Experience Paper: Is This the Real Life? Is This Just Fantasy? Hunting for Hallucinated References in Paper Submissions

Gianluca Stringhini
Boston University
gian@bu.edu

Jeremy Blackburn
Binghamton University
jblackbu@binghamton.edu

***Abstract*—The scientific community is getting flooded by papers that are either fully or partially generated by AI, containing sloppy, fabricated, or completely hallucinated claims. While it can be difficult to conclusively determine if content is AI generated, academic references pointing to non-existing work are a clear sign of AI usage that we can reliably measure. We present HALLUCINATOR, a system that extracts references from academic PDFs and checks them with multiple reference databases, helping users identify non-existing references. We discuss the capabilities of our system and our experience in developing and running it as Program Chairs of large computer science conferences. HALLUCINATOR is open source and publicly available, and we encourage the academic community to adopt it and improve it to ensure that the body of scientific literature in our conferences is protected from hallucinated references.**

1. Introduction

In the past year, we have seen a rise in academic papers containing references that do not exist, likely due to the heavy reliance on Generative Artificial Intelligence (Gen AI) tools during paper preparation. Recent research found over 300 papers containing hallucinated references at NLP venues (ACL, NAACL, EMNLP) between 2024 and 2025 [1] and over 100 citations from 50 papers at NeurIps 2025 [2]. Cybersecurity conferences are not immune from this issue, although the problem has not reached the level of machine learning venues yet. The Program Chairs at USENIX Security 2026 reported they desk rejected 21 out of 1,181 valid submission in the first submission cycle alone (1.78% of the total) due to references pointing to non-existing papers [3].

Hallucinated references are just the symptom of a bigger problem. Anecdotal evidence shows that AI is heavily used in paper preparation, with a recent analysis estimating that 199 ICLR 2025 manuscripts were fully AI generated [4]. There is no question that AI can help authors improve the clarity of their papers, and recent policies like the ACM's [5] allow the use of AI in paper preparation given that 1) all uses of generative AI are disclosed in the paper and 2) the authors check for the accuracy and correctness of any analysis or claim made in the paper. Hallucinated references left unchecked clearly break the latter rule. A similar issue can arise during peer review, where reviewers ask AI to generate a full review for a paper. Recent analysis found that 21% of all submitted reviews at ICLR 2025 were also AI generated [6]. Keeping aside the privacy concerns of uploading confidential material to an untrusted third party, the model can make mistakes and also include research work that does not exist in the review text.

Why does this happen? The source of the problem is authors asking a Gen AI tool to produce a reference for claims instead of conducting a thorough literature review or even checking that the returned reference exists. Of course, there are different levels of AI use and misconduct. Authors could use AI to fully generate their paper, and hallucinated references would come with that. While this seems to be happening in AI and ML venues, we have not encountered cybersecurity papers for which we are confident that they were 100% AI generated. As an exercise, we prompted two AI agents, Claude Code Opus 4.5 and OpenAI Codex 5.3, to "*generate an entire survey paper on the academic field of online social network abuse.*" Both models included several hallucinated references in the produced PDF. For example, Codex generated a citation to a non-existing reference in the paper (see entry 1 in Table 1), while Claude generated a reference to a paper that does exist, but attributed it to the wrong set of authors (see entry 2 in Table 1).

In a less extreme (but still worrying) case, authors could ask AI to generate full paragraphs in their paper, to help them meet the strict deadline set by the Program Chairs. As it produces the text, the AI adds references to back up the claims it made, but these references do not exist. We observed this happening multiple times in Related Work sections, where all the references in one or more paragraphs were hallucinated. To demonstrate it, we asked ChatGPT 5.2 to write the Related Work for a paper we were writing, saying that the goal of the paper was to "*design a system to detect memory corruption vulnerabilities in smart refrigerators.*" The model happily complied, but while doing so it generated multiple non-

existing references, including entry three in Table 1. In less extreme cases, the model will produce paper titles that exist, but will get the author list completely wrong. Entry 4 in Table 1 shows an example of a reference of this type produced upon asking "*Find me references to prominent fuzzing papers.*" The entry in the table is followed by the correct bibliographic information for that paper as reported by Google Scholar.

Finally, authors might ask the AI to produce just one reference to back up a claim made in the paper, perhaps to address the feedback received by the faculty advisor. From our experience, when asked an overly narrow question about a study that may or may not exist, AI will most likely produce a hallucinated reference. Entry 5 in Table 1 shows an example of a hallucinated reference produced by Llama 4 when asked to "*Find an academic reference to a paper that studied the victimization prevalence in cybercrime in the US and the UK.*"

Our philosophy. Regardless of the reason why they slip into papers, we argue that hallucinated references are scientific misconduct. As authors, it is our responsibility to ensure that all content we include in our papers is accurate [5]. As Program Committee chairs and members, it is our job to identify papers that include hallucinated references and take strong action. Hallucinated references are just a symptom of potentially broader and undisclosed AI usage: if we cannot trust the references that a paper cites and potentially builds upon, how can we trust anything else that is reported in it? Even more so, allowing hallucinated references to poison our literature can have detrimental downstream effects, like being cited in follow up research and having the spurious claims made around the hallucinated reference become "canon" in the field. This will take the phenomenon known as citation distortion [7] to the extreme, eventually poisoning the entire field with unfounded and bogus claims. This is the reason why Program Chairs of prominent cybersecurity conferences (IEEE S&P, ACM CCS, USENIX Security, ACSAC) have started desk rejecting papers that contain hallucinated references. Venues from other computer science areas, e.g., ACM CHI, ACM CSCW, and AAAI ICWSM have also followed suit with desk rejects and policies directly addressing hallucinated references.

