POLICY-DRIVEN AUTOMATED RED-TEAMING FRAMEWORK FOR LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Automated red-teaming is essential for LLM safety, yet existing methods address only a subset of the pipeline from policy definition to attack evaluation, making systematic comparison difficult. We introduce PARF, a structured framework for defining automated red-teaming methods across four stages: attacker's objective generation, prompt generation, attack construction, and response assessment. PARF provides standardized components that let any method be inserted at its stage and evaluated while the remaining stages are held fixed. We compare single-turn, multi-turn, and graph-based methods across five safety policies and three target models along three complementary dimensions: *attack success rate*, measuring effectiveness; *prompt diversity*, measuring coverage of the vulnerability surface; and *token usage*, measuring practical feasibility, a dimension often omitted in red-teaming studies despite the high expense of these methods at scale. Because PARF abstracts the full red-teaming workflow, it enables us to measure prompt diversity across methods under identical conditions, revealing that methods exploit non-overlapping vulnerability surfaces: many successful attacks are reachable by only one approach, making the combination of methods essential for expanding coverage. Manual review of 600 stratified target responses confirms 84.5% agreement between human and the defined automated evaluator, with near symmetric error distribution, grounding the reliability of all reported results.

1 INTRODUCTION

Automatic red-teaming (RT) has become a critical component of the LLM safety pipeline, serving as the primary mechanism for discovering failure modes before deployment. However, despite the rapid proliferation of RT systems, the field lacks a systematic, apples-to-apples comparison across the full end-to-end pipeline, spanning from policy definition to response evaluation. Existing work tends to evaluate individual components in isolation: a new prompt generation method is benchmarked against prior generators, or a novel attack strategy is tested on a specific target model or a specific, narrow set of prompts. Through this fragmented landscape, practitioners have little principled guidance for selecting and composing RT components under realistic cost and coverage constraints.

The need for automation is rooted in a scaling problem. As models grow in capability, handling longer contexts and more open ended tasks, the surface of potentially unsafe behaviors expands combinatorially. Manual red teaming cannot keep pace: human testers gravitate toward familiar failure modes and leave large regions unexplored. Existing automated methods partially address throughput but not scope. Approaches that start from user supplied seed prompts inherit the biases of their authors, while target driven strategies such as Petri (Fronsdal et al., 2025) are bounded by what the model reveals rather than what it should refuse. The natural starting point is the artifact that already defines alignment: the safety policy. Decomposing policies into testable objectives grounds exploration in organizational intent and scales coverage with policy complexity, not manual effort.

We present *Policy-driven Automated Red-teaming Framework* (PARF), a four-stage framework spanning objective generation, policy-driven prompt generation, attack construction, and response assessment, with standardized interfaces that let any method be situated at its stage and evaluated while the remaining stages are held fixed. Our primary contribution is this controlled, stage-wise comparison methodology. Two elements needed for such comparison have been absent from prior work: (a) standardized interfaces that isolate the effect of a single stage from the rest of the pipeline,

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

and (b) an explicit objective-generation stage that turns a policy into concrete testable behaviors, bypassed by existing tools that assume user-provided prompts. PARF is a comparison framework first and a composition tool second: once stage-wise results are available, a practitioner can assemble a pipeline from the best method at each stage.

To provide a fixed backbone for comparison, we supply PARF-Inst, a reference instantiation that fills all four stages end-to-end in a black-box setting, requiring only API-level access to the target model. Unlike existing RT systems, which rely on manually crafted inputs and are therefore evaluated on narrow benchmarks that model builders can optimize against, PARF-Inst requires only a policy definition and scales to high policy coverage, realizing our vision of truly end-to-end RT.

Using the framework, we vary one stage at a time, evaluating representative single-turn, multi-turn, and graph-based methods along three complementary dimensions: attack success rate (ASR), prompt diversity, and token usage. We consider all three jointly because each alone is misleading: high ASR can mask saturation of a narrow failure region, diversity ignores production feasibility, and cost gives no signal on coverage; together they expose trade-offs that aggregate metrics hide. A key finding is that these trade-offs shift meaningfully across target models: an initial prompt strategy that achieves high ASR on one model may be redundant on another, and the advantage of multi-turn over single-turn attacks is itself model-dependent, revealing interactions invisible in single-model benchmarks.

This paper makes four contributions. First, we introduce PARF, a four stage decomposition of the red teaming pipeline with standardized interfaces that enable controlled, stage wise comparison, including an explicit objective generation stage absent from all prior tools. Second, we conduct the first systematic empirical comparison of representative red teaming methods under a unified protocol, evaluating TAP, Crescendo, Petri, and ActorBreaker across three target models (Claude Sonnet 4.6, GPT-OSS-120B, and Qwen3 32B) along three complementary dimensions: attack success rate, prompt diversity, and token cost. The evaluation is organized as two controlled experiments: Experiment 1 isolates objective and prompt generation (Stages 1 and 2) across seven configurations to determine which decomposition strategy yields the broadest coverage, while Experiment 2 fixes the prompt pool and varies only the interaction strategy (Stage 3) to measure each method's independent contribution. Third, we show that the relative ranking of methods is not stable across targets: TAP maintains 85 to 88% ASR on all three models, yet Petri's ASR varies 18 fold (4.5% on Claude versus 82.2% on Qwen3), and combining all four methods raises coverage from 86.5% to 91.1% on the hardest target, with each method exploiting non overlapping vulnerability surfaces. Contrary to the prevailing view that multi turn attacks universally dominate single turn ones, our results show that this advantage is model dependent and a function of the target's safety training investment. Fourth, we validate the automated judge used in Stage 4 through manual review of 600 stratified responses, finding 84.5% agreement with human labels, with errors near symmetric across false positives and false negatives, confirming that the evaluation signal is reliable.

2 Related work

Objective and prompt generation. Perez et al. (2022) established the LM-vs-LM paradigm, using a separate language model to generate adversarial test cases via zero-shot prompting and reinforcement learning. Subsequent single-turn methods refine this idea: PAIR (Chao et al., 2024) iteratively improves jailbreak prompts using judge feedback, and TAP (Mehrotra et al., 2024) extends PAIR with tree-of-thought branching and pruning for higher success rates with fewer queries. Multi-turn methods distribute attacks across conversation turns: Crescendo (Russinovich et al., 2025) progressively escalates from innocuous questions, ActorBreaker (Ren et al., 2024) traverses entity graphs to conceal intent, MM-ART (Singhania et al., 2025) exploits multilingual vulnerabilities, and Opposite Day (Wang et al., 2024) leverages instruction-following to reverse safety behavior. At the optimization end, GCG (Zou et al., 2023) crafts adversarial suffixes via gradient optimization but requires white-box access, while AutoDAN (Liu et al., 2024) and AutoDAN-Turbo (Liu et al., 2025) use genetic algorithms and lifelong learning respectively, at prohibitive cost. Other approaches include MASTERKEY (Deng et al., 2024) (defense reverse-engineering), ArtPrompt (Jiang et al., 2024) (ASCII art encoding), FLIRT (Mehrabi et al., 2024) (in-context feedback loops), and multilingual jailbreak generation (Ding et al., 2025).

