When the Attack Is Not an Attack: Validity Failures in LLM-Generated Red-Teaming

Yihang Jiao
University of Illinois Urbana-Champaign
yihangj3@illinois.edu

Abstract

Language models are increasingly used to generate the probes that test other models. The appeal is scale and fluency: an LLM can rewrite an adversarial example into text a person might actually write. This paper asks what such a pipeline is measuring. I identify four ways a generated probe can be counted in a red-teaming result without being a valid test, and I quantify each on 32 model–attack–dataset configurations and 600 fresh generations. **Label validity**: the rewriting prompt that produces the most label flips is the one that instructs the model to change the label, so the probe is a different input rather than an attack. An independently trained referee model flips on 86–87% of those successes, and requiring the flip to be target-specific *reverses* the ranking of the two prompts — 29/180 genuine vulnerabilities for the generic prompt against 16/180 for the task-specific one, where flip rate alone had said 119/180 against 58/180. The cosine-similarity constraint usually offered as a defence catches this on topic classification and misses it on sentiment, where an embedding barely moves when a review is rewritten from 3/10 to 8/10. **Input validity**: on sentence-pair tasks most rewrites collapse the two segments and are not well-formed classifier inputs; because malformed probes are scored as failures, leaving them in *understates* attack success, 12.5% against 41.7%. The two errors therefore push in opposite directions and do not cancel. **Generator validity**: the generator is itself safety-trained and refuses 6.2% of the time, at a rate that varies with the prompt being studied (11.0% against 0.6%), and a fifth of those refusals are scored as successful attacks. **Measurement validity**: the reported metric must be a function of the generated text, a condition an earlier version of this study did not meet. Underneath these, LLM rewriting is a trade: it keeps only 44.5% of the attacks it is applied to while making them far more fluent (median perplexity 22.9 against 522). I close with five numbers any automated red-teaming report should carry.

1 Introduction

Automated red-teaming increasingly uses one language model to probe another. The attraction is obvious. Word-substitution attacks such as TextFooler (Jin et al., 2020) and PWWS (Ren et al., 2019) produce inputs that are effective but stilted, and a perturbation no fluent speaker would write says less about deployment risk than one that reads normally. An LLM can rewrite the whole sentence rather than swap tokens, and can therefore generate probes that are fluent by construction.

The generated probe, however, only tells us something about the target model if it is a valid test of that model. This paper asks how often it is, and what the answer does to the numbers such pipelines report.

A probe can fail to be a valid test in three distinct ways, and each one is invisible in the summary statistic that red-teaming reports usually carry:

- **Label validity.** The rewrite may have changed the input's true label. Then a prediction change is correct behaviour, not a vulnerability.
- **Input validity.** The rewrite may not be a well-formed input for the task at all — for a sentence-pair task, not a pair.

- **Generator validity.** The generating model may have refused, and its refusal is then scored as if it were a probe.
- **Measurement validity.** The reported number may not be a function of the generated text, if the pipeline scores something else.

I measure all four. The first three are quantified on 600 generations produced for this paper under controlled prompt conditions; the fourth is demonstrated on a worked instance, an earlier version of this study, where it silently determined the published result.

Underneath the three failures is a simpler question that the same experiments answer. Rewriting is applied to attacks that *already succeed*, so its cost is directly measurable: how many survive being rewritten?

My contributions are:

- **Rewriting keeps under half the attacks it is applied to.** Across 32 configurations, LLaMA-2 rewrites of successful word-substitution attacks still flip the classifier 44.5% of the time, while being far more fluent (Section 3).
- **Label validity fails in the direction that flatters the attacker.** A task-specific prompt roughly doubles retention, but on topic classification it does so by instructing a topic change. Flip rate alone overstates attack success by $3.05\times$ pooled and $19.5\times$ on AG News (Section 4).
- **Input validity fails in the opposite direction.** Malformed sentence-pair probes are scored as failures, so not checking them understates attack success — 12.5% against 41.7%. An explicit structure constraint nearly doubles validity but does not fix it (Section 5).
- **Generator refusals contaminate the comparison.** The generator declines 6.2% of requests, at 11.0% under the generic prompt against 0.6% under the task-specific one, and a fifth of those refusals are scored as successful attacks (Section 6). Excluding them leaves both results above intact.
- **I measure the generate-and-filter pipeline** proposed for this setting and report what each constraint removes; the similarity constraint, not fluency, does most of the work (Section 7).
- **A worked instance of measurement-validity failure**, and the one-line check that catches it (Section 8), followed by three numbers a red-teaming report should carry (Section 9).

