USENIX Security ’25 Artifact Appendix: JBShield: Defending Large Language Models from Jailbreak Attacks through Activated Concept Analysis and Manipulation

Shenyi Zhang[1], Yuchen Zhai[1], Keyan Guo[2], Hongxin Hu[2], Shengnan Guo[1], Zheng Fang[1],
Lingchen Zhao[1], Chao Shen[3], Cong Wang[4], and Qian Wang[1]
[1] Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education,
School of Cyber Science and Engineering, Wuhan University,
[2] University at Buffalo, [3] Xi’an Jiaotong University, [4] City University of Hong Kong

A Artifact Appendix

A.1 Abstract

This artifact supports the evaluation of JBShield, a comprehensive jailbreak defense framework for large language models (LLMs). JBShield leverages the Linear Representation Hypothesis (LRH) to investigate the mechanisms underlying jailbreak attacks, defining and analyzing high-level toxic and jailbreak concepts encoded in LLM hidden representations. By identifying and manipulating these concepts, JBShield effectively detects and mitigates jailbreak prompts, ensuring safer outputs. This artifact includes: (1) datasets used for evaluation, encompassing nine types of jailbreak prompts across five target LLMs, benign prompts from the Alpaca dataset, and harmful prompts from AdvBench and Hex-PHI; (2) implementations for extracting concept-related interpretable tokens; and (3) testing scripts for evaluating JBShield-D (detection) and JBShield-M (mitigation) across five LLMs. These resources facilitate reproducibility and provide a comprehensive assessment of JBShield’s effectiveness in reducing attack success rates and enhancing LLM safety.

A.2 Description & Requirements

A.2.1 Security, privacy, and ethical concerns

Our research focuses on defending jailbreak attacks on large language models (LLMs). As such, the provided dataset contains examples of harmful language, and the evaluation process may involve executing jailbreak attacks that could lead the model to generate ethically or morally questionable content. While these outputs are used strictly for research and defense evaluation purposes, reader discretion is recommended. Evaluators should take necessary precautions when reviewing the dataset and generated outputs to ensure responsible handling of potentially harmful content.

A.2.2 How to access

The artifact is publicly available and can be accessed at the newest version under the Zenodo URL https://zenodo.org/records/14732884 and GitHub repo https://github.com/NISPLab/JBShield. This repository contains all necessary components for evaluating JBShield, including the datasets, code for extracting concept-related tokens, and implementations of JBShield-D (detection) and JBShield-M (mitigation) for testing across five target LLMs.

A.2.3 Hardware dependencies

Since our research focuses on evaluating large language models (LLMs), the primary hardware requirement is a GPU with sufficient memory to handle inference and jailbreak detection tasks. The minimum hardware requirement is two GPUs with at least 24GB VRAM each (e.g., RTX 3090 or RTX 4090). For optimal performance, we recommend a setup with 4 RTX 4090 GPUs (24GB VRAM each) or 1 A100 GPUs (80GB VRAM each). These configurations ensure efficient execution of JBShield’s detection and mitigation components across multiple LLMs.

A.2.4 Software dependencies

We recommend evaluating our artifact on a Linux-based system. Our implementation has been tested and verified on Ubuntu 20.04.2 with Linux kernel versions 5.15.0-130-generic and 5.15.0-125-generic. The artifact requires Python for execution, along with several third-party libraries. We suggest following the official PyTorch installation guide[1] to set up PyTorch, while other dependencies can be installed using Pypi. Additionally, our evaluation utilizes five target LLMs and one evaluation LLM, all of which can be downloaded from HuggingFace. For further details, please refer to the README file included in the artifact.

[1] https://pytorch.org/get-started/locally

A.2.5 Benchmarks

The dataset required for the experiments reported in our paper is fully included in the artifact (in *./data*). It comprises harmful prompts (*./data/harmful{}.csv*), benign prompts (*./data/harmless{}.csv*) and jailbreak prompts (in *./data/jailbreak*). Harmful prompts sourced from AdvBench and HEx-PHI datasets. Benign prompts sampled from the Alpaca dataset. Jailbreak prompts generated using nine different jailbreak attack methods, targeting five different LLMs. All jailbreak prompts were generated by our own. The dataset is structured and readily accessible within the artifact for ease of use in reproducing our experimental results. Note that our mitigation evaluation is conducted using 50 samples per jailbreak attack. Specifically, for transfer-based attacks (IJP, Puzzler, Zulu, and Base64) that do not directly exploit target LLM information, we randomly select 50 corresponding jailbreak prompts from our dataset to assess the attack success rate (ASR). For the other jailbreak methods, we treat the defended model as a new target LLM, generate 50 new jailbreak prompts, and compute the ASR accordingly. The evaluation data for jailbreak mitigation is provided in *./data/mitigation*, following the same file structure as *./data/jailbreak*.

A.3 Set-up

A.3.1 Installation

The code for JBShield runs with Python 3 and requires Pytorch. We recommend using Anaconda or miniconda for python. Our code has been tested with python=3.12.8 and torch=2.5.1 on linux. First, install conda[2] (if not already installed), create a conda environment and activate it. Follow the official PyTorch installation guide to install Pytorch and pip install the other packages. Than download and set up all models required. For further details on installation and configuration, please refer to the README file included in the artifact.

