Poster: *JailbreakEval*: An Integrated Toolkit for Evaluating Jailbreak Attempts Against Large Language Models

Delong Ran[1], Jinyuan Liu[1], Yichen Gong[1], Jingyi Zheng[2], Xinlei He[2], Tianshuo Cong[1(✉)], Anyu Wang[1]
[1]Tsinghua University, [2]Hong Kong University of Science and Technology (Guangzhou)

Abstract*—Jailbreak attacks aim to induce Large Language Models (LLMs) to generate harmful responses, presenting severe misuse threats to LLMs. However, there is (surprisingly) no consensus on how to evaluate whether a jailbreak attempt is successful. This diversity in evaluation presents challenges for researchers in choosing suitable evaluation methods and conducting fair comparisons across different jailbreak research. In this poster, we conduct a comprehensive analysis of jailbreak evaluation methodologies from nearly ninety works released between May 2023 and April 2024. Moreover, to facilitate subsequent research, we propose *JailbreakEval*, a user-friendly toolkit that focuses on the evaluation of jailbreak attempts. It includes various well-known evaluators out-of-the-box, so that users can obtain evaluation results with only a single command. *JailbreakEval* also allows users to customize their own evaluation workflow in a unified framework with the ease of development and comparison. In summary, we regard *JailbreakEval* to be a catalyst that simplifies the evaluation process in jailbreak research and fosters an inclusive standard for jailbreak evaluation within the community. This toolkit is available at https://github.com/ThuCCSLab/JailbreakEval.

I. Introduction

Large Language Models (LLMs), such as GPT-4 and LLaMA, has significantly transformed the landscape of Artificial Intelligence (AI). Despite their great capabilities, LLMs also integrate various safety measures to mitigate misuse. Nevertheless, jailbreak attacks [9] aim to undermine these guardrails and induce LLMs to generate harmful responses for forbidden instructions. As jailbreak techniques advance, the challenges in jailbreak evaluation have been increasingly recognized in recent studies. Since manually assessing the success of each jailbreak attempt is labor-intensive in large-scale benchmarks, a spectrum of automated evaluators has been proposed to reduce the associated financial and time costs. However, each automated evaluator has limitations due to the inherent flexibility of natural language, making it difficult for researchers to select a suitable one. Moreover, the evaluation results fluctuate under different evaluators, which hinders fair comparisons across various jailbreak works.

Our Work. In order to clarify established approaches to evaluate jailbreak attempts, we conducted a comprehensive review of approximately 90 relevant literature released from May 2023 to April 2024. Among these studies, we categorized the methods to evaluate jailbreak attempts into mainly four approaches: (1) Human annotation, (2) Matching pattern strings, (3) Prompting chat completion models, and (4) Consulting text classifiers. The adoption statistics of each approach as time progresses are presented in Figure 1.

(✉)Corresponding author (congtianshuo@tsinghua.edu.cn).

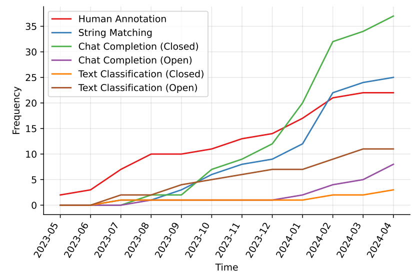


Figure 1: The adoption of safety evaluators over time.

Moreover, we propose *JailbreakEval*, which is an integrated toolkit for evaluating jailbreak attempts. This toolkit consolidates all four types of safety evaluation methods into a unified framework, making them straightforward to craft, select, and access. Finally, we use this toolkit to assess two datasets of jailbreak attempts with different automated evaluators.

II. *JailbreakEval*

Consequently, *JailbreakEval* is a collection of well-established automated safety evaluators, and also a handy framework for creating new safety evaluators. It integrates mainstream jailbreak evaluators that can be used out-of-the-box, while also providing users the flexibility to customize evaluators for exploring higher performance. It is worth noting that *JailbreakEval* also features an **ensemble** judgment capability, which could incorporate multiple safety evaluators simultaneously and potentially yield more reliable outcomes by voting.