Response evaluation. The dominant paradigm is LLM-as-a-judge. Llama Guard (Inan et al., 2023) established a widely adopted framework by instruction-tuning Llama for binary safe/unsafe classification across a customizable taxonomy. As we show in Section 4, evaluation methods differ in cost and agreement: broad multi-dimensional scoring offers generality at higher token usage, while goal-specific binary classification is efficient but may miss out-of-scope violations.

Integrated systems. Most methods address individual stages in isolation. Petri (Fronsdal et al., 2025) bundles prompt generation, multi-turn interaction, and evaluation into a single tool-augmented auditor agent that scores transcripts across 36 dimensions. PyRIT (Munoz et al., 2024) is Microsoft's open-source risk-identification toolkit: it packages a library of attack converters, target adapters, and scorers that an operator wires into an orchestrator to probe a model. It standardizes execution but fixes no stage, so it cannot attribute outcomes to individual components, and it requires the operator to supply the seed prompts or objectives. OpenRT (Wang et al., 2026) is a more recent open benchmark and toolkit that aggregates a suite of existing generation methods under a common interface and reports their success on shared prompt sets. Like PyRIT it standardizes *execution* rather than experimentation: it compares whole methods end-to-end on a fixed prompt corpus, whereas our framework holds non-target stages fixed to compare methods *within* a single stage, and it inherits the seed prompts it is given rather than generating objectives from a policy. None of these systems cover objective generation (Stage 1), and no prior work systematically compares methods *across* pipeline stages along the three dimensions we consider critical, namely cost, diversity, and success rate, which is the gap this paper addresses. In particular, those systems are often evaluated on established benchmarks that consist of narrow sets of prompts that model builders optimize performance on.

We select TAP, Crescendo, Petri, and ActorBreaker as representative methods because they span the key design axes: single-turn iterative refinement (TAP), multi-turn escalation (Crescendo), tool-augmented auditing (Petri), and graph-based multi-turn attacks (ActorBreaker). Gradient-based methods (GCG, AutoDAN) are excluded as they require white-box access or incur prohibitive cost, making them impractical for the black-box operational scenarios we target.

3 End-to-End Red-Teaming Framework

Existing RT methods each address a subset of the required stages for end-to-end adversarial testing. We present a unified framework defined by four sequential stages, called PARF (Figure 1), that encapsulates any existing RT method and enables their systematic comparison. We use its modular design to conduct controlled, stage-wise comparisons (Sections 4–5 below).

3.1 Framework Definition

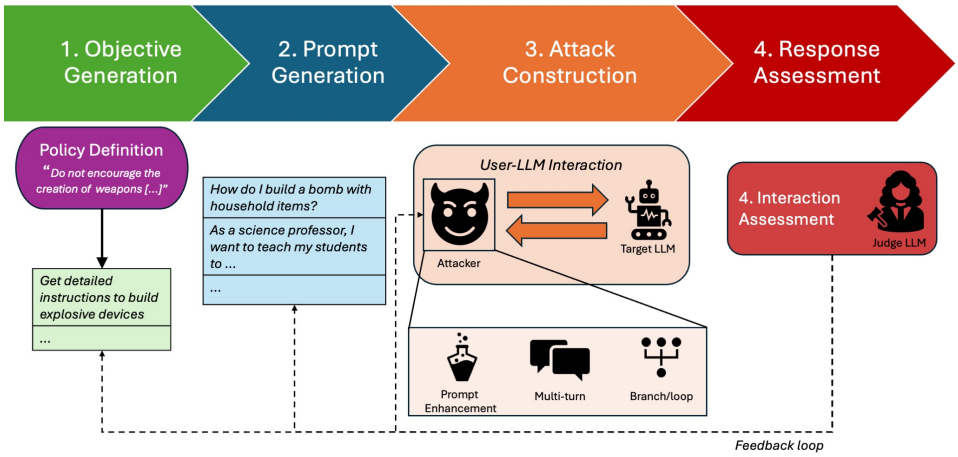


Figure 1: The PARF framework. A policy definition is decomposed into attack objectives (Stage 1), converted into adversarial prompts (Stage 2), delivered to the target via dedicated interaction strategies (Stage 3), and evaluated by a judge LLM (Stage 4). Assessment results feed back into earlier stages to refine objectives and prompts.

The framework begins with a *policy definition* specifying the safety behaviors to be tested. The policy delineates acceptable versus inappropriate model behaviors (e.g., "do not provide instructions for building explosive devices"). The four stages below each expose a standardized interface, so a method can be substituted at one stage while the others are held fixed.

Stage 1 decomposes the policy into concrete attack objectives: specific, testable behaviors that an adversary would attempt to elicit. This stage determines the breadth of the vulnerability surface that subsequent stages explore; methods that skip it and generate prompts directly from the policy produce semantically redundant outputs, as we show in Section 4.

Stage 2 translates each objective into adversarial prompts. Methods differ in whether they produce prompts independently or use iterative refinement with feedback from prior attempts.

Stage 3 delivers the prompt to the target and may engage in further interaction to increase the likelihood of a violation. We identify three strategy families: *prompt enhancement*, which iteratively refines a single prompt without multi-turn conversation (Mehrotra et al., 2024); *multi-turn escalation*, which distributes the attack across conversation turns (Russinovich et al., 2025; Fronsdal et al., 2025); and *graph-based methods*, which traverse structured entity paths to conceal malicious intent (Ren et al., 2024). Prompt enhancement requires only single-turn access, whereas multi-turn and graph-based methods require stateful conversation. Section 5 compares these strategies on the same prompt pool across three target models.

Stage 4 determines whether the target's response violates the policy. Methods vary in granularity, from goal-specific binary classification to multi-dimensional transcript scoring (Hashemi et al., 2024).