2 Experimental Setup

Attacks and targets. I use TextFooler (Jin et al., 2020) and PWWS (Ren et al., 2019) as implemented in TextAttack (Morris et al., 2020), against `bert-base-uncased` and `roberta-base` classifiers fine-tuned on each dataset. Attacks were run over 500 test examples per configuration (277–500 after dataset exhaustion), yielding 190–479 successful attacks per configuration to sample from.

Datasets. Single-sentence tasks: IMDb, SST-2, MR (Rotten Tomatoes), AG News, and CoLA. Sentence-pair tasks: RTE, QNLI, and MRPC. Pair inputs are two segments joined by a `<SPLIT>` marker, following TextAttack's convention.

Probe generation. Probes are produced by `Llama-2-7b-chat-hf` (Touvron et al., 2023) in half precision, with nucleus sampling ($p = 0.9$, $T = 0.8$, 160 new tokens), through the model's chat template. I strip the scaffolding chat models place at both ends of an answer ("Here is the revised text:" and trailing explanations) before scoring; leaving it in would inflate perplexity and depress similarity for whichever prompt elicits more chattiness, which is the quantity under study.

Measurements. For each probe I record whether the target classifier's prediction differs from its prediction on the original text (*flip*); cosine similarity between sentence embeddings of the original and the probe, using `all-MiniLM-L6-v2`; and perplexity under GPT-2-XL. For pair tasks I also record whether the probe preserved the two-segment structure and is a well-formed input at all.

The comparison of interest. Probes are generated only for examples on which the word-substitution attack already succeeded, so the baseline is fixed at 100% by construction and the quantity to report is what fraction of those attacks survives rewriting.

3 Rewriting Keeps Under Half the Attacks

I first re-evaluate a stored corpus of rewrites spanning 32 configurations (2 target architectures $\times$ 2 attack recipes $\times$ 8 datasets), running each target classifier on the rewritten text.

On the 20 single-sentence configurations, rewrites still flip the classifier on average 44.5% of the time. Every one of these was a successful attack before rewriting, so more than half are destroyed by it. Retention varies enormously:

Dataset	Mean	Min	Max
AG News	0.907	0.706	1.000
IMDb	0.764	0.667	0.889
MR	0.268	0.077	0.462
SST-2	0.217	0.105	0.300
CoLA	0.069	0.000	0.133

The spread is a factor of thirteen, and it lines up with an implementation detail: AG News and IMDb were the two datasets for which a hand-written prompt template existed, while the others fell through to a generic one. Dataset and prompt are confounded here, so the next section separates them — and in doing so uncovers the first validity failure.

4 Label Validity

To break the confound I run *both* prompt variants on *every* dataset. The generic prompt is the original fallback ("Improve the text fluency while altering the classification…"). The task-specific prompts follow the shape of the hand-written ones: for IMDb and AG News these are the originals; for SST-2 I write one in the same shape, so that "has a task-specific prompt" becomes a manipulable variable rather than a property of the dataset. I sample 30 successful TextFooler attacks per cell.

Pooled over 360 probes, the task-specific prompt roughly doubles the fraction of attacks that survive: 119/180 against 58/180 ($p = 1.7 \times 10^{-10}$). The effect holds in 5 of 6 cells and is significant in 4; the two SST-2 cells show no reliable difference.

