

USENIX

THE ADVANCED COMPUTING SYSTEMS ASSOCIATION

What the Eyes See, the LLMs Miss: Exploiting Human Perception for Adversarial Text Attacks

Qin Yang, *University of Connecticut;* Lu Malloy, *University of Tennessee;* Joshua Lee, *University of California, Santa Barbara;* Xiaohan Chang, *University of Connecticut;* Meisam Mohammady, *Iowa State University;* Doowon Kim, *University of Tennessee;* Yuan Hong, *University of Connecticut*

https://www.usenix.org/conference/usenixsecurity26/presentation/yang-qin

This artifact appendix is included in the Artifact Appendices to the Proceedings of the 35th USENIX Security Symposium and appends to the paper of the same name that appears in the Proceedings of the 35th USENIX Security Symposium.

August 12–14, 2026 • Baltimore, MD, USA

ISBN 978-1-939133-58-8

Open access to the Artifact Appendices to the Proceedings of the 35th USENIX Security Symposium is sponsored by



جامعة الملك عبدالله للعلوم والتقنية

King Abdullah University of Science and Technology



USENIX Security '26 Artifact Appendix: What the Eyes See, the LLMs Miss: Exploiting Human Perception for Adversarial Text Attacks

Qin Yang[1]*, Lu Malloy[2]*, Joshua Lee[3], Xiaohan Chang[1], Meisam Mohammady[4]
Doowon Kim[2], Yuan Hong[1]

[1]*University of Connecticut*, [2]*University of Tennessee*,
[3]*University of California, Santa Barbara*, [4]*Iowa State University*

A Artifact Appendix

A.1 Abstract

This artifact contains the implementation of Human-Perceptible Adversarial Attacks (HPAA), together with datasets, detector-evaluation pipelines, user-study materials, and analysis scripts required to reproduce the results reported in the paper.

The artifact supports both commercial moderation systems (OpenAI Moderation, Google Perspective API, AWS Bedrock Guardrails, and Azure Content Safety) and local safety models (Llama Guard and ShieldGemma). All experiments were tested on Ubuntu Linux using Python and PyTorch.

A.2 Description & Requirements

A.2.1 Security, privacy, and ethical concerns

The artifact does not execute destructive operations, modify system configurations, or require disabling security mechanisms. The artifact itself does not collect personal information from evaluators. The user-study data included in the artifact have been anonymized and contain no personally identifiable information.

Some evaluation scripts interact with external moderation APIs, including OpenAI Moderation, Google Perspective API, AWS Bedrock Guardrails, and Azure Content Safety. These experiments require Internet connectivity and user-provided API credentials. Requests submitted to these services are subject to the corresponding providers' privacy policies and terms of service.

A.2.2 How to access

The artifact is archived on Zenodo and available at: https://doi.org/10.5281/zenodo.20335299. The source code is also available at: https://github.com/datasec-lab/hpaa.

*Both authors contributed equally to this work.

A.2.3 Hardware dependencies

Most experiments, including attack generation and API-based evaluations, can be executed on a standard workstation. For experiments involving local detector models (e.g., Llama Guard and ShieldGemma), we recommend a GPU with at least 24 GB of GPU memory.

A.2.4 Software dependencies

The artifact has been tested on Ubuntu 22.04 LTS with Python 3.10. All required packages can be installed using the provided `requirements.txt` file.

A.2.5 Benchmarks

The artifact includes the datasets and evaluation resources used in the paper, including datasets for adversarial-example generation, user-study datasets and survey instruments, detector evaluation scripts, configuration files, and analysis utilities.

Evaluations are supported for both commercial moderation APIs (OpenAI Moderation, Google Perspective API, AWS Bedrock Guardrails, and Azure Content Safety) and local safety models (Llama Guard and ShieldGemma). Commercial API evaluations require valid API credentials.

All datasets, configuration files, and analysis scripts required to reproduce the reported experiments are included in the artifact package.

A.3 Set-up

Please confirm that the required software packages are installed on the system as stated above, before proceeding with the following steps.

A.3.1 Installation

Download and extract the artifact package:

```
unzip hpaa_artifact.zip
cd hpaa
```

Install the required dependencies:

```
pip install -r requirements.txt
```

For experiments involving local detector models (Llama Guard and ShieldGemma), install the optional dependencies:

```
pip install torch transformers accelerate
```

For experiments involving commercial moderation APIs, copy the API template and provide valid credentials:

```
cp .env.example .env
```

Edit `.env` and populate the required API keys.

For evaluations using local detector models, download the corresponding model weights and update the `download_path` field in `src/detectors.yaml` as described in the repository `run.sh` or `README`.

A.3.2 Basic Test

Run the quick single-sample test (Option A in `run.sh`). Successful execution generates the file `./HPAA/optA.M1-W-Hi.csv`. To generate adversarial examples under different settings, uncomment the corresponding options in `run.sh`. To reproduce the configurations evaluated in the paper, adjust -m (M1–M6), -l (W, T, Mix), and -s (B, Col, Hi, Pre, Cap, Cloze) parameters.

A.4 Evaluation workflow

A.4.1 Major Claims

(C1): HPAA generates human-perceptible adversarial examples by combining benign and toxic text inputs. This claim is supported by experiment (E1), which reproduces the adversarial-example generation pipeline described in Section 4 of the paper.

(C2): HPAA achieves strong evasion performance against both commercial moderation systems and local safety models. This claim is supported by experiment (E2), which reproduces the detector-evaluation results reported in Section 5.

(C3): HPAA preserves human recognition while reducing detector effectiveness, as demonstrated by the user-study results reported in Table 1 and Figure 8. This claim is supported by experiment (E3), which reproduces the corresponding user-study analyses.

A.4.2 Experiments

(E1): Adversarial Example Generation [10 human-minutes + 2 compute-minutes]
Preparation: Install the artifact and dependencies as described in Section A.3.1.
Execution: Run:

```
python HPAA.py
-bc Hotel -b text
-tc Advbench_10 -t text
-m M1 -l W -s Hi
--seed 42 -p adv
```

Results: Generated adversarial examples are saved to: `./HPAA/adv.M1-W-Hi.csv`

(E2): Detector Evaluation [30 human-minutes + variable compute-hours] The runtime depends on the number of adversarial examples, the selected detector, and API response latency.
Preparation: Generate adversarial examples using Experiment (E1). Configure API credentials or local detector models.
Execution: Run:

```
python HPAA.py
-f ./HPAA/adv.M1-W-Hi.csv
-dn perspective_api
-ep verify
```

Additional detectors can be evaluated using the commands provided in `run.sh`.
Results: The evaluation results are written to: `./HPAA/verify.*.csv`

(E3): User-Study Analysis [30 human-minutes + 2 compute-hour]
Preparation: No additional preparation is required.
Execution: Follow the instructions provided in:
`user_study/SurveyDataRound1/README.md`
`user_study/SurveyDataRound2/README.md`
Results: The analysis reproduces user-study statistics.

A.5 Notes on Reusability

The artifact supports extension to new datasets, typographic configurations, and moderation systems. New datasets can be incorporated by placing the corresponding files in the appropriate data directory and registering them in `./src/gen_HPAA.py`. Additional typographic configurations can be supported by extending the generation logic in `./src/gen_HPAA.py`. New moderation systems can be integrated by implementing detector wrappers in `./src/detectors.py` and `./src/eval_HPAA.py`, and registering the detector configuration in `./src/detectors.yaml`.

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/usenixsec2026/.