A.3.2 Basic Test

To verify that the installation is successful and all required software components are functioning correctly, Run a sample Concept Extraction test in shell by

```
python interpret.py –model mistral
```

Expected output is as

```
[nltk_data] Downloading package words to <path>...
[nltk_data] Package words is already up-to-date!
Number of harmful prompts: 850
Number of harmless prompts: 850
Number of gcg prompts: 850
Number of autodan prompts: 850
Number of saa prompts: 850
Number of drattack prompts: 520
Number of pair prompts: 850
Number of puzzler prompts: 50
Number of ijp prompts: 850
Number of base64 prompts: 850
Number of zulu prompts: 850
Number of harmful prompts: 850
Number of harmless prompts: 850
mistral Interpretation done.
```

A.4 Evaluation workflow

A.4.1 Major Claims

The major claims made in your paper are enumerated as follows:

(C1): Our method for extracting concept-related interpretable tokens enables a deeper understanding of jailbreak mechanisms by identifying toxic and jailbreak concepts encoded in LLM representations. This is demonstrated through in Section 3 (Activated Concept Analysis), with supporting results shown in Table 2 and Appendix A.

(C2): JBShield-D accurately detects jailbreak prompts by analyzing the activation of both toxic and jailbreak concepts. The effectiveness of JBShield-D is validated by the experiments in Section 5.4 (Jailbreak Detection), where it achieves an average detection accuracy of 0.95 across five LLMs. The results are presented in Table 4.

(C3): JBShield-M successfully mitigates jailbreak attacks by adjusting the hidden representations of LLMs, enhancing toxic concepts while suppressing jailbreak concepts. The effectiveness of JBShield-M is evaluated in Section 5.5 (Jailbreak Mitigation), demonstrating a reduction in jailbreak attack success rates from 61% to 2% across five LLMs. The results are detailed in Table 7.

A.4.2 Experiments

To evaluate JBShield and validate our key claims, we conduct the following experiments. Each experiment is associated with one of the major claims listed in Section A.4.1 and provides details on the execution steps, expected runtime, and required resources.

(E1): [Concept-related Interpretable Token Extraction] [~1 compute-hour]: validate that JBShield's extraction of concept-related interpretable tokens can help in understanding jailbreak mechanisms and identifying toxic and jailbreak concepts.

[2] https://docs.anaconda.com/anaconda/install

How to: Extract concept-related tokens from five target LLMs using JBShield's interpretation module.
Preparation: Ensure that the environment and models is set up following the installation instructions in Section A.3. Verify that calibration and test datasets are properly structured in $./data/jailbreak$ as mentioned in README.
Execution: Run the interpretation script:

```
chmod +x ./interpret.sh
./interpret.sh
```

Results: The extracted tokens for each model are saved in $./interpret\_results/\{model\_name\}.txt$. These results should align with the findings presented in Table 2 and Appendix A of the paper.

(E2): [Jailbreak Detection Performance of JBShield-D] [~4 compute-hour]: evaluate JBShield-D's jailbreak detection performance across five LLMs and verifies that the detection achieves high accuracy and F1-Score.
How to: Run the jailbreak detection module on the five target models and evaluate detection accuracy and F1-Score.
Preparation: Ensure that the environment and models is set up following the installation instructions in Section A.3. Verify that calibration and test datasets are properly structured in $./data/jailbreak$ as mentioned in README.
Execution: Run the detection script:

```
chmod +x ./evaluate_detection.sh
./evaluate_detection.sh
```

Results: The detection performance logs are saved in $./logs/JBShield-D\_\{model\_name\}.log$. Expected results should match Table 4 in the paper.

(E3): [Jailbreak Mitigation Performance of JBShield-M] [~2 compute-hour]: evaluates JBShield-M's ability to mitigate jailbreak attacks and reduce the attack success rate.
How to: Apply JBShield-M to the five target models and measure the attack success rate.
Preparation: Ensure the detection module is functioning correctly by verifying results from (E2).
Execution: Run the mitigation script:

```
chmod +x ./evaluate_mitigation.sh
./evaluate_mitigation.sh
```

Results: The mitigation results are saved in /logs/JBShield-M.log. Expected results should align with Table 7 in the paper, showing an attack success rate reduction from 61% to 2%.

A.5 Notes on Reusability

Please note that the experimental results presented in this artifact may exhibit some variations due to differences in testing environments, the randomness in calibration set selection, and dataset size discrepancies across different jailbreak methods (DrAttack and Puzzler contain fewer samples compared to others). Despite these variations, the overall trends and effectiveness of JBShield remain stable, as demonstrated by the extensive evaluations conducted in our study.

A.6 Version

Based on the LaTeX template for Artifact Evaluation V20231005. Submission, reviewing and badging methodology followed for the evaluation of this artifact can be found at `https://secartifacts.github.io/usenixsec2025/`.