Why the prompt wins matters more than that it wins. An adversarial example must also preserve the input's meaning, and therefore its true label. Splitting by task type shows two very different mechanisms behind the same doubling:

Dataset	Target	Specific	Generic	p
AG News	BERT	20/30	10/30	0.019
AG News	RoBERTa	19/30	8/30	0.009
IMDb	BERT	22/30	7/30	0.0002
IMDb	RoBERTa	28/30	6/30	$< 10^{-4}$
SST-2	BERT	12/30	13/30	1.000
SST-2	RoBERTa	18/30	14/30	0.438
Pooled		**119/180**	**58/180**	1.7×10^{-10}

Table 1: Probes that still flip the classifier, by prompt variant, with dataset and target held fixed. Two-sided Fisher exact test per cell.

Dataset	Flip rate	Meaning-preserving	Ratio
AG News	65.0%	**3.3%**	19.5×
SST-2	50.0%	15.0%	3.3×
IMDb	83.3%	46.7%	1.8×
Pooled	**66.1%**	**21.7%**	**3.05×**

Table 2: Attack success under the task-specific prompt as a flip-rate-only report would state it, against the same probes required to keep similarity ≥ 0.75.

	Flip rate	Similarity
Sentiment (IMDb, SST-2)		
specific	0.667	0.713
generic	0.333	0.705
Topic (AG News)		
specific	0.650	**0.361**
generic	0.300	0.691

On sentiment the specific prompt doubles the flip rate at no cost in similarity, 0.713 against 0.705. On AG News the same doubling arrives with similarity collapsing from 0.691 to 0.361 — and the reason is written in the prompt, which instructs the model to change the topic "from world to sports". A text whose topic has actually changed is not an adversarial example. It is a different input with a different correct label, and a prediction change on it is the classifier behaving correctly.

How much this distorts a report. Table 2 puts a first number on it. If a pipeline reports flip rate alone, and we then ask how many of those flips came from probes that still mean what the original meant, the reported figure overstates measured attack success by $3.05\times$ pooled — and by $19.5\times$ on the task where the prompt instructs a semantic change.

The similarity constraint is not enough, and fails unevenly. A cosine similarity threshold is the standard defence against this, and on AG News it works. It does not work on sentiment, because sentence embeddings track topic far more than valence. This rewrite of an IMDb review has cosine similarity 0.964:

> **Original.** Sloppy film noir thriller which doesn't make much of its tension promising set-up. (3/10)
> **Rewrite.** Engaging film noir thriller that successfully builds on its intriguing premise. (8/10)

Dataset	Prompt	Referee also flips	Mean sim.
AG News	specific	87.2%	0.268
AG News	generic	44.4%	0.612
IMDb	specific	86.0%	0.749
IMDb	generic	23.1%	0.719
SST-2	specific	86.7%	0.640
SST-2	generic	66.7%	0.667

Table 3: Among probes the target counted as successful attacks, the fraction on which an independently trained referee of the other architecture flips too.

The review's verdict has been reversed — 3/10 to 8/10 — and the embedding barely moves, because both sentences are about the same film noir thriller. A similarity filter admits it.

A referee model separates attacks from label changes. An adversarial example is supposed to fool *one* model, not to persuade every model. So I re-score each probe with an independently trained classifier of the other architecture: RoBERTa referees BERT's probes and BERT referees RoBERTa's. If the referee changes its prediction in the same direction, the probe most likely changed the true label; if only the target changes, the probe found something specific to the target — an actual vulnerability.

Table 3 shows that **86–87% of the task-specific prompt's successes also flip the referee**. They are not vulnerabilities; they are inputs whose label changed. Requiring the flip to be target-specific reverses the comparison:

Prompt	Flip rate	Target-specific flips
task-specific	**119/180** (0.661)	16/180 (0.089)
generic	58/180 (0.322)	**29/180** (0.161)

By flip rate the task-specific prompt wins two to one. By actual vulnerabilities found, the generic prompt wins, 29 against 16 ($p = 0.055$, Fisher exact). **The prompt that scores best on the reported metric is the worse red-teaming prompt.**

A stricter reading of the referee. Counting a probe as target-specific whenever the referee does not flip is generous, because the referee may fail to flip only for having disagreed with the target on the original text in the first place. The two models do agree on the original 94.4% of the time, but the loophole is real: among the five task-specific probes that pass both the similarity and referee checks, one is a review the target read as negative and the referee already read as positive. Requiring the two models to agree on the original as well leaves 12/180 genuine vulnerabilities for the task-specific prompt against 25/180 for the generic one. The reversal is not an artifact of the looser criterion; under the stricter one the ratio widens.

