USENIX Security '25 Artifact Appendix: <Topic-FlipRAG: Topic-Orientated Adversarial Opinion Manipulation Attacks to Retrieval-Augmented Generation Models>

Yuyang Gong[1]*, Zhuo Chen[1], Jiawei Liu[1]†, Miaokun Chen[1],
Fengchang Yu[1], Wei Lu[1], XiaoFeng Wang[2], Xiaozhong Liu[3]
[1]Wuhan University, [2]Nanyang Technological University, [3]Worcester Polytechnic Institute

A Artifact Appendix

A.1 Abstract

*This artifact contains the complete implementation of the **Topic-FlipRAG** attack, along with all components necessary to reproduce the results presented in our USENIX Security paper. The repository includes: (1) the full attack codebase for Topic-FlipRAG and baseline methods, (2) evaluation scripts for computing opinion shift and retrieval consistency metrics, (3) ethically filtered synthetic datasets generated by GPT-4o, (4) comprehensive documentation for setup and usage, and (5) pre-generated poisoned document examples for quick evaluation. These resources enable reproducibility.*

A.2 Description & Requirements

The artifact is fully executable via Google Colab and does not require local installation. All experimental code is configured with default parameters that exactly match those used in the main experiments of our USENIX Security paper. No changes are needed to reproduce the core results.

The repository includes pre-generated adversarial documents to facilitate rapid evaluation. Reviewers may directly run the evaluation notebooks to verify the effects of Topic-FlipRAG without executing the full document poisoning pipeline, thereby reducing time and API usage.

A.2.1 Security, privacy, and ethical concerns

Describe any risk for evaluators while executing your artifact to their machines security, data privacy or others ethical concerns. This is particularly important if destructive steps are taken or security mechanisms are disabled during the execution.

A.2.2 How to access

All artifact code, data, and documentation are publicly available at the following locations:

- **GitHub repository (for AEC evaluation and iterative updates):** https://github.com/LauJames/Topic-FlipRAG
- **Zenodo archive (stable version for the camera-ready submission):** https://doi.org/10.5281/zenodo.15523434

The GitHub repository is actively maintained during the Artifact Evaluation period and may receive minor updates in response to reviewer feedback. The Zenodo snapshot will serve as the final archived version for the camera-ready artifact.

A.2.3 Hardware dependencies

The artifact runs entirely on Google Colab without the need for any specialized local hardware or remote infrastructure.

- **GPU:** NVIDIA T4 (for generating poisoned documents via Topic-FlipRAG), A100 (recommended for evaluating RAG system response)

All components execute within Colab with GPU acceleration enabled; no SSH access or external hardware setup is required.

A.2.4 Software dependencies

The artifact runs in the default Google Colab environment using Python 3.9+. All required dependencies are open-source and automatically installed via `pip` at the beginning of each notebook.

*Email: 2498002636gyy@gmail.com
†Corresponding author. Email: laujames2017@whu.edu.cn

Benchmarks

The artifact includes the dataset used for the opinion manipulation experiments. These consist of the PROCON dataset and GPT-4o–generated queries designed to simulate realistic manipulation tasks. All queries have been ethically filtered to ensure responsible usage and safe evaluation.

In addition, we provide pre-generated poisoned document examples used in our Topic-FlipRAG attack pipeline. These are included to help AEC reviewers quickly evaluate manipulation outcomes without running the full document poisoning process, thereby reducing computational cost and API usage.

A.3 Set-up

A.3.1 Installation

No manual installation is required. All dependencies are installed automatically when executing the code blocks in the provided Colab notebooks.

A.3.2 Basic Test

To perform a quick test, please follow the instructions in the GitHub repository's `README.md`. Replace the relevant path variables with the provided example data files. Default parameters have been pre-configured in the Colab notebooks, so reviewers can directly run the cells without any modification.

A.4 Evaluation workflow

The evaluation workflow is provided in `RAG_pipeline.ipynb`, which includes code for assessing both ranking manipulation and opinion manipulation. Reviewers can directly execute this notebook in Google Colab to reproduce the main experimental results described in the paper.