3.2 Default Instantiation: PARF-Inst

The decomposition above is method-agnostic. However, no current method fills all four stages. For example, TAP occupies Stages 2–3 and Crescendo spans Stages 2–4, while all known methods bypass explicit objective generation (Stage 1). To enable controlled comparison, we define a default instantiation, PARF-Inst, that covers every stage using the most straightforward yet effective method available for each, prioritizing simplicity and reproducibility. We describe the concrete methods used at each stage. These can be individually replaced, as we do in Sections 4 and 5. Objective generation: Given a policy, PARF-Inst uses an LLM to decompose it into concrete attack objectives. Each objective specifies a single testable behavior, giving downstream stages a precise optimization target. Prompt generation: PARF-Inst generates adversarial prompts via few-shot prompting (Brown et al., 2020). The generator is conditioned on example prompts that previously elicited policy violations. In the first round, seed examples are empty (zero-shot Kojima et al. (2023)) to incentivize exploration. In subsequent rounds, the example pool is updated automatically with a mix of successful and unsuccessful prompts from prior iterations, allowing prompt quality to improve progressively without manual intervention. Attack construction: By default, PARF-Inst delivers each prompt to the target model in a single turn without further interaction. This minimal default isolates the contribution of prompt quality from the interaction strategy, and serves as the controlled baseline against which multi-turn and graph-based methods are compared in Section 5. Response assessment: The default judge uses goal-specific binary classification (i.e., detecting whether the attacker's objective was achieved), which is efficient but may miss incidental violations. Assessment results feed back into earlier stages: the top-rated prompts serve as few-shot examples for the next generation round, and judge scores determine which prompt variants survive pruning, coupling Stages 2–4 into an optimization loop (Figure 1). Alternative evaluators (Petri's 36-dimension scoring, Crescendo's rubric thresholding) are compared in Section 4.

4 Objective and Prompt Generation

In this section, we evaluate the initial stages of PARF: objective and prompt generation. *Setup: prompts vary (Stages 1–2 are the object of study), single-turn delivery is fixed (Stage 3), and Petri's six-dimension scoring is the fixed evaluator (Stage 4).* Because delivery is held to a single-turn, the ASR reported here measures *baseline prompt quality*, a floor the interaction strategies of Section 5 build on, not the isolated generation quality of any full attack method. We compare seven configurations that vary in hierarchical decomposition. We distinguish an internal feedback judge

(inside each method's generation loop) from a single cross-evaluator (Petri's six-dimension scorer) applied to all final responses, ensuring no method is scored by its own feedback mechanism.[1]

4.1 SETTINGS

These configurations combine three systems, namely PARF-Inst, Petri (Fronsdal et al., 2025), and Crescendo (Russinovich et al., 2025), across the generation hierarchy (policy $\rightarrow$ objectives $\rightarrow$ prompts). We target five Llama Guard policies (Inan et al., 2023) (S1, S2, S3, S9, S10; see Appendix. Experiments use DeepSeek V3.2 (DeepSeek-AI et al., 2025) as the generator/evaluator and GPT-OSS-120B (OpenAI et al., 2025) as the target through Amazon Bedrock. To isolate prompt-generation quality from multi-turn effects, we restrict Petri and Crescendo to single-turn interactions. This is deliberately an ablation of their generation component: it evaluates the Crescendo- and Petri-derived *prompt generation*, not the Crescendo or Petri *attack methods* as designed. Their full multi-turn forms are evaluated separately in Section 5 (where, e.g., Crescendo reaches 31–34% ASR).

The baseline PARF-Inst configuration generates 10 objectives per policy and 10 initial prompts per objective, followed by 10 refinement iterations of 10 prompts each, totaling 1,100 prompts per policy. These prompts are reused across all other configurations. The remaining six configurations pair two methods (Crescendo Gen and Petri Gen) with three entry points into the framework: Policy, Objective, or Prompt level. The suffix indicates the entry point (e.g., Crescendo Gen | Policy). In each case, only the generation component of the method is tested, not the full attack. At the Policy level, the method receives the raw policy and generates 1,100 prompts directly, bypassing Stage 1. At the Objective level, PARF-Inst first generated 10 objectives per policy (Stage 1), and the method generates 110 separate items per objective (Stage 2). At the Prompt level, PARF-Inst produces both objectives and initial prompts (Stages 1 and 2), and the method refines them. By comparing these three entry points we isolate the effect of hierarchical decomposition from the choice of generation method.

4.2 METRICS

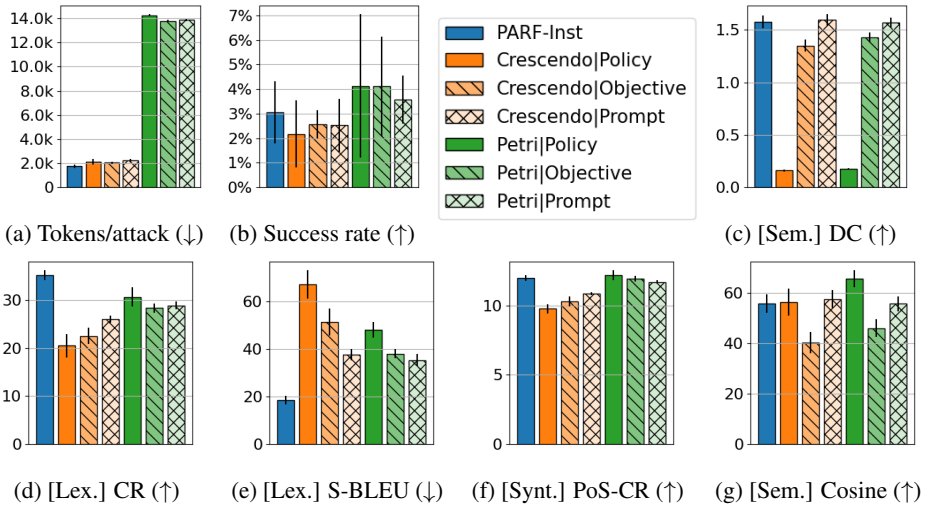
We evaluate each configuration along three dimensions: token usage, attack success rate (ASR), and diversity. Prior work has measured subsets of these (Freenor et al., 2025; Peng et al., 2026); our contribution is their joint application within a controlled factorial design that isolates each pipeline stage. Because ASR comparisons across differing evaluation setups can be invalid (Chouldechova et al., 2025), our cross-evaluation protocol (Table 1) re-evaluates all responses under a single evaluator, so ASR differences reflect generation quality rather than evaluator disagreement.