Why the similarity check misses this. The two checks catch different things, and only together do they catch much. Of the 119 flips under the task-specific prompt, 39 survive the similarity threshold, 16 survive the referee, and **only 5 survive both**. Broken out by task, the similarity threshold lets through 3% of AG News label changes but 35% on SST-2 and 56% on IMDb — the failure is task-dependent in exactly the way an embedding would predict, and a practitioner applying one uniform threshold has no way to see it.

Prompt	Well-formed	Unchecked	Valid only
generic	0.300	12.5%	**41.7%**
structured	**0.575**	21.7%	37.7%

Table 4: Sentence-pair tasks, 240 probes. *Well-formed*: fraction preserving the two-segment structure ($p = 2.8 \times 10^{-5}$ between prompts, Fisher exact). *Unchecked*: attack success if malformed probes are left in the denominator. *Valid only*: attack success among well-formed probes.

5 Input Validity

For RTE, QNLI and MRPC the classifier expects two segments. Most probes do not supply them: the model merges premise and hypothesis into a single passage, often appending a commentary sentence. In the stored 32-configuration corpus, only 100 of 198 pair-task rewrites preserved the structure; for the two RTE configurations under PWWS, *zero* of 31 did.

I test whether an explicit constraint fixes this, comparing the generic prompt against one stating that the input is a pair, that the segments are separated by `<SPLIT>`, and that the answer must return exactly two sentences in the same order.

Two things follow. First, stating the constraint nearly doubles structural validity, from 30.0% to 57.5%, and raises the yield of usable probes from 12.5% to 21.7% — but the model still violates an explicitly stated format more than 40% of the time, so the constraint is a mitigation and not a fix.

Second, and more consequential for anyone reading a red-teaming number: malformed probes are counted as attack failures, never as successes — none of the malformed probes registered a flip. Leaving them in the denominator therefore **understates** attack success, 12.5% against 41.7% for the generic prompt. Label-validity failure inflates the reported number and input-validity failure deflates it. **The two errors push in opposite directions, so they do not cancel**; which one dominates depends on the task, and neither is visible in the summary statistic.

6 Generator Validity: When the Red-Teamer Refuses

A fourth failure has nothing to do with the target model. The generator is itself safety-trained, and it sometimes declines to produce the probe:

> "I cannot provide a revised text that could potentially be used to mislead or deceive, as it is unethical and goes against my programming rules…"

The refusal is then scored as though it were a probe. Of 600 generations, 37 (6.2%) begin with a refusal, and 7 of those 37 (18.9%) are counted as successful attacks — the refusal text carries a different predicted label than the original, at a mean similarity of 0.216 to it. The target model is being credited with a vulnerability to a sentence about the generator's programming rules.

What makes this more than noise is that it is *not* distributed evenly over the conditions under study (Table 5). The generic prompt, which asks the model to "maintain adversarial properties to fool the classifier", is refused at 11.0%; the task-specific prompts, which ask for a sentiment or topic change without naming the adversarial purpose, at 0.6%. RTE draws refusals at 17.5%, plausibly because instructing a model to flip an entailment relation reads as a request to assert something false.

So the prompt variable in Section 4 moves refusal rate and attack success together, and refusals are scored as failures. That is a confound in my own headline result, and it has to be checked rather than assumed away.

	Refusal rate	
By prompt		
generic	11.0%	(33/300)
structured	2.5%	(3/120)
task-specific	0.6%	(1/180)
By task		
RTE	17.5%	(21/120)
AG News	6.7%	(8/120)
QNLI	4.2%	(5/120)
IMDb	1.7%	(2/120)
SST-2	0.8%	(1/120)

Table 5: Generations beginning with a refusal, by prompt variant and by task. Detection requires a refusal formula at the start of the generation; all sampled matches were genuine refusals.