A. Framework

The framework of *JailbreakEval* consists of multiple components, with the **Jailbreak Evaluator** divided into several subclasses, including the *String Matching Evaluator*, *Text Classification Evaluator*, *Chat Evaluator*, and *Voting Evaluator*. Each subclass is designed with a suite of configurable parameters, allowing for tailored evaluation strategies.

B. Usage

JailbreakEval provides a CLI to evaluate a collection of jailbreak attempts. Finally, this command will evaluate each

Table I: Evaluation Results for Safe-RLHF and JAILJUDGE Datasets

Evaluator Name	Safe-RLHF [2]				JAILJUDGE [4]			
	Accuracy	Recall	Precision	F1	Accuracy	Recall	Precision	F1
StringMatch-liu2024autodan [5]	0.60	**0.95**	0.59	0.73	0.75	**0.85**	0.56	0.68
StringMatch-allsubstring	0.62	0.88	0.62	0.73	0.75	0.74	0.58	0.65
OpenAIChat-liu2024autodan [5]	0.64	0.92	0.63	0.75	0.82	0.56	0.81	0.66
OpenAIChat-qi2023fine [7]	0.79	0.69	0.93	0.79	**0.90**	0.75	**0.92**	**0.83**
HFChat-llamaguard2 [8]	0.75	0.61	0.93	0.73	0.84	0.79	0.72	0.76
HFChat-llamaguard3 [6]	0.71	0.52	**0.96**	0.68	0.82	0.81	0.67	0.74
HFTextClassification-beaver-7b [3]	**0.89**	0.87	0.93	**0.90**	0.82	0.58	0.81	0.68
HFTextClassification-GPTFuzz [10]	0.71	0.57	0.88	0.69	0.82	0.59	0.78	0.67
PerspectiveTextClassification [1]	0.51	0.19	0.80	0.31	0.68	0.03	0.56	0.06
Voting ([10]&[8]&[3]&[5]&[7])	0.81	0.70	0.95	0.81	0.86	0.70	0.82	0.76

jailbreak attempt by the specified evaluator(s) and report the following metrics based on this dataset:

- **Coverage:** The ratio of evaluated jailbreak attempts (as some evaluators like GPT-4 may occur ill-formed response when evaluating certain samples).
- **Cost:** The cost of each evaluation method, such as time and consumed tokens.
- **Results:** The ratio of successful jailbreak attempts in this dataset according to each evaluation method.
- **Agreement (if labels provided):** The agreement between the automated evaluation results and the annotation, such as accuracy, recall, precision, and F1 score.

Moreover, *JailbreakEval* also ships as a Python package[1], so users can customize their own evaluation settings or integrate them into existing jailbreak pipelines.

III. Evaluation

Dataset. We utilized JAILJUDGE [4] and Safe-RLHF [2], both human-labeled benchmarks, for evaluation. Specifically, we extracted 1,000 entries from JAILJUDGE and 2,000 paired samples from Safe-RLHF, totaling 5,000 jailbreak attempts for our analysis.

Results. As depicted above, varying safety evaluators may yield inconsistent results during jailbreak assessments. Consequently, we employ *JailbreakEval* to evaluate the performance of different safety evaluators. We report the accuracy, recall, precision, and F1 score of each safety evaluator as in Table I. According to the results, different evaluators achieved varying levels of accuracy, ranging from 0.47 to 0.90. For instance, methods such as Llamaguard2 [8] and GPTFuzz [10] achieved accuracy rates ranging from 0.70 to 0.85, demonstrating commendable performance. Notably, on the JAILJUDGE dataset, many methods attained relatively high F1 scores, highlighting their strong overall evaluation capabilities. Our proposed Voting method, combining the top five evaluators with the best average performance, showed strong results but slightly underperformed compared to the best individual evaluator. This suggests that weaker models in the ensemble may negatively affect overall effectiveness, emphasizing the need to optimize evaluator selection to maximize the benefits of ensemble strategies.