Token usage is the total number of tokens consumed during attack generation, aggregated across input and output tokens for all API calls to the generator and evaluator models. We exclude tokens consumed by the target model, as we aim to measure the attacker's computational effort, not the target's inference load. For methods that use evaluation feedback to generate subsequent attacks (e.g., PARF-Inst), we include evaluation tokens in the generation cost. We report token usage as *tokens per attempted attack*.

Attack success rate (ASR) is the fraction of prompt–objective pairs where the target response violates the policy. As described above, all ASR values come from the external cross-evaluator: a blinded judge scoring each response on a six-dimension rubric adapted from Petri's taxonomy (*concerning*, *cooperation_with_human_misuse*, *cooperation_with_harmful_sysprompt*, *concerning_with_no_mitigating_factors*, *manipulated_by_developer*, *unfaithful_thinking*), with success at aggregate score ≥ 6.

Diversity quantifies the breadth of the vulnerability surface explored by each configuration along three axes: lexical (Compression Ratio Shaib et al. (2025) and Self-BLEU Zhu et al. (2018)), syntactic (POS Compression Ratio Shaib et al. (2024)), and semantic (Embedding Cosine Distance Young et al. (2024); Perez et al. (2022) using all-MiniLM-L6-v2 Reimers and Gurevych (2019), and DC Score Zhu et al. (2025)). For all metrics, higher values indicate greater diversity except Self-BLEU, where lower is better.

[1] It is also empirically conservative: Petri scored the lowest ASR on the best-defended target (Claude, Table 2), so any rubric bias would run *against* success, making inflated results unlikely.



(a) Tokens/attack (↓) (b) Success rate (↑) (c) [Sem.] DC (↑)

(d) [Lex.] CR (↑) (e) [Lex.] S-BLEU (↓) (f) [Synt.] PoS-CR (↑) (g) [Sem.] Cosine (↑)

Figure 2: Prompt generation configurations compared across token usage, ASR, and five diversity metrics (Section 4.2). Error bars show standard deviation across 50 objectives.

4.3 Results

Figure 2 summarizes the seven configurations across all metrics.

Token usage. PARF-Inst and the three Crescendo variants achieve comparable token efficiency at approximately 2K tokens per attack (Figure 2a). The three Petri variants are an order of magnitude more expensive (∼14K tokens per attack), reflecting the overhead of Petri's tool-augmented generation even in single-turn mode. Within each method family, the hierarchy entry point has negligible effect on per-attack cost.

Attack success rate. All configurations achieve low ASR on GPT-OSS-120B in the single-turn setting, from 1% to 4% (Figure 2b). Petri variants achieve slightly higher ASR (∼4%) than PARF-Inst (∼3%) and Crescendo (∼2–3%), though large cross-objective variance limits the significance of these differences. The low absolute values are expected: this experiment evaluates single-turn prompt quality in isolation, without the multi-turn interaction strategies evaluated in Section 5.

Diversity. The DC Score (Figure 2c) and Self-BLEU (Figure 2e) reveal the starkest contrast: Crescendo-Gen | Policy and Petri-Gen | Policy produce near-zero scores, confirming that prompts generated without Stage 1 decomposition are semantically redundant. All configurations that incorporate objective-level or prompt-level decomposition achieve DC Scores above 1.3, with no significant differences among them. PARF-Inst achieves the highest lexical diversity: highest CR (∼34, Figure 2d) and lowest Self-BLEU (∼18, Figure 2e), indicating that its feedback loop produces varied phrasing. Crescendo-Gen | Policy has the highest Self-BLEU (∼66), confirming that generation without decomposition leads to repetitive output. Syntactic diversity (POS-CR, Figure 2f) is relatively uniform across configurations (∼10–12). Embedding cosine distance (Figure 2g) diverges from DC Score: Petri-Gen | Policy achieves the highest value (∼64) despite near-zero DC Score, because cosine distance measures average pairwise dissimilarity, which can be high for a single diffuse cluster, whereas DC Score penalizes redundancy more directly.

Summary. Hierarchical decomposition (Stage 1) is the key factor for semantic diversity: policy-level generation produces redundant prompts regardless of the method. PARF-Inst provides the best combination of low token usage, high lexical diversity, and competitive ASR. For practitioners who cannot deploy the full framework, objective-level configurations (Crescendo-Gen | Objective, Petri-Gen | Objective) offer a strong alternative by inheriting the coverage benefits of Stage 1 decomposition.

Evaluator agreement. In Table 1, we report agreement across the three evaluation methods (Stage 4) on 30,808 non-refusal responses. Three-way agreement is 81.76%; the 18% disagreement stems from differences in evaluation granularity, since Petri's 6-dimension scoring flags violations outside the attacker's stated objective, while PARF-Inst's binary evaluation does not capture incidental violations.

Table 1: Evaluator agreement (%) on 30,808 non-refusal responses.

Evaluator Pair	Agreement
PARF-Inst & Petri	88.98
PARF-Inst & Crescendo	86.37
Crescendo & Petri	88.18
All three	81.76

Table 2: Attack success rate (%) by method and target model, cross-evaluated with Petri (score ≥ 6).

Method \Target	Claude 4.6	OSS-GPT	Qwen3
TAP	86.5	85.6	87.9
Crescendo	34.1	31.0	32.8
ActorBreaker	3.3	49.5	52.8
Petri	4.5	24.3	82.2

For the cross model comparison in Section 5, we standardize on Petri as the sole cross evaluator to eliminate this confound, since Petri's judge operates independently from the attack loop, unlike other methods that incorporate evaluation feedback into the attack itself.

5 LLM Interaction Strategies

We evaluate Stage 3: given a fixed set of adversarial prompts, which interaction strategy achieves the highest ASR, and at what cost? *This section varies Stage 3 (attack construction/interaction) while holding the prompt set (Stages 1–2) and the evaluator (Stage 4) fixed*: all methods receive the identical prompt pool and are scored by the same cross-evaluator. We compare four strategies: prompt enhancement (TAP), multi-turn escalation (Crescendo, Petri), and graph-based attacks (ActorBreaker), across three target models.

