USENIX Security '25 Artifact Appendix: Refusal Is Not an Option: Unlearning Safety Alignment of Large Language Models

Minkyoo Song Hanna Kim Jaehan Kim Seungwon Shin Sooel Son
KAIST, South Korea
{minkyoo9, gkssk3654, jaehan, claude, sl.son}@kaist.ac.kr

A Artifact Appendix

A.1 Abstract

This artifact provides the full implementation, experimental pipeline, and datasets used in our paper, which studies adversarial unlearning attacks that compromise the safety alignment of large language models (LLMs). Our codebase supports reproduction of all main results, including the construction and optimization of malicious unlearning datasets, application of four major unlearning algorithms to state-of-the-art open-source LLMs, and comprehensive evaluation against utility and safety benchmarks. Complete setup, training, and evaluation instructions are provided in the repository.

A.2 Description & Requirements

Hardware and Software Requirements:

- **Hardware**: Two NVIDIA A100 GPUs (80GB each), at least 100GB disk space (about 12GB for each unlearned LLaMA model).
- **OS**: Ubuntu 20.04 LTS.
- **CUDA**: Version 11.8.
- **Python**: 3.10.
- **Memory**: At least 256GB RAM is recommended.

See the `environment.yaml` in the repository for complete dependencies.

Benchmarks: All datasets and evaluation scripts needed to reproduce results are included or auto-downloaded. Benchmarks include harmfulness (AdvBench, HEx-Phi, LLM-LAT) and utility (MMLU, BBH, TruthfulQA, TriviaQA, AlpacaEval). For details, see the README or Section 5.1 of the paper.

A.2.1 Security, privacy, and ethical concerns

- No destructive operations or requirement to disable security mechanisms.
- All PII is synthetic or public; no use of real personal data.
- Models capable of harmful output should **not** be deployed beyond research.
- Compromised models are not distributed in the artifact.

A.2.2 How to access

- **Main repository**: `https://github.com/minkyoo9/Unlearning-safety-alignment`
- **Stable DOI**: `https://doi.org/10.5281/zenodo.16740884`

A.2.3 Hardware dependencies

Two NVIDIA A100 80GB GPUs (or equivalent with sufficient memory for 3B–4B parameter LLMs) are required.

Important: If you change the hardware setup, you must review and potentially update the following configuration and script files to ensure compatibility with your new environment:

- `harmfulness/default_config.yaml`
- `harmfulness/inference_config.yaml`
- `LLaMA-Factory/scripts/default_config.yaml`
- Hyperparameters in `LLaMA-Factory/scripts/full`
- Hyperparameters in `muse_bench/baselines/scripts`
- `harmfulness/run_eval_harmful.sh`

A.2.4 Software dependencies

- Ubuntu 20.04; Python 3.10; CUDA 11.8.
- All additional packages (PyTorch, Huggingface Transformers, etc.) listed in `environment.yaml`.
- Install:

```
conda env create -f environment.yaml
conda activate unlearning
```

A.2.5 Benchmarks

- Harmfulness: AdvBench, HEx-Phi, LLM-LAT
- Utility: MMLU, BBH, TruthfulQA, TriviaQA, AlpacaEval
- Filtering: PII (synthetic), fake news (Kaggle), copyright (Harry Potter)
- All datasets are referenced or provided.

A.3 Set-up

A.3.1 Installation

1. Clone repository:
   `git clone https://github.com/minkyoo9/Unlearning-safety-alignment.git`
   `cd Unlearning-safety-alignment`
2. Install dependencies:
   `conda env create -f environment.yaml`
   `conda activate unlearning`
3. Set your Hugging Face token as instructed in the README.

A.3.2 Basic Test

To verify that the environment supports inference with the target model, perform the steps following the instructions in the README (see "Installation and Setup" and "Scenario I: Step 1-1"). After execution, verify that the output file `outs_llama/out_AdvBench` is generated and contains the model's inference results (i.e., rejection responses). Successful completion of this step confirms that the environment is correctly set up for running inference with the target model.

A.4 Evaluation workflow

A.4.1 Major Claims

- (C1): Adversarial unlearning attacks significantly increase the harmfulness of LLMs by more than 10× while preserving general utility.
- (C2): Malicious unlearning requests can bypass filtering systems, compromising LLM safety.
- (C3): The proposed defense detects, on average, >90% of malicious unlearning attempts, thereby mitigating the impact of such attacks.

A.4.2 Experiments

(E1) Unlearning Attacks on Open-Source LLMs:
< 1 human-hour + < 1 GPU-hour per model.
Steps (Scenario I and II in the README):

- Prepare adversarial unlearning datasets.
- Apply DPO/NPO/GA/TV unlearning via provided scripts.
- Evaluate harmfulness/utility.

Outcome: Significantly increased harmfulness scores with utility preserved (Table 2 and 3 of the paper).

(E2) Bypassing Filtering Systems:
< 30 human-minutes + < 30 GPU-minutes.
Steps (Step 1 of Scenario II in the README):

- Construct merged datasets using two LLM agents.
- Apply filtering/classification scripts.

Outcome: >96% adversarial samples bypass filters (Table 4 of the paper).

(E3) Defense Evaluation:
< 30 human-minutes + < 30 GPU-minutes.
Steps (Mitigation in the README):

- Train/apply the defensive classifier.
- Test detection recall on adversarial datasets.

Outcome: Defense detects most of malicious requests, reducing harmfulness (Table 7 of the paper).

See all operational details and examples in the README.

A.5 Notes on Reusability

- The artifact is modular and can be adapted for alternative LLMs and custom datasets.
- Unlearning, data construction, and mitigation modules are independent.

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/.