Stage	Kept	Frac.	Sim. / PPL
All probes	360	1.000	0.648 / 33.8
+ flips the classifier	177	0.492	0.596 / 28.1
+ similarity ≥ 0.75	64	0.178	0.840 / 27.2
+ perplexity ≤ 150	55	0.153	0.834 / 22.9

Table 6: Cumulative effect of the three conditions. Sim. is mean cosine similarity, PPL median GPT-2-XL perplexity, of the surviving set.

Robustness check. Excluding refusals does not change the finding. On single-sentence tasks the task-specific prompt retains 118/179 against 58/170 for the generic prompt once refusals are dropped ($p = 4.2 \times 10^{-11}$), against 119/180 and 58/180 with them included — the effect is if anything slightly stronger. On sentence-pair tasks, structural validity among non-refusals is 0.590 for the structured prompt against 0.351 for the generic one, compared with 0.575 and 0.300 overall. Both conclusions survive.

Why it still matters. The check is cheap and the correction is small here, but neither is guaranteed in general. A red-teaming pipeline whose generator refuses at a rate that varies with the condition being compared will attribute the generator's safety training to the target's robustness. As generator models are trained to decline adversarial requests more reliably, this failure grows rather than shrinks, and it is invisible in an attack success rate: a refusal and a failed attack look identical in the output column.

7 What the Filter Trades

The standard response to unreliable generations is to filter them: keep only probes that flip the label, stay semantically close, and read fluently. I elaborate that filter into a per-candidate test — each probe scored on all three conditions and kept or dropped individually — and apply it to all 360 single-sentence probes.

The filter retains 15.3% of generations and does what it claims: mean similarity rises from 0.648 to 0.834. Two observations matter for practice. First, the *similarity* constraint does nearly all the filtering — it removes 113 of the 177 probes that flip, while the fluency bound removes 9 more. This inverts the usual framing, in which fluency is the reason to filter; here fluency is nearly free and meaning preservation is what costs.

Second, the fluency gain over the attacks being replaced is real and large. Over 428 attacked examples, the median GPT-2-XL perplexity of a TextFooler perturbation is 522, against 22.9 for filtered rewrites, and only 15.4% of 403 successful word-substitution attacks satisfy the same similarity and fluency constraints. **LLM**

rewriting buys a large improvement in fluency and pays for it in attack retention. Neither side of the trade is small, and the exchange rate is set by the prompt.

8 Measurement Validity: A Worked Instance

The third failure is the one that most easily escapes notice, because it leaves no trace in the outputs. An earlier version of this study reported that filtered LLaMA-2 generations outperformed word-substitution attacks, with improvements up to 19.31% relative. Those numbers do not survive scrutiny, and the reasons are worth recording because the failure mode is easy to reproduce.

The classifier was never run on the probes. The pipeline wrote LLaMA output into a `llama_text` column, and the reported attack success rate was computed from TextAttack's `result_type` field. That field describes the word-substitution attack, which ran *before* rewriting. The probes therefore never entered the reported numbers. Section 3 supplies the missing evaluation.

The headline table compared sample sizes. Each "filtered" figure was the attack's success rate over 20 examples; each "baseline" figure was the same attack over 277–500. All five rows reproduce exactly from those two runs, so the reported improvement was a difference in sample size.

Filtering happened at a different layer than described. The pipeline did constrain and score generations, but not where the published algorithm places it. A similarity constraint, $\texttt{min\_cos\_sim} = 0.6$, was passed to the attack's word-substitution step, and perplexity and grammar were computed afterwards as aggregate statistics over each result file. A per-candidate test on all three thresholds is not in that code, and the values 0.75 and 150 appear nowhere in it. Section 7 implements the per-candidate test.