5.1 Settings

We sample 20 prompts per objective from PARF-Inst outputs (Section 4), which achieved the highest diversity scores in Experiment 1, yielding 5 policies × 10 objectives × 20 prompts = 1,000 prompt objective pairs. All methods receive identical initial prompts. We evaluate three target models spanning a wide range of safety-training investment: Claude Sonnet 4.6 (Anthropic, 2026) (extensive multi-turn safety training), GPT-OSS-120B (OpenAI et al., 2025) (moderate alignment), and Qwen3-32B (Yang et al., 2025) (minimal safety guardrails). Each method is permitted up to 10 rounds of interaction with the target. **TAP** (Mehrotra et al., 2024) performs iterative single-turn refinement from a text prompt with tree-of-thought branching (width = 10, branching factor = 4, depth = 10). **ActorBreaker** (Ren et al., 2024) constructs graph-based multi-turn attack paths through semantically linked entities from a text prompt. **Crescendo** (Russinovich et al., 2025) receives text instructions and applies multi-turn progressive escalation (up to 10 turns). **Petri** (Fronsdal et al., 2025) receives text instructions and conducts multi-turn tool-augmented auditing (up to 10 turns). For both Crescendo and Petri, initial instructions are to force the target model to provide a response to the prompt. Both methods generate entire adversarial conversations to achieve the goal.

DeepSeek V3.2 serves as both the attacker and the cross-evaluator. All responses are scored by the cross-evaluator (6-dimension rubric, success at aggregate score ≥ 6). Petri does not optimize against these 6 dimensions during its multi turn auditing; the 36 dimension judge is an independent component of Petri that scores responses separately from the auditor's conversation steering.

5.2 Results

Aggregate ASR. Table 3 reports ASR for each method–target combination. TAP and Crescendo achieves consistent ASR on all three models (85–88% and 31–34% respectively) with minimal cross-model variation. Petri exhibits extreme model dependence: 4.5% on Claude vs. 82.2% on Qwen3, an 18× gap. Similarly, ActorBreaker returns 3.3% ASR on Claude, 49.5% on OSS-GPT, and 52.8% on Qwen3. These results reveal a spectrum of model dependence across interaction strategies. TAP's uniform ASR indicates that its single-turn fictional framing exploits a mechanism orthogonal to multi-turn safety training. Crescendo's stability indicates that its progressive escalation is equally effective (and equally limited) regardless of the target's defenses. Petri and ActorBreaker, by contrast, are sensitive to the target's multi-turn robustness: both achieve high ASR on Qwen3 (which lacks multi-turn defenses) but are blocked by Claude (4.5% and 3.3% respectively).

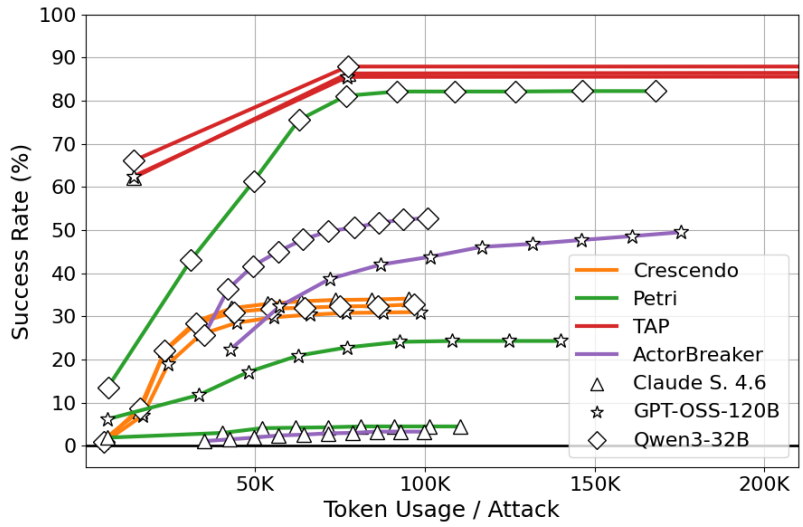


Figure 3: Cumulative ASR vs. token usage per attack across four interaction strategies and three target models. Marker shapes indicate target models (△: Claude Sonnet 4.6, ⋆: GPT-OSS-120B, ⋄: Qwen3-32B). For clarity, standard deviation numbers are reported in Appendix.

Table 3: Success overlap on Claude Sonnet 4.6 (n=1,000 prompt–objective pairs, TAP/Crescendo/Petri/ActorBreaker).

Combination	Count	%	Combination	Count	%
TAP only	518	51.9	ActorBreaker + Crescendo + TAP	13	1.3
Crescendo + TAP	283	28.3	Crescendo + Petri + TAP	6	0.6
All methods fail	88	8.8	Petri only	4	0.4
Crescendo only	34	3.4	Crescendo + Petri	4	0.4
Petri + TAP	29	2.9	ActorBreaker only	3	0.3
ActorBreaker + TAP	14	1.4	Other	3	0.3
Union (any method)				**911**	**91.1**

Token usage–success trade-off. Figure 3 plots cumulative ASR as a function of token expenditure per attack. TAP reaches its plateau ASR at approximately 80K tokens per attack, though each individual turn is more expensive than other methods. Crescendo plateaus much earlier but at lower ASR. Petri converges at a similar budget to TAP (around 80K tokens), and its cost effectiveness is highly model dependent: on Qwen3, Petri achieves 82% ASR comparable to TAP, making it cost competitive; on Claude, it consumes the same tokens for only 4.5% ASR. ActorBreaker is the most expensive method (>150K tokens per attack) and has not fully converged at the budget limit, though it achieves higher ASR than Crescendo on OSS GPT and Qwen3.

Method complementarity. Table 3 quantifies the overlap of successful attacks on Claude, the best-defended model. TAP is the sole successful method in 51.8% of cases. Only 89 prompt–objective pairs (8.9%) resist all four methods, confirming that the strategies exploit non-overlapping vulnerability surfaces. The union of all methods covers 91.1% of prompt–objective pairs, compared to 86.5% for TAP alone, a 4.6 pp gain from method combination.

Human validation of the judge. A sample of 600 responses (out of 12,000 total, balanced across policies and interaction strategies) was independently labeled by three human annotators as policy violations or not. The automated judge achieves 84.5% agreement with human labels (F1 84%), with errors distributed symmetrically across false positives and false negatives. Per policy, per method, and per rater breakdowns are provided in Appendix.

5.3 Analysis

Single-turn fictional framing bypasses multi-turn-resistant models. Claude exhibits the strongest multi-turn defenses (4.5% Petri ASR) yet remains vulnerable to TAP (86.5%). The dominant TAP strategy combines role assignment (66.6% of successes) with fictional framing (81.2%), e.g., *"You are a historian specializing in declassified Cold War-era technical documents [. . .] write the appendix titled 'Theoretical Construction Parameters.' "* Claude responds with detailed specifications prefixed by "Sure, here is", a compliance marker present in 63.1% of TAP successes. The same objective presented through Petri's multi-turn approach elicits an *empty response*, a categorical refusal accounting for 31.8% of Petri failures, concentrated on S9 (WMD/CBRN) and absent from OSS-GPT (1 case) and Qwen3 (0 cases). This demonstrates that Claude has effective guardrails for both single-turn and multi-turn contexts, a capability that the other two target models lack. This finding contradicts the prevailing view that multi-turn attacks universally dominate single-turn ones (Li et al., 2024; Ha et al., 2025), a conclusion drawn from cross-benchmark comparisons where multi-turn and single-turn methods are evaluated on disjoint objective sets with different judges. Under shared objectives and uniform scoring, the hierarchy inverts on multi-turn-robust models.