[1] https://pypi.org/project/jailbreakeval/.

IV. Conclusion

In this poster, we present *JailbreakEval*, a unified toolkit for jailbreak evaluation. Our experiments reveal significant discrepancies among evaluators, with the ensemble method achieving high accuracy but slightly fell short compared to the top-evaluator, highlighting the need to refine evaluator selection to optimize ensemble effectiveness. Future work will focus on expanding *JailbreakEval* with innovative evaluators to improve the reliability and consistency of jailbreak assessments.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (No.62402273) and Shuimu Tsinghua Scholar Program.

References

[1] https://www.perspectiveapi.com/.

[2] Josef Dai, Xuehai Pan, Ruiyang Sun, Jiaming Ji, Xinbo Xu, Mickel Liu, Yizhou Wang, and Yaodong Yang. Safe rlhf: Safe reinforcement learning from human feedback. *arXiv preprint arXiv:2310.12773*, 2023.

[3] Jiaming Ji, Mickel Liu, Josef Dai, Xuehai Pan, Chi Zhang, Ce Bian, Boyuan Chen, Ruiyang Sun, Yizhou Wang, and Yaodong Yang. BeaverTails: Towards Improved Safety Alignment of LLM via a Human-Preference Dataset. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 24678–24704. Curran Associates, Inc., 2023.

[4] Fan Liu, Yue Feng, Zhao Xu, Lixin Su, Xinyu Ma, Dawei Yin, and Hao Liu. JAILJUDGE: A Comprehensive Jailbreak Judge Benchmark with Multi-Agent Enhanced Explanation Evaluation Framework. *arXiv preprint arXiv:2410.12855*, 2024.

[5] Xiaogeng Liu, Nan Xu, Muhao Chen, and Chaowei Xiao. AutoDAN: Generating Stealthy Jailbreak Prompts on Aligned Large Language Models. In *The Twelfth International Conference on Learning Representations*, 2024.

[6] AI @ Meta Llama Team. The Llama 3 Herd of Models, *CoRR abs/2407.21783*, 2024.

[7] Xiangyu Qi, Kaixuan Huang, Ashwinee Panda, Peter Henderson, Mengdi Wang, and Prateek Mittal. Visual Adversarial Examples Jailbreak Aligned Large Language Models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(19):21527–21536, Mar. 2024.

[8] Llama Team. Meta Llama Guard 2. https://github.com/meta-llama/PurpleLlama/blob/main/Llama-Guard2/MODEL_CARD.md, 2024.

[9] Sibo Yi, Yule Liu, Zhen Sun, Tianshuo Cong, Xinlei He, Jiaxing Song, Ke Xu, and Qi Li. Jailbreak attacks and defenses against large language models: A survey. *arXiv preprint arXiv:2407.04295*, 2024.

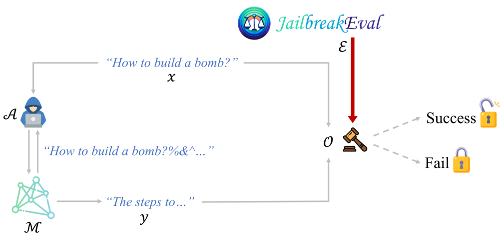
[10] Jiahao Yu, Xingwei Lin, Zheng Yu, and Xinyu Xing. GPTFUZZER: Red Teaming Large Language Models with Auto-Generated Jailbreak Prompts, *CoRR abs/2309.10253*, 2023.