The check that catches it. The lesson is narrow and mechanical: *if a pipeline produces new text, the reported metric must be a function of that text.* A one-line assertion — that the evaluated column is the generated column — would have caught all three problems. That this happened in a study whose entire subject was generated text is the point: measurement-validity failure is not a sign of carelessness about the topic, it is a default of the tooling, which will happily join a generations column to a metric computed from something else.

9 Five Numbers a Red-Teaming Report Should Carry

Each failure above is invisible in an attack success rate, and each is visible in one additional number:

1. **Semantic similarity beside every flip rate.** Without it, a prompt that changes the label scores best. On AG News this is the difference between reporting 65.0% and 3.3%.
2. **The fraction of successes an independent referee model reproduces.** Similarity is not sufficient on its own — it misses 56% of the label changes on IMDb. A referee of a different architecture catches them across all three tasks, and here it reverses which prompt is better.
3. **The fraction of generations that are well-formed inputs.** Without it, malformed probes sit in the denominator as failures. On sentence-pair tasks this is the difference between 12.5% and 41.7%.
4. **The generator's refusal rate, per condition.** Without it, a generator that declines more often under one prompt than another will move the comparison. Here that is 11.0% against 0.6%.

5. **The number of generations actually scored by the target model.** Without it, a pipeline can report a metric computed from something other than what it generated, and nothing in the output will look wrong.

Limitations. Probes come from a single 7B model; larger or format-tuned models may preserve structure better. I sample 30 per cell, adequate for the effect sizes reported but not for small ones, which is why the SST-2 cells are inconclusive rather than negative. Refusal detection is a surface pattern match at the start of a generation; it will miss a refusal phrased unusually, so 6.2% is a lower bound. Similarity is measured with one sentence encoder, and encoder choice shifts how severe the 0.75 threshold is; the referee check is a partial remedy but assumes the two architectures fail independently, which they only approximately do, since both are pretrained transformers fine-tuned on the same data. The task-specific prompt for SST-2 was written by me in the shape of the originals; a different phrasing might move the effect. Finally, the three failures are demonstrated on classification; whether they take the same form in generative red-teaming is untested here.

10 Conclusion

An automated red-teaming result is a claim about a target model, but it is produced by a second model whose output may not be a valid test at all. I measured four ways that happens. A prompt can raise the flip rate by changing the input's true label: an independent referee reproduces 86–87% of the best-scoring prompt's successes, and once the flip is required to be target-specific that prompt goes from winning two to one to losing, 16/180 against 29/180, or 12/180 against 25/180 if the two models must also agree on the original. A probe can fail to be a well-formed input, which understates it, 12.5% against 41.7%. The generator can refuse, at a rate that tracks the very prompt under comparison, and have its refusal scored as a probe. And a pipeline can report a number computed from something other than what it generated, which is undetectable from the outputs alone.

Underneath, LLM rewriting is a trade rather than an improvement: it makes attacks dramatically more fluent, median perplexity 22.9 against 522, and destroys more than half of them. None of this makes the approach unusable. It makes it measurable, which requires reporting similarity beside flip rate, a referee model's agreement beside similarity, well-formedness beside both, the generator's refusal rate per condition, and how many generations the target model actually saw.

Reproducibility

Code, the stored probes, and every result table in this paper are available at `https://github.com/yhj3/counterfactual-textattack`. The repository documents which components are original and which were reconstructed, and includes the audit described in Section 8.

References

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is BERT really robust? A strong baseline for natural language attack on text classification and entailment. In *AAAI*, volume 34, pages 8018–8025.

John X. Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. TextAttack: A framework for adversarial attacks, data augmentation, and adversarial training in NLP. In *EMNLP: System Demonstrations*, pages 119–126.

Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. 2019. Generating natural language adversarial examples through probability weighted word saliency. In *ACL*, pages 1085–1097.

Hugo Touvron, Louis Martin, Kevin Stone, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. HotFlip: White-box adversarial examples for text classification. In *ACL (Short Papers)*, pages 31–36.

Divyansh Kaushik, Eduard Hovy, and Zachary C. Lipton. 2020. Learning the difference that makes a difference with counterfactually-augmented data. In *ICLR*.