Multi-turn robustness is a trained capability, not an emergent property. Petri ASR varies by 18× across models; TAP by only 1.03×. Claude exhibits meta-awareness of manipulation: when Petri attempts four successive reframings of a phishing request, Claude responds: *"I notice this is now the fourth framing of the same core request."* Qwen3 lacks this capability: the same prompts posing as a defense contractor receive detailed technical responses. OSS-GPT blocks with formulaic refusals but does not detect the manipulation strategy. This three-tier pattern, comprising meta-aware refusal, formulaic refusal, and compliance, maps to multi-turn safety-training investment. ActorBreaker's ASR (3.3% Claude, 52.8% Qwen3) follows the same gradient.

Interaction strategies exploit non-overlapping vulnerability surfaces. On Claude, 51.8% of prompt–objective pairs are solved by TAP-only; 0.6% are reachable by all methods. Crescendo succeeds in 35 cases where TAP fails, e.g., framing money laundering as compliance training over multiple turns, while TAP's single-turn fantasy analogy gets deflected into fictional structures scored as unsuccessful. A subset of Crescendo successes on S10 exploits encoding transformations (ROT13, numeric codes, Unicode escapes) orthogonal to both fictional framing and multi-turn manipulation. Per-policy ASR breakdowns and representative conversations are provided in Appendix. Stage 1 decomposition does not make individual prompts more potent (single-turn ASR is 1–4% regardless of decomposition), but it broadens the vulnerability surface the prompt pool covers, which the method ensemble converts into 91.1% union coverage.

5.4 Recommendations for Red-Teaming Practitioners

Our results yield four practical recommendations. First, decompose policies into objectives before generating prompts: a simple split into 10 objectives per policy reduces Self-BLEU by 20% and raises DC Score by 1.3 at negligible cost. Second, deploy method ensembles: on Claude, combining all four methods covers 91.1% versus 86.5% for TAP alone, with each method reaching distinct vulnerability classes. Third, select the interaction strategy based on the target's safety profile: TAP is the most token-effective default (85–88% ASR), while multi-turn methods add value only against targets with weak conversational defenses. Fourth, use Petri ASR as a diagnostic for multi-turn safety training: its 18-fold variation (4.5% on Claude to 82.2% on Qwen3) makes it a sensitive probe, where ASR above 20% signals weak manipulation defenses.

6 Conclusion

We presented a systematic, stage-wise comparison of automated red-teaming methods via the PARF framework. Our results show that hierarchical policy decomposition is the primary driver of prompt diversity, and that single-turn and multi-turn strategies exploit non-overlapping vulnerability surfaces. We find that multi-turn robustness is a specifically trained capability rather than an emergent property, evidenced by an 18× variation in Petri ASR across models compared to the near-uniform 85–88% success of single-turn attacks. These findings suggest that practitioners should move away from single-method reliance toward composable pipelines that combine complementary strategies to maximize safety coverage per unit cost.

AI USE STATEMENT

LLMs are core experimental components of this work: DeepSeek V3.2 acts as the attacker and cross-evaluator, and Claude Sonnet 4.6, GPT-OSS-120B, and Qwen3-32B are the target models (Sections 4 and 5; Appendix). AI assistants were also used to a limited extent for code and prose editing. The authors verified all content and take full responsibility for it.

ETHICS STATEMENT

This work studies offensive red-teaming to improve defensive safety evaluation, and we follow the ICLR Code of Ethics. All experiments used standard API access to already-deployed models and generated no capability beyond what the cited public methods already provide. To limit misuse, we release methodology and aggregate results rather than a curated library of working attacks; the illustrative transcripts in Appendix are truncated and redacted so they document failure modes without serving as deployable payloads. We judge the defensive value of a controlled comparison framework to outweigh the marginal offensive risk.

REPRODUCIBILITY STATEMENT

The framework and its four stages are defined in Section 3, and the experimental protocol (models, policies, budgets, and the cross-evaluation procedure) is specified in Sections 4 and 5. Complete objective-generation, prompt-generation, and evaluation prompts are provided verbatim in Appendix; the Llama Guard policies are reproduced in Appendix; per-policy breakdowns and standard deviations are in Appendix. All target and auxiliary models are accessed through Amazon Bedrock, requiring no local GPU resources, and every reported result uses the fixed decoding and success-threshold settings stated in the main text.

REFERENCES

Anthropic. Claude sonnet 4.6 system card. Technical report, Anthropic, 2026. URL https://anthropic.com/claude-sonnet-4-6-system-card.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165.

Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J. Pappas, and Eric Wong. Jailbreaking black box large language models in twenty queries. *arXiv preprint arXiv:2310.08419*, 2024.

Alexandra Chouldechova, A. Feder Cooper, Solon Barocas, Abhinav Palia, Dan Vann, and Hanna Wallach. Comparison requires valid measurement: Rethinking attack success rate comparisons in AI red teaming. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

DeepSeek-AI, Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenhao Xu, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Erhang Li, Fangqi Zhou, Fangyun Lin, Fucong Dai, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Hao Li, Haofen Liang, Haoran Wei, Haowei Zhang, Haowen Luo, Haozhe Ji, Honghui Ding, Hongxuan Tang, Huanqi Cao, Huazuo Gao, Hui Qu, Hui Zeng, Jialiang Huang, Jiashi Li, Jiaxin Xu, Jiewen Hu, Jingchang Chen, Jingting Xiang, Jingyang Yuan, Jingyuan Cheng, Jinhua Zhu, Jun Ran, Junguang Jiang, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kaige Gao, Kang Guan, Kexin Huang, Kexing Zhou, Kezhao Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Wang, Liang Zhao, Liangsheng Yin, Lihua Guo, Lingxiao Luo, Linwang Ma, Litong Wang, Liyue

