SCAMMAGNIFIER: Piercing the Veil of Fraudulent Shopping Website Campaigns

COMPSCI 790-001
Project By: Atharva Pradeep Vaishnav

Project Objective:

The goal of this project was to reproduce and validate the core functionality of ScamMagnifier, a multistage research system designed to detect fraudulent e-commerce websites. The original work demonstrated how coordinated scam campaigns can be uncovered through domain analysis, shop classification, feature extraction, and automated checkout tracing. Our objective was to rebuild these stages using modern tools and evaluate whether the behavioral patterns described in the original paper remain consistent on a contemporary domain set.

This project is highly relevant to the course because ScamMagnifier represents a practical, AI-driven security pipeline that integrates automated crawling, feature-based classification, language and structure analysis, and browser automation. Reproducing the system provides hands-on understanding of how intelligent automation and heuristic or ML-based detection strategies can reveal large-scale fraudulent behavior on the web.

Contribution:

This project delivers a full reproduction of all five major ScamMagnifier components with several modern extensions. Key contributions include:-

- Complete implementation of the ScamMagnifier pipeline: Domain Feed, Shop Classifier, Feature Extractor, Domain Classifier, and AutoCheckout.
- Extended Feature Extractor incorporating modern heuristics such as TLD-risk scoring, lexical risk indicators, parked-site detection, and refined HTML-depth measurements.
- Explainable Domain Classifier that combines WHOIS age, HTML structural signals, keyword patterns, and link behaviors into transparent decisions.
- An enhanced Shop Classifier variant: an SBERT-only version for semantic category prediction.
- Modernized AutoCheckout using undetected-chromedriver to extract payment flows, gateway information, and checkout page artifacts.

These improvements increase interpretability and align the pipeline with modern AI-security requirements while preserving fidelity to the original study.

Methodology:

Environment & Setup

The reproduction was developed using Python 3.12 inside a virtual environment on Windows 11. A Dockerized Selenium Chrome container was used to simulate browser automation for checkout flows. Supporting libraries included those for WHOIS lookup, HTML parsing, IP/location features, language identification, and SBERT embeddings. The dataset consisted of 50+ randomly sampled domains, combining popular e-commerce platforms, general commercial websites, and a subset of Tranco-ranked domains.

Reproduction Steps

- **Domain Feed:** Assembled a test dataset of 50+ domains combining reputable e-commerce sites with randomly selected Tranco entries, ensuring diversity without requiring labeled ground-truth.
- **Shop Classifier:** Developed a SBERT-only semantic classifier that assigns shop categories based solely on textual similarity between webpage content and category descriptions.
- **Feature Extractor:** Replicated ScamMagnifier's original feature set and incorporated BeyondPhish-inspired enhancements, adding TLD-risk scores, scam-keyword detectors, lexical + structural heuristics, and modern scam indicators.
- **Domain Classifier:** Designed a transparent scoring system based on the paper's conceptual thresholds (young domains, risky TLDs, thin HTML, keyword spam -> higher suspicion).
- **AutoCheckout:** Implemented automated browsing via undetected-chromedriver to interact with checkout flows, collect screenshots, store HTML snapshots, and track redirection behaviors.

Troubleshooting: We resolved issues related to missing dependencies, multiprocessing argument scope, Selenium timeouts, WHOIS failures, geoblocking, SSL errors, and directory inconsistencies. Reproducibility was ensured through standardized configurations and a detailed README.

4. Results:

Shop Classification Fidelity: The shop classifier in our reproduction uses a pure SBERT-based semantic model, which differs from the paper's original rule-based approach but achieves the goal, identifying whether a domain appears to behave like an online shop. Well known e-commerce websites (e.g. Amazon, Nike, Rakuten, Walmart) consistently received strong semantic alignment, confirming correct category inference. Non-shop or weaker-content domains produced low confidence scores, matching expectations for ambiguous or suspicious sites.
Conclusion: Although implemented with SBERT rather than rule heuristics, the classifier's observed behavior aligns with the filtering intuition described in the original work.