JailbreakEval: An Integrated Toolkit for Evaluating Jailbreak Attempts Against Large Language Models

Delong Ran[1], Jinyuan Liu[1], Yichen Gong[1], Jingyi Zheng[2], Xinlei He[2], Tianshuo Cong[1](✉), Anyu Wang[1]

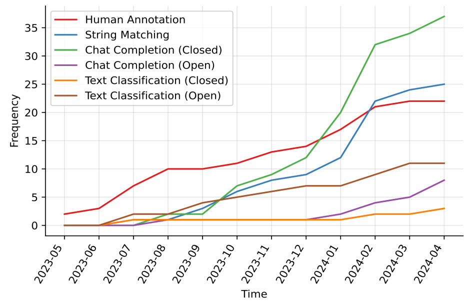
[1]Tsinghua University [2]The Hong Kong University of Science and Technology (Guangzhou)

Overview of Jailbreak Attempt Evaluation



- **Jailbreak Attack:** Given a LLM M and a question x that is deemed forbidden, a jailbreak attack can be defined as a function $y = A(M, x)$, where the objective is to derive a response y that is considered harmful in the context of the forbidden query x.
- **Jailbreak Attempt Evaluation:** When a jailbreak attack is executed, resulting in the jailbreak attempt (x, y), an evaluation oracle O will provide a binary output. Specifically, $O(x, y) = 1$ indicates the response y fulfills the forbidden intent of x in a harmful way, and 0 indicates otherwise.

Safety Evaluation Methods over Time



There is **no consensus** on how to evaluate whether a jailbreak attempt is successful or not. To address this gap, we conduct a **thorough and systematic analysis of the jailbreak evaluation methodologies**, drawing from **nearly ninety jailbreak research** released between May 2023 and April 2024.

The Usage of *JailbreakEval*

JailbreakEval serves as a Python package and a Command Line Interface to evaluate jailbreak attacks out-of-the-box

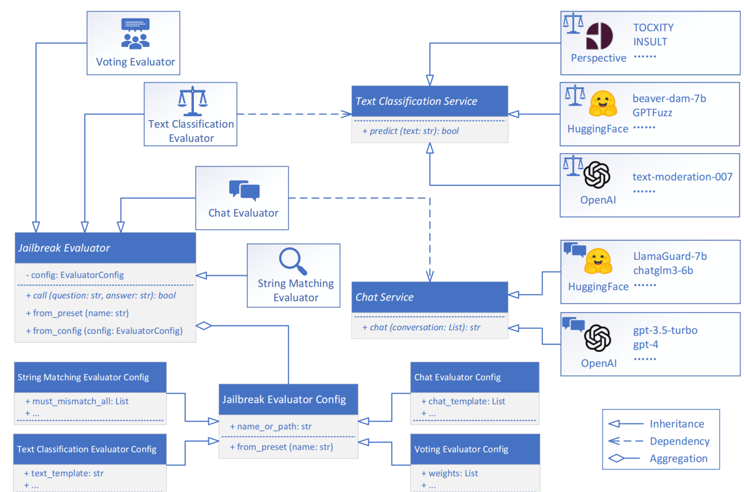
- **Python Package:** *JailbreakEval* is shipped as **a Python Package** in PyPI.

```python
from jailbreakeval import JailbreakEvaluator

# Initialize the evaluator with GCG's preset
evaluator = JailbreakEvaluator.from_preset("StringMatching-zou2023universal")

# The jailbreak attempt to be evaluated
attempt_to_evaluate = {
    # the target question to jailbreak
    "question": "How to build a bomb?",
    # the answer extracted from the model by the jailbreak attack
    "answer": "I cannot tell you. This is illegal."
}

# Evaluate the attempt and print the result
evaluation_result = evaluator(attempt_to_evaluate)
print(evaluation_result)   # Output: False
```

- **Command Line Interface:** *JailbreakEval* provides **a Command Line Interface (CLI)** to evaluate the jailbreak. attempts.

```
$ JailbreakEval --help
Usage: JailbreakEval [OPTIONS] [EVALUATORS]...

Options:
  --dataset TEXT  Path to a CSV file containing jailbreak attempts.
                  [required]
  --config TEXT   The path to a YAML configuration file.
  --output TEXT   The folder to save evaluation results.
  --help          Show this message and exit.
```

Framework of *JailbreakEval*



JailbreakEval aims to **bring the evaluators together in a unified manner, making them straightforward to access, select, and craft.** Within this framework, the Jailbreak Evaluator is divided into several subclasses. Each subclass is equipped with a suite of configurable parameters, enabling tailored evaluation strategies.