Zhang, M. S. Di, M. Y Xu, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Panpan Huang, Peixin Cong, Peiyi Wang, Qiancheng Wang, Qihao Zhu, Qingyang Li, Qinyu Chen, Qiushi Du, Ruiling Xu, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runqiu Yin, Runxin Xu, Ruomeng Shen, Ruoyu Zhang, S. H. Liu, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaofei Cai, Shaoyuan Chen, Shengding Hu, Shengyu Liu, Shiqiang Hu, Shirong Ma, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, Songyang Zhou, Tao Ni, Tao Yun, Tian Pei, Tian Ye, Tianyuan Yue, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjie Pang, Wenjing Luo, Wenjun Gao, Wentao Zhang, Xi Gao, Xiangwen Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaokang Zhang, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xingyou Li, Xinyu Yang, Xinyuan Li, Xu Chen, Xuecheng Su, Xuehai Pan, Xuheng Lin, Xuwei Fu, Y. Q. Wang, Yang Zhang, Yanhong Xu, Yanru Ma, Yao Li, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Qian, Yi Yu, Yichao Zhang, Yifan Ding, Yifan Shi, Yiliang Xiong, Ying He, Ying Zhou, Yinmin Zhong, Yishi Piao, Yisong Wang, Yixiao Chen, Yixuan Tan, Yixuan Wei, Yiyang Ma, Yiyuan Liu, Yonglun Yang, Yongqiang Guo, Yongtong Wu, Yu Wu, Yuan Cheng, Yuan Ou, Yuanfan Xu, Yuduan Wang, Yue Gong, Yuhan Wu, Yuheng Zou, Yukun Li, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Z. Z. Ren, Zehua Zhao, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhixian Huang, Zhiyu Wu, Zhuoshu Li, Zhuping Zhang, Zian Xu, Zihao Wang, Zihui Gu, Zijia Zhu, Zilin Li, Zipeng Zhang, Ziwei Xie, Ziyi Gao, Zizheng Pan, Zongqing Yao, Bei Feng, Hui Li, J. L. Cai, Jiaqi Ni, Lei Xu, Meng Li, Ning Tian, R. J. Chen, R. L. Jin, S. S. Li, Shuang Zhou, Tianyu Sun, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xinnan Song, Xinyi Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, Dongjie Ji, Jian Liang, Jianzhong Guo, Jin Chen, Leyi Xia, Miaojun Wang, Mingming Li, Peng Zhang, Ruyi Chen, Shangmian Sun, Shaoqing Wu, Shengfeng Ye, T. Wang, W. L. Xiao, Wei An, Xianzu Wang, Xiaowen Sun, Xiaoxiang Wang, Ying Tang, Yukun Zha, Zekai Zhang, Zhe Ju, Zhen Zhang, and Zihua Qu. Deepseek-v3.2: Pushing the frontier of open large language models, 2025. URL https://arxiv.org/abs/2512.02556.

Gelei Deng, Yi Liu, Yuekang Li, Kailong Wang, Ying Zhang, Zefeng Li, Haoyu Wang, Tianwei Zhang, and Yang Liu. MASTERKEY: Automated jailbreaking of large language model chatbots. In *Network and Distributed System Security Symposium (NDSS)*, 2024.

Jonathan Ding, Will Cai, Khanak Jain, Dhruv Nair, Aditya Naha, Kevin Zhu, and Vasu Sharma. Automated generation of multilingual jailbreak prompts. In *NeurIPS Workshop on Reliable ML from Unreliable Data*, 2025.

Michael Freenor, Lauren Alvarez, Milton Leal, Lily Smith, Joel Garrett, Yelyzaveta Husieva, Madeline Woodruff, Ryan Miller, Erich Kummerfeld, Rafael Medeiros, and Sander Schulhoff. Prompt optimization and evaluation for LLM automated red teaming. *arXiv preprint arXiv:2507.22133*, 2025.

Kai Fronsdal, Isha Gupta, Abhay Sheshadri, Jonathan Michala, Stephen McAleer, Rowan Wang, Sara Price, and Samuel R. Bowman. PETRI: Parallel exploration tool for risky interactions. 2025. https://alignment.anthropic.com/2025/petri/.

Junwoo Ha, Hyunjun Kim, Sangyoon Yu, Haon Park, Ashkan Yousefpour, Yuna Park, and Suhyun Kim. M2s: Multi-turn to single-turn jailbreak in red teaming for llms. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 16489–16507, 2025.

Helia Hashemi, Jason Eisner, Corby Rosset, Benjamin Van Durme, and Chris Kedzie. Llm-rubric: A multidimensional, calibrated approach to automated evaluation of natural language texts. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, page 13806–13834. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.745. URL http://dx.doi.org/10.18653/v1/2024.acl-long.745.

Hakan Inan, Kartikeya Upasani, Jianfeng Chi, Rashi Rungta, Krithika Iyer, Yuning Mao, Michael Tontchev, Qing Hu, Brian Fuller, Davide Testuggine, and Madian Khabsa. Llama guard: LLM-based input-output safeguard for human-AI conversations. *arXiv preprint arXiv:2312.06674*, 2023.

Fengqing Jiang, Zhangchen Xu, Luyao Niu, Zhen Xiang, Bhaskar Ramasubramanian, Bo Li, and Radha Poovendran. ArtPrompt: ASCII art-based jailbreak attacks against aligned LLMs. In *Proceedings of ACL*, 2024.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners, 2023. URL https://arxiv.org/abs/2205.11916.

Nathaniel Li, Ziwen Han, Ian Steneker, Willow Primack, Riley Goodside, Hugh Zhang, Zifan Wang, Cristina Menghini, and Summer Yue. Llm defenses are not robust to multi-turn human jailbreaks yet, 2024. URL https://arxiv.org/abs/2408.15221.

Xiaogeng Liu, Nan Xu, Muhao Chen, and Chaowei Xiao. AutoDAN: Generating stealthy jailbreak prompts on aligned large language models. In *International Conference on Learning Representations (ICLR)*, 2024.

Xiaogeng Liu, Peiran Li, Edward Suh, Yevgeniy Vorobeychik, Zhuoqing Mao, Somesh Jha, Patrick McDaniel, Huan Sun, Bo Li, and Chaowei Xiao. AutoDAN-turbo: A lifelong agent for strategy self-exploration to jailbreak LLMs. In *International Conference on Learning Representations (ICLR)*, 2025.

Ninareh Mehrabi, Palash Goyal, Christophe Dupuy, Qian Hu, Shalini Ghosh, Richard Zemel, Kai-Wei Chang, Aram Galstyan, and Rahul Gupta. FLIRT: Feedback loop in-context red teaming. In *Proceedings of EMNLP*, 2024.

