility tools, and voice-driven human–computer interaction. Aggregate text-overlap metrics remain useful, but word error rate (WER) and character error rate (CER) do not

distinguish harmless edits from substitutions with high downstream cost. Severity-aware ASR evaluation therefore argues for measuring not only whether a transcript changes, but which semantic spans change [41, 6]. Clinical-dictation studies further show that recognition errors can survive into reviewed notes and vary across terminology, speakers, accents, and workflows [48, 23]. SMASH focuses on this severity-aware regime: it probes whether sparse parameter faults can induce targeted substitutions at meaning-critical spans, rather than measuring naturally occurring transcription errors.

Hardware-origin bit flips: Sparse bit-level parameter faults are motivated by a literature spanning natural soft errors, reliability faults, and adversarial fault induction. Natural single-event upsets and alpha-particle effects in semiconductor memory have been documented for decades [49, 2], and large-scale studies quantify DRAM error behavior in production systems [36]. Retention-time variation and refresh-period effects motivate retention-aware mitigation studies [17, 12]. Adversarial mechanisms extend this setting: RowHammer induces bit flips through repeated row activation [13], while Half-Double, Blacksmith, and RowPress explore stronger disturbance patterns and locality regimes [14, 11, 18]. Voltage and frequency manipulation attacks extend the fault surface beyond DRAM disturbance [38, 20]. We use this literature to motivate sparse parameter faults as a meaningful robustness regime, not to claim a complete physical exploit.

Quantized neural-network bit-flip attacks: Adversarial robustness has often studied input perturbations under norm-bounded budgets [37, 9, 19], with certified-robustness extensions [42, 4] and weight-space robustness analyses [7, 43]. Quantized inference creates a different perturbation space: discrete stored weights whose bits can be flipped with deterministic effects [10]. Progressive Bit Search showed that selected bit flips can collapse quantized image classifiers [33]; DeepHammer connected this attack class to RowHammer-style system mechanisms [45]; and targeted bit-flip work studies redirection of classifier labels under sparse budgets [34, 1]. Transformer-oriented work studies defenses and resilience to parameter faults [39, 21], and recent language-model attacks study stealthy bit-flip degradation while preserving output naturalness [44]. These works establish sparse bit-level parameter faults as important, but their success events are typically classification failure, target-label redirection, or aggregate degradation.

Input-side ASR adversarial attacks: ASR adversarial-robustness work has mostly studied perturbations to the input waveform. Targeted audio adversarial examples can induce attacker-chosen transcriptions [3], with later work improving imperceptibility and psychoacoustic realism [28, 35]. Practical voice-command attacks and universal audio perturbations broaden the input-side surface [46, 22]. Recent work on Whisper further shows that robustness to noise or distribution shift does not imply adversarial robustness, including attacks that mute or redirect model behavior [24, 32, 31]. SMASH asks the analogous parameter-side question: with the input waveform fixed, which semantic substitutions are reachable through sparse quantized weight-bit faults?

High-value semantic spans and the SMASH gap: Within transcripts, not all spans carry equal operational importance. Inverse text normalization systems explicitly handle semiotic classes such as cardinals, decimals, dates, measures, money, and identifiers [27, 47]; named entities are another high-value class motivating dedicated correction methods [40]. Prior bit-flip work targets classifier labels or aggregate model degradation, and prior ASR adversarial work perturbs the input waveform to obtain targeted transcripts. To our knowledge, prior work has not studied targeted semantic substitution in seq2seq ASR induced by sparse quantized weight-bit faults, where success requires both localized target realization and transcript-level plausibility. This is the gap SMASH addresses, with numeric/scalar pivots as the empirical main study and entity/negation pivots reported as scope evidence in Appendix L.