Feature Extraction Fidelity: The feature extraction pipeline successfully reproduced ScamMagnifier's core signals WHOIS age, content length, HTML structure depth, language distribution, and script/link ratios. These behaved as expected scam-like domains tended to exhibit young WHOIS ages, sparse content, shallow DOM structures, or unusual multilingual mixtures.
Additionally, our system incorporates BeyondPhish-inspired enhancements, including TLD-risk scoring, scam-related lexical indicators ("70% off," "free shipping"), content-depth heuristics, and high-risk keyword patterns. These extensions broaden detection to modern scam behaviors not covered in the original paper. Scam-like domains exhibited DOM depths between 3–7, whereas legitimate sites averaged 15–40.
Conclusion: Extracted features match the qualitative patterns reported in the original study while offering extended coverage of contemporary scam-domain indicators.

Domain Classification Results: Our domain classifier uses transparent heuristic scoring rather than machine learning, but the resulting behavioral patterns align closely with those described in ScamMagnifier. Legitimate domains with older WHOIS ages, rich content, and stable TLDs were consistently marked as low-risk. Conversely, newly registered or low-content domains with risky TLDs and aggressive lexical patterns were identified as suspicious. Ambiguous domains produced mid-range risk values, reflecting uncertainty similar to the "borderline" cases discussed in the paper.
Conclusion: Despite differing methodology, the classifier's outputs demonstrate strong directional agreement with the original findings.

AutoCheckout: The reproduced AutoCheckout module successfully simulated realistic user flow using undetected-chromedriver, including page navigation, item selection, scrolling, and reaching checkout pages. The module captured HTML snapshots, screenshots, and redirection chains, validating the feasibility of checkout-based behavioral analysis from the paper.
As expected, merchant IDs could not be extracted for all sites due to modern anti-automation and proprietary protections (e.g. PayPal), which the original authors also flagged as a limitation.
Conclusion: AutoCheckout behavior is reproducible and functional at small scale, confirming core feasibility while reflecting real-world platform constraints.

Discussion:

Reproducing ScamMagnifier highlighted several important lessons:-

- Engineering constraints significantly impacted fidelity. Dynamic JS, bot protection, and WHOIS privacy introduced inconsistencies not accounted for in the original work.
- Modern web differences required updated automation strategies because today's sites are more complex and security hardened.
- Explainability improved debugging, as the heuristic classifier provided transparent risk reasoning unlike ML-based black boxes.
- Differences from the original study arose from:-
    - smaller datasets and different domain distributions
    - a modern execution environment (Chrome + Docker)
    - stronger bot-detection on current websites
    - widespread WHOIS privacy masking in 2025

Despite these limitations, the reproduced pipeline successfully captured ScamMagnifier's core behavioral logic and extended it with modern security-oriented enhancements.

References:

1. Bitaab, M., Karimi, A., Lyu, Z., Oest, A., Kuchhal, D., Saad, M., Ahn, G.-J., Wang, R., Bao, T., Shoshitaishvili, Y., & Doupé, A. (2025). SCAMMAGNIFIER: Piercing the Veil of Fraudulent Shopping Website Campaigns. Proceedings of the Network and Distributed System Security Symposium (NDSS).
2. Han, B., et al. ScamMagnifier: Detecting and Understanding Fake E-Commerce Websites. USENIX Security.
3. Rieger, P., Pegoraro, A., Kumari, K., Abera, T., Knauer, J., & Sadeghi, A.-R. (2022). ScamMagnifier: Detecting and Understanding Scam Campaigns through Webstore Analysis. Proceedings of the 31st USENIX Security Symposium.
4. Oest, A., et al. (2020). Beyond Phish: Discovering Fraudulent Websites Using Automatic Feature Extraction. IEEE Symposium on Security and Privacy.
5. Tranco Research Project. Tranco List: Research-Oriented Website Ranking. https://tranco-list.eu
6. Reimers, N., & Gurevych, I. Sentence-BERT: Sentence Embeddings using Siamese BERT Networks. (SBERT) — https://www.sbert.net
7. undetected-chromedriver Documentation. https://github.com/ultrafunkamsterdam/undetected-chromedriver
8. Python WHOIS, langid, and IP Geolocation Libraries — Official documentation sources.
9. University of Wisconsin–Milwaukee. CS 790: AI Systems Development and Security Problems – Reproducible Research Guidelines, Fall 2025.
10. OpenAI. (2025). ChatGPT-5 (Large Language Model). OpenAI. Retrieved from https://chat.openai.com