A.4.1 Major Claims

(C1): *Topic-FlipRAG, under black-box conditions, effectively boosts the rankings of target documents across queries within the retriever module of RAG. This is demonstrated by experiment (E1,E2,E3) described in Section 6.1, with results shown in Table 2.*

(C2): *Topic-FlipRAG significantly affects the answers generated by the target RAG systems. This is supported by experiment (E1,E2,E3) described in Section 6.2, with results shown in Table 3 and Table 11 (Appendix).*

A.4.2 Experiments

(E1): *[Knowledge-Guided attack] [5 human-minutes + 45 compute-minutes per topic + OpenAI API + NVIDIA T4]: This stage corresponds to generating poisoned documents (doc_know) aligned with the target opinions, as described in Section 4.2. These are used in subsequent trigger optimization and evaluation steps.*
How to: *We recommend testing on a single topic to reduce API costs and runtime. The generated doc_know will serve as input to Stage 2. Even if this step is skipped, subsequent evaluation can still be performed using our pre-generated example data.*
Preparation: *Open Stage 1 Notebook in Google Colab. Ensure GPU (T4) is enabled. Install dependencies via the first code cell. Place `PROCON_data.json` from the GitHub repo in the required data path. An OpenAI API key is also needed.*
Execution: *Run the notebook as instructed. Select a single topic and execute the document generation blocks.*
Results: *The notebook will output a doc_know JSON file for the selected topic, which can be saved to Colab or Google Drive. This file is used as input to Stage 2 (E2).*

(E2): *[Adversarial Trigger Generation] [5 human-minutes + 75 compute-minutes per topic (5 passages per topic, 42 topics total) + NVIDIA T4]: This stage corresponds to generating adversarial triggers based on the doc_know files produced in Stage 1 (E1), as described in Section 4.3 of the paper. The output is a poisoned document file (doc_adv) used in final RAG evaluation.*
How to: *Follow the GitHub `README.md` to run the corresponding Colab notebook for Stage 2. Even if E1 is not executed, we provide example doc_know files that can be used to run E2 directly.*
Preparation: *Open Stage 2 Notebook in Colab. Enable GPU (T4), install dependencies, and ensure `PROCON_data.json` is placed in the correct path. For quick testing, update the input path to use the provided example doc_know file, as explained in the README.*
Execution: *Run all cells in the notebook. The script will optimize adversarial triggers for five passages per topic based on the input doc_know.*
Results: *The output is a doc_adv file containing adversarially modified documents, stored locally or in Drive. This file will be used as input in Stage 3 (E3) to evaluate attack effectiveness.*

(E3): *[RAG Pipeline and Evaluation] [5 human-minutes + 3 compute-minutes per topic + A100 GPU + OpenAI API]: This stage involves injecting adversarial documents into the RAG system and evaluating downstream impacts, as described in Section 6. This step validates Claims C1 and C2 through end-to-end experiments.*
How to: *Follow the instructions in the GitHub `README.md` to run the `RAG_pipeline.ipynb` notebook.*
Preparation: *Launch the notebook in Google Colab and ensure GPU (A100) is enabled. Install dependencies by running the setup cells. Place*

`PROCON_data.json` in the correct path. To simplify evaluation, we provide pre-generated poisoned passages (doc_adv) for 9 topics in the "Society" domain with target stance "CON." Update the input path as described in the README.

Execution: *Run all cells in the notebook. This step loads the poisoned documents, runs the retrieval-augmented generation process, and computes evaluation metrics.*

Results: *The expected metrics, based on our paper results using the provided poisoned examples (Society domain, target stance: CON), are: RASR = 51.9, Top3-v = 25.57, and ASV = 0.55 (ranging approximately from 0.5 to 0.6). These results were obtained using the **Qwen2.5** LLM with a **Contriever** retriever setup, and demonstrate the manipulation effectiveness of Topic-FlipRAG in a full RAG pipeline.*

If you wish to evaluate other combinations of LLMs and retrievers, please modify the corresponding configuration parameters in the notebook. Refer to the experimental comparisons reported in the main paper for baseline expectations across different model setups.

A.5 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/.