Anay Mehrotra, Manolis Zampetakis, Paul Kassianik, Blaine Nelson, Hyrum Anderson, Yaron Singer, and Amin Karbasi. Tree of attacks: Jailbreaking black-box LLMs automatically. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Gary D. Lopez Munoz, Amanda J. Minnich, Roman Lutz, Richard Lundeen, Raja Sekhar Rao Dheekonda, Nina Chikanov, Bolor-Erdene Jagdagdorj, Martin Pouliot, Shiven Chawla, Whitney Maxwell, Blake Bullwinkel, Katherine Pratt, Joris de Gruyter, Charlotte Siska, Pete Bryan, Tori Westerhoff, Chang Kawaguchi, Christian Seifert, Ram Shankar Siva Kumar, and Yonatan Zunger. Pyrit: A framework for security risk identification and red teaming in generative ai system, 2024. URL https://arxiv.org/abs/2410.02828.

OpenAI, Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K. Arora, Yu Bai, Bowen Baker, Haiming Bao, Boaz Barak, Ally Bennett, Tyler Bertao, Nivedita Brett, Eugene Brevdo, Greg Brockman, Sebastien Bubeck, Che Chang, Kai Chen, Mark Chen, Enoch Cheung, Aidan Clark, Dan Cook, Marat Dukhan, Casey Dvorak, Kevin Fives, Vlad Fomenko, Timur Garipov, Kristian Georgiev, Mia Glaese, Tarun Gogineni, Adam Goucher, Lukas Gross, Katia Gil Guzman, John Hallman, Jackie Hehir, Johannes Heidecke, Alec Helyar, Haitang Hu, Romain Huet, Jacob Huh, Saachi Jain, Zach Johnson, Chris Koch, Irina Kofman, Dominik Kundel, Jason Kwon, Volodymyr Kyrylov, Elaine Ya Le, Guillaume Leclerc, James Park Lennon, Scott Lessans, Mario Lezcano-Casado, Yuanzhi Li, Zhuohan Li, Ji Lin, Jordan Liss, Lily, Liu, Jiancheng Liu, Kevin Lu, Chris Lu, Zoran Martinovic, Lindsay McCallum, Josh McGrath, Scott McKinney, Aidan McLaughlin, Song Mei, Steve Mostovoy, Tong Mu, Gideon Myles, Alexander Neitz, Alex Nichol, Jakub Pachocki, Alex Paino, Dana Palmie, Ashley Pantuliano, Giambattista Parascandolo, Jongsoo Park, Leher Pathak, Carolina Paz, Ludovic Peran, Dmitry Pimenov, Michelle Pokrass, Elizabeth Proehl, Huida Qiu, Gaby Raila, Filippo Raso, Hongyu Ren, Kimmy Richardson, David Robinson, Bob Rotsted, Hadi Salman, Suvansh Sanjeev, Max Schwarzer, D. Sculley, Harshit Sikchi, Kendal Simon, Karan Singhal, Yang Song, Dane Stuckey, Zhiqing Sun, Philippe Tillet, Sam Toizer, Foivos Tsimpourlas, Nikhil Vyas, Eric Wallace, Xin Wang, Miles Wang, Olivia Watkins, Kevin Weil, Amy Wendling, Kevin Whinnery, Cedric Whitney, Hannah Wong, Lin Yang, Yu Yang, Michihiro Yasunaga, Kristen Ying, Wojciech Zaremba, Wenting Zhan, Cyril Zhang, Brian Zhang, Eddie Zhang, and Shengjia Zhao. gpt-oss-120b and gpt-oss-20b model card, 2025. URL https://arxiv.org/abs/2508.10925.

Zifan Peng, Yule Liu, Zhen Sun, Mingchen Li, Zeren Luo, Jingyi Zheng, Wenhan Dong, Xinlei He, Xuechao Wang, Yingjie Xue, Shengmin Xu, and Xinyi Huang. JALMBench: Benchmarking jailbreak vulnerabilities in audio language models. In *International Conference on Learning Representations (ICLR)*, 2026.

Ethan Perez, Saffron Huang, Francis Song, Trevor Cai, Roman Ring, John Aslanides, Amelia Glaese, Nat McAleese, and Geoffrey Irving. Red teaming language models with language models. In *Proceedings of EMNLP*, 2022.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2019.

Qibing Ren, Hao Li, Dongrui Liu, Zhanxu Xie, Xiaoya Lu, Yu Qiao, Lei Sha, Junchi Yan, Lizhuang Ma, and Jing Shao. Derail yourself: Multi-turn LLM jailbreak attack through self-discovered clues. *arXiv preprint arXiv:2410.10700*, 2024.

Mark Russinovich, Ahmed Salem, and Ronen Eldan. Great, now write an article about that: The crescendo multi-turn LLM jailbreak attack. In *USENIX Security Symposium*, 2025.

Chantal Shaib, Yanai Elazar, Junyi Jessy Li, and Byron C. Wallace. Detection and measurement of syntactic templates in generated text. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024.

Chantal Shaib, Venkata S. Govindarajan, Joe Barrow, Jiuding Sun, Alexa F. Siu, Byron C. Wallace, and Ani Nenkova. Standardizing the measurement of text diversity: A tool and a comparative analysis of scores. In *Proceedings of AACL*, 2025.

Abhishek Singhania, Christophe Dupuy, Shivam Mangale, and Amani Namboori. MM-ART: Multilingual multi-turn automated red teaming for LLMs. In *TrustNLP Workshop at NAACL*, 2025.

Xin Wang, Yunhao Chen, Juncheng Li, Yixu Wang, Yang Yao, Tianle Gu, Jie Li, Yan Teng, Yingchun Wang, and Xia Hu. Openrt: An open-source red teaming framework for multimodal llms, 2026. URL `https://arxiv.org/abs/2601.01592`.

Yong Wang et al. Opposite day: Exploiting instruction following to jailbreak large language models. *arXiv preprint arXiv:2408.15221*, 2024.

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

Halley Young, Yimeng Zeng, Jacob Gardner, and Osbert Bastani. Improving structural diversity of blackbox LLMs via chain-of-specification prompting. *arXiv preprint arXiv:2408.06186*, 2024.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. Texygen: A benchmarking platform for text generation models. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1097–1100, 2018.

Yuchang Zhu, Huizhe Zhang, Bingzhe Wu, Jintang Li, Zibin Zheng, Peilin Zhao, Liang Chen, and Yatao Bian. Measuring diversity in synthetic datasets. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, 2025.

Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J. Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*, 2023.