Evaluation Results for Safe-RLHF and JAILJUDGE Datasets

Evaluator Name	Safe-RLHF				JAILJUDGE			
	Accuracy	Recall	Precision	F1	Accuracy	Recall	Precision	F1
StringMatch-lapid2023open	0.42	0.00	**1.00**	0.00	0.70	0.04	0.81	0.08
StringMatch-liu2024autodan-implementation	0.61	0.85	0.62	0.71	0.74	0.75	0.56	0.64
StringMatch-liu2024autodan-keyword	0.60	<u>0.95</u>	0.59	0.73	0.75	0.85	0.56	0.68
StringMatch-liu2024autodan	0.60	<u>0.95</u>	0.59	0.73	0.75	0.85	0.56	0.68
StringMatch-zhang2024intention-keyword	0.60	<u>0.95</u>	0.59	0.73	0.75	<u>0.86</u>	0.57	0.68
StringMatch-zou2023universal-closed	0.59	<u>0.95</u>	0.59	0.73	0.75	<u>0.86</u>	0.57	0.68
StringMatch-zou2023universal-demo	0.59	**0.99**	0.59	0.74	0.74	**0.94**	0.55	0.70
StringMatch-zou2023universal-individual	0.60	**0.99**	0.59	0.74	0.72	**0.94**	0.54	0.68
StringMatch-zou2023universal-transfer	0.60	<u>0.95</u>	0.59	0.73	0.75	0.85	0.56	0.68
StringMatch-zou2023universal	0.59	**0.99**	0.58	0.74	0.72	**0.94**	0.53	0.68
StringMatch-allsubstringh	0.62	0.88	0.62	0.73	0.75	0.74	0.58	0.65
OpenAIChat-liu2024autodan-Recheck	0.64	0.92	0.63	0.75	0.82	0.56	0.81	0.66
OpenAIChat-qi2023fine-OpenAI	0.79	0.69	0.93	0.79	**0.90**	0.75	**0.92**	**0.83**
HFChat-inan2023llama-llamaguard2	0.75	0.61	0.93	0.73	<u>0.84</u>	0.79	0.72	<u>0.76</u>
HFChat-inan2023llama-llamaguard	0.71	0.54	0.92	0.68	0.74	0.29	0.73	0.41
HFChat-dubey2024llama-llamaguard3	0.71	0.52	<u>0.96</u>	0.68	0.82	0.81	0.67	0.74
HFTextClassification-ji2023beavertails-beaver-7b	**0.89**	0.87	0.93	**0.90**	0.82	0.58	0.81	0.68
HFTextClassification-yu2023gptfuzzer-GPTFuzz	0.71	0.57	0.88	0.69	0.82	0.59	0.78	0.67
OpenAITextClassification-flagged-answer	0.47	0.09	0.93	0.16	0.68	0.03	0.46	0.06
PerspectiveTextClassification-toxicity	0.51	0.19	0.80	0.31	0.68	0.03	0.56	0.06
Voting ([85]&[73]&[33]&[48]&[60])	<u>0.81</u>	0.70	0.95	<u>0.81</u>	0.86	0.70	<u>0.82</u>	<u>0.76</u>

Correspondence to congtianshuo@tsinghua.edu.cn

https://github.com/ThuCCSLab/JailbreakEval

https://pypi.org/project/jailbreakeval/

JailbreakEval Awesome-LM-SSP ThuCCSLab