Event mismatch and comparison scope: The comparison target in this paper is not generic bit-flip effectiveness, but declared source→target semantic span realization in a seq2seq ASR transcript under sparse quantized weight faults. This event differs from prior bit-flip objectives: PBS and DeepHammer primarily measure classifier collapse or aggregate degradation; T-BFA and integer-programming variants target discrete classifier-label redirection; SilentStriker targets LLM task degradation under naturalness constraints; and input-side ASR attacks perturb the audio waveform rather than stored ASR parameters. None of these methods natively defines success as localized realization of a declared semantic replacement inside an ASR transcript. We therefore treat them as closest adaptable baselines under matched post-hoc G1–G4 gates. The comparison asks whether

movement- or degradation-oriented fault methods realize declared semantic targets; it does not establish optimality against a future targeted span-CE or target-margin baseline designed specifically for this event.

R Extended Limitations

This appendix expands Section 7 with longer-form discussion of the main scope boundaries.

Harm operationalization: We validate semantic errors in ASR transcripts, not end-to-end real-world harm. Numeric, entity, and date substitutions are safety-relevant because they can affect clinical, dispatch, financial, accessibility, or records workflows, but whether a changed transcript causes harm depends on deployment context: interface design, human review, downstream automation, and organizational safeguards. Studying those deployment-time effects is important but outside the scope of this paper.

Threat-model scope: SMASH models a white-box evaluator with read access to the quantized model and write access to a declared parameter preset $\mathcal{W}$ under a sparse INT8 bit-flip budget (Section 3). This is a stronger capability than most realistic physical attackers have, and we use it deliberately as a diagnostic upper-bound probe. Hardware fault mechanisms such as RowHammer [13], RowPress [18], Half-Double [14], and Blacksmith [11] motivate sparse-bit perturbations as a relevant setting. Our locality experiments (Section 6) add software-side placement constraints, but they do not model DRAM row mapping, refresh behavior, cache coherence, or a full exploit chain. Physical realization of any specific certificate remains future work.

Quantization, architecture, and dataset scope: Experiments cover INT8 round-to-nearest weight quantization on three seq2seq ASR backbones: WS, WLv3, and S-M4T, evaluated on LACC, MultiMed, and LibriSpeech. We do not evaluate other quantization regimes such as INT4, AWQ, GPTQ, or K-quants, nor other ASR families such as CTC, Conformer, or RNN-T systems. Generalization to other languages, acoustic conditions, and tokenizer designs remains open. Numeric-pivot reachability may be tokenizer-sensitive because character-level, subword, and multi-digit numeric tokenization expose different reachable sets.

Negation reachability: Negation is a boundary case for the current SMASH formulation. Under the canonical configuration, negation produced zero strict successes across the evaluated model–dataset cells (Appendix L). We therefore do not treat negation as a successful target class in this paper. This result is consistent with the edit structure: numeric and many entity pivots are mostly substitutional and approximately position-preserving, whereas negation often requires insertion, deletion, auxiliary rewriting, or contraction-level changes. The current span-conditioned target loss is better matched to substitutional edits than to insertion/deletion-style polarity edits. We treat the zero-success negation result as a limitation of the current target construction, not as evidence that negation is intrinsically robust to sparse parameter faults.

Audit protocol, reproducibility, and defenses: Strict success requires a constructible target and an accepted decoded transcript satisfying **G1**–**G4**. The denominator $|T|$ therefore depends on the target-construction pipeline: a more aggressive constructor could increase target coverage, while a stricter constructor could reduce it. We address this by reporting N and $|T|$ for every cell, releasing the construction scripts and verified-fault ledger (Appendices P and O), and separating target coverage from search efficacy. SMASH is an evaluation method rather than a defense; certified mitigations, integrity checks, and deployment-time monitors for the identified reachable set are outside the scope of this work.

Targeted-baseline scope: The comparison target in this paper is declared source→target semantic span realization in a seq2seq ASR transcript under sparse quantized weight faults. Existing bit-flip methods optimize different events: classifier collapse, discrete label redirection, or LLM degradation under naturalness constraints. We therefore treat PBS, SilentStriker, URBF, and TGRF as closest adaptable baselines evaluated under matched post-hoc G1–G4 gates. This comparison tests whether movement- or degradation-oriented methods realize declared semantic targets. It does not establish optimality against a future denominator-matched targeted span-CE or target-margin baseline using the same target tuples, budget, preset, bit policy, decoding configuration, and gates.

S Extended Broader Impacts

This appendix expands the Broader Impacts paragraph in Section 8 with concrete deployment scenarios, defender-side guidance, and release-scope boundaries.

Deployment scenarios: The motivating settings are ASR workflows where localized semantic errors can have operational cost. In clinical dictation, substituted dosages, medication names, or numeric values can affect downstream documentation. In emergency dispatch, substituted addresses, room numbers, or coordinates can misroute responders. In financial and administrative transcription, substituted amounts, account identifiers, or dates can alter records used by downstream systems. In accessibility tools, substituted polarity, numbers, or named entities can change the meaning delivered to users. SMASH does not show that these harms occur end-to-end; it provides a worst-case audit of whether such substitutions are reachable in quantized ASR transcripts.

Defender-side recommendations: We recommend treating SMASH-style certificates as a worst-case evaluation harness for ASR pipelines in safety-relevant settings. Practical mitigations include semantic-span checks for numeric, identifier, date, and named-entity content; redundancy for high-impact fields; human review before downstream automation acts; runtime model-integrity checks such as weight-buffer hashing or attestation; and robustness tests that go beyond aggregate WER. These mitigations target the failure mode surfaced here: localized, meaning-critical transcript substitutions.

Primary audience: The parameter-fault literature already establishes sparse bit flips as a public robustness concern. SMASH's contribution is not a new physical exploit, but a semantic-substitution audit: it asks which meanings are fragile under sparse quantized weight-bit perturbations. This is useful to defenders because integrity checks and deployment monitors need to prioritize high-impact spans rather than only aggregate transcript quality.

Release-scope boundaries: The supplementary release is scoped to controlled evaluation and demonstration materials. It does not include a RowHammer payload, DRAM-mapping tooling, an end-to-end physical fault-injection chain, or a script targeting a deployed ASR service. The threat model assumes white-box read/write access to a declared parameter preset of a quantized model (Section 3); this is a diagnostic measurement instrument, not a deployed attack. External audio is not redistributed, and no new pre-trained ASR weights are released.

Misuse risk: The work is dual-use: a sufficiently capable actor with parameter-write access could use semantic-substitution certificates to understand which targets are reachable on similar quantized ASR backbones. We accept this trade-off because the public bit-flip literature already establishes the parameter-side attack surface, while defenders need concrete evidence about which semantic spans require integrity checks. Deploying these certificates against real systems would still require substantial additional capabilities that are outside the paper's scope.

Audit value: SMASH is intended to make the question "which meanings can be changed?" concrete and measurable for safety-relevant ASR. We hope the framework, LACC corpus, and verified-fault ledger support future work on defended quantized inference, severity-aware ASR robustness benchmarks, and integrity-aware deployment patterns for ML systems beyond ASR.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The abstract and Section 1 state the contribution (SMASH: a targeted semantic bit-flip search framework for quantized seq2seq ASR), the empirical scope (three backbones $\times$ three corpora; numeric/scalar pivots in the main study; entity and negation pivots in Appendix L), the headline reachability counts at $B{=}20$ on LACC ($24/45$ WS, $19/44$ WLv3, $5/34$ S-M4T), the realized bit-cost regime, and the $\kappa{=}1.00$ human/LLM-ensemble validation.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: A dedicated §7 is included before the conclusion, with additional inline caveats in §8. We discuss: (i) the white-box read/write threat model as a diagnostic upper bound rather than a realistic attacker; (ii) hardware-fault realism (DRAM mapping, refresh, cache, exploit chains are not modeled); (iii) scope to INT8 round-to-nearest quantization on the named backbones, datasets; (iv) the gap between transcript-level semantic substitution and downstream real-world harm; and (v) the lower reachability of negation pivots under the canonical configuration.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.

- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

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

   Answer: [N/A]

   Justification: The paper is empirical. The formulas in §3 and §4 (budget constraint, INT8 bit-flip semantics, gate cascade G_1–G_4) are definitional, not theoretical results, and no theorems or proofs are claimed.

   Guidelines:

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

4. **Experimental result reproducibility**

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

   Answer: [Yes]

   Justification: Appendix P documents the canonical SMASH-hybrid configuration, fixed seed (42), deterministic decoding (no sampling, beam size 1, max generation length 128), per-cell run manifests with stored `command.txt` and `run_meta.json`, the accepted-fault ledger (Appendix O), and the table-recomputation pipeline. Quantization presets and tensor lists are in Appendix A; LACC construction is in Appendix C; baseline grids and matched-cell protocol are in Appendix K.

   Guidelines:

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

(a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
(b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

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

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

   Answer: [No]

   Justification: The supplementary release ships a self-contained live-attack demonstration pack: 15 LACC samples (audio, per-sample search-time JSONL records, and a manifest with source→target pairs), the live SMASH-hybrid attack implementation and a Jupyter notebook driver, environment specifications and OS-specific setup scripts, and a README. Reviewers can reproduce the live targeted-substitution result on their own hardware in minutes. The complete reference implementation, full LACC corpus, verified-fault ledger, audit summaries, and validation packets are documented in Appendix P; public release is planned in anonymized form alongside or after review, subject to conference policy. External audio (LibriSpeech, MultiMed-ST) is not redistributed; per-cell sample manifests are provided so the evaluated subsets can be reconstructed from public sources. All artifacts in the submitted zip are anonymized. The complete implementation is not yet publicly available; the submitted demonstration covers only a subset of experiments.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

Justification: §5 fixes per-cell sample size (N=100), seed (42), deterministic decoding, the canonical SMASH-hybrid configuration (γ=10, `smash_decode_trials=2`, $\tau_{\text{wer}} = 0.20$, K=3, max 30 outer search steps, bundle size 1, bit positions 0–6 excluding sign bit), the budget grid $B \in \{1, 2, 3, 4, 5, 10, 20, 40\}$ with a LACC-only B=100 sidecar, and the per-backbone quantization presets. Sensitivity analyses on γ, τ_{wer}, K, and bundle size are in Appendix G; full canonical settings are tabulated in Appendix P.

Guidelines:

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

7. **Experiment statistical significance**

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

Answer: [Yes]

Justification: All headline reachability rates are reported as $k/|T|$ with the constructible denominator visible. Wilson 95% confidence intervals on strict rate are reported in Appendix J for every model×corpus×budget cell, and the budget-sweep figure displays Wilson CI bands. The validation κ=1.00 is computed across two independent rater pools (human-majority and four-judge LLM ensemble) on the 105-item shared set using Cohen's κ; per-question κ (0.51–0.85) is also reported.

Guidelines:

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

8. **Experiments compute resources**

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

Answer: [Yes]

Justification: Compute resources are documented in Appendix P (§Compute resources). All experiments are inference-only attack runs with deterministic decoding and no fine-tuning or training. Each task used a single GPU on heterogeneous NVIDIA hardware; no multi-GPU or distributed inference was used. Per-cell command lines, hardware metadata, and wallclock durations are logged in `run_meta.json`. Workstation runs included in the paper

account for approximately 215 logged GPU-hours; cluster runtimes are logged per job but not centrally aggregated.

Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: The work is dual-use ASR robustness research. Our threat model is a white-box diagnostic upper bound (§3, §7); we do not demonstrate a complete physical exploit, target deployed systems, or redistribute external audio. Released artifacts are limited to controlled evaluation and demonstration materials on LACC, plus manifests for reconstructing public-corpus subsets. Anonymity is preserved at submission. Human and LLM-ensemble validation use blinded text-only annotation of model outputs; no real personally identifying information is collected or processed.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: §8 (Broader Impacts paragraph) discusses positive impacts through robustness auditing for ASR systems in safety-relevant settings such as clinical dictation, emergency dispatch, financial records, accessibility tools, and administrative transcription. It also discusses negative impacts because the method is dual-use: sparse model-side perturbations could in principle be misused to induce targeted transcript errors. The paper limits this risk by framing SMASH as a white-box diagnostic upper bound, not a complete physical exploit or deployed-system attack, and by documenting limitations and safeguards in Sections 3 and 7. We also discuss mitigations including semantic-span checks, redundancy for high-impact fields, human review, runtime integrity checks, and robustness tests beyond WER.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate

to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

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

    Answer: [Yes]

    Justification: The release is scoped to controlled evaluation artifacts and a LACC demonstration pack, not a deployed-system attack. It does not include a RowHammer payload, DRAM-mapping tooling, or an end-to-end physical fault-injection chain; the threat model assumes white-box read/write access to a declared parameter preset as a diagnostic measurement instrument. External audio is not redistributed, LACC is a small synthesized controlled corpus rather than scraped data, and no new pre-trained ASR weights are released.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

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

    Answer: [Yes]

    Justification: Third-party assets are credited at first use and used under their stated terms: Whisper-small.en and Whisper-large-v3 [29, 30] under MIT; SeamlessM4T-v2-large [5] under CC-BY-NC 4.0 and used only for non-commercial research; LibriSpeech test-clean [25] under CC BY 4.0; and MultiMed-ST [16] under its released dataset terms. LACC audio is synthesized with the Coqui `TTS` package and the documented `tts_models/en/ljspeech/vits` model pipeline; the supplementary release includes only synthesized LACC audio and our evaluation artifacts, not third-party model weights.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

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

    Answer: [Yes]

    Justification: The paper introduces three new assets, each documented in the appendix and released or linked with the supplementary materials as appropriate: (i) the LACC controlled corpus (100 utterances over four pivot categories), with construction protocol, validation rules, and TTS configuration in Appendix C; (ii) the verified-fault ledger, with schema and per-row provenance fields documented in Appendix O; and (iii) the SMASH reference implementation, with per-cell launch commands and run-cell layout documented in Appendix P. All submitted assets are anonymized.

    Guidelines:

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

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

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

    Answer: [Yes]

    Justification: Thirteen volunteer annotators, peers and friends of the authors and with no monetary compensation, labeled a 137-entry annotation packet via a blinded Streamlit interface. The headline human–LLM comparison uses 105 shared outputs. Appendix M.1 reproduces the full four-question rubric (Q1–Q4: source disappearance, target appearance, extra meaning change, and broken output), and Appendix M.2 documents the blinding policy, with method, model, dataset, budget, and source-run directory hidden, plus a screenshot of the grading app (Figure 4). Annotator coverage and the majority-vote rule are reported, including the criterion that annotators with $\geq 94/105$ completed items contribute to the headline composite.

    Guidelines:

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

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

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

Answer: [Yes]

Justification: The study involved blinded, text-only annotation of machine-produced ASR outputs by volunteer annotators. No audio, demographic data, or sensitive personal data was collected, and labels were not used to make decisions about annotators. Under applicable institutional policy, this minimal-risk annotation activity did not require IRB review.

Guidelines:

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

16. **Declaration of LLM usage**

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

Answer: [Yes]

Justification: LLMs are used in two methodology-relevant roles. First, LLM assistance was used informally during early LACC sentence-shell drafting, followed by manual rewriting, curation, and validation; the released corpus text is the post-curation output (Appendix C). Second, a four-judge LLM ensemble (DeepSeek-R1-70B, Llama-3.1-70B, Qwen-2.5-14B, GPT-4o) is used as an independent validation rater pool. The prompt, judge configuration, parser/retry policy, JSON schema, and aggregation rule are documented in Appendix M. Generative AI and language-editing tools (e.g., ChatGPT, Claude) were also used for grammar correction and language refinement. All suggested edits were reviewed and verified by the authors, who take full responsibility for the final manuscript.

Guidelines:

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