References have the advantage that they can be checked in a systematic way, enabling fully explainable and interpretable results. We can extract their details from papers and cross-reference them with online repositories of published research (e.g., DBLP). We built a tool that does precisely this, enabling authors and reviewers to check the references in a paper and confirm if they are hallucinated. As it turns out, PDF parsing is hard and matching records against a diverse set of online resources is non trivial. Being busy professors whose joint body of work [8, 9, 10, 11] has resulted in a well developed sense of irony, we turned to Claude Code (*extensively*) to help us build this tool. It turns out that AI is pretty good at developing and testing regular expressions.

In this paper, we present HALLUCINATOR, our hallucination detection tool, and we report our results running it as Program Chairs of two prestigious academic conferences (ACSAC 2025 and ICWSM 2026). We then discuss the implications of releasing our tool to the public and how we envision hallucinated reference detection to evolve in the coming years.

2. Methodology

When looking up a reference against the databases, we consider it a match if 1) the title is the same and 2) the set of authors is the same. We have not implemented a year and venue match yet, in part because the same venue can be referred to in various ways (e.g., Oakland, IEEE Symposium on Security and Privacy, IEEE S&P, etc.) and because these days papers are often uploaded to arXiv first, and online resources like Google Scholar are unable to easily resolve where a paper was published when multiple online versions are found. In addition to title and authors, HALLUCINATOR looks up DOI numbers and arXiv IDs when provided, because we found that hallucinated references often include correct information but the links to DOIs and arXiv actually point to a different paper.

We encountered several challenges when developing HALLUCINATOR:

PDF parsing and DB matching. Parsing references correctly from a PDF turned out to be more difficult than expected. Where does the author list end and the title start? How are subtitles handled? What if the venue name is not clearly delimited? This is particularly challenging, especially when several citation formats need to be supported (e.g., IEEE, ACM, and USENIX for cybersecurity venues) and authors may use their own formatting style.

We experimented with existing PDF parsing tools (MuPDF, Grobid), but none of them is able to reliably extract references from PDFs. Eventually, we decided to use MuPDF as PDF parser, but we developed reference extraction logic in HALLUCINATOR to handle 20 reference formats (including IEEE, USENIX, ACM, and Springer). Among other things, this extraction logic handles hyphenations introduced by new lines in PDFs.

On top of this, the extracted title might not match the reference returned by a DB, for example because the two strings encode special characters differently, or the DB only indexes the main title of a paper and not the subtitle. Other reasons for discrepancies are when DBs and papers use a different convention to list authors (e.g., listing first names before last names or vice versa).

DB rate limiting. Some online references aggressively limit the number of queries that HALLUCINATOR can perform. We did our best ensuring that limits are not hit, allowing the user to set optional API keys for those services that offer them, allowing for the download and

#	Source	Title	Authors	Venue	Year
1	Codex 5.3	The rise of a new type of spammers in Twitter	Metaxas, Panagiotis; Mustafaraj, Eni; Gayo-Avello, Daniel	First Monday 20(6)	2015
2.1	Claude Opus 4.5	Detecting malicious web links and identifying their attack types	Wang, Gang; Mohanlal, Manish; Wilson, Christo; Wang, Xiao; Metzger, Miriam; Zheng, Haitao; Zhao, Ben Y	USENIX Conference on Web application Development	2011
2.2	Google Scholar	Detecting malicious web links and identifying their attack types	Choi, Hyunsang and Zhu, Bin B and Lee, Heejo	USENIX Conference on Web application Development	2011
3	ChatGPT 5.2	AVATAR: A Framework to Support Dynamic Security Analysis of Embedded Systems' Firmwares	Zhang, Feng; Zhang, Zhiqiang; Xu, Dongyan; Xu, Shouhuai	ACM SIGSAC Conference on Computer and Communications Security (CCS)	2017
4.1	ChatGPT 5.2	QSYM: A Practical Concolic Execution Engine Tailored for Hybrid Fuzzing	Yun, Sang Kil; Kim, Junghwan R.; Lee, Kwangkeun; Woo, Maverick; Paek, Il	USENIX Security Symposium	2018
4.2	Google Scholar	QSYM: A practical concolic execution engine tailored for hybrid fuzzing	Yun, Insu; Lee, Sangho; Xu, Meng; Jang, Yeongjin; Kim, Taesoo	USENIX Security Symposium	2018
5	Llama 4	Assessing the impact of cybercrime: A review of the literature	Gill, Mark	Crime Prevention & Community Safety	2007

TABLE 1: Examples of hallucinated references. References marked in red indicate that the paper does not exist, while references marked in Yellow indicate that the paper exists, but the list of authors was hallucinated by the AI. The Google Scholar entries show the correct author set for the corresponding AI-generated one.

querying of local databases where possible (i.e., DBLP and ACL Anthology), as well as a caching layer to avoid making duplicate queries.

Supported DBs. We have implemented support for nine online sources against which HALLUCINATOR can check reference details. Table 2 lists them all, showing whether they offer an offline version where users can download a local DB (much faster for lookups). Our sources include computer science-specific databases like DBLP, preprint services like arXiv, general purpose publication databases like CrossRef, Semantic Scholar, and OpenAlex, and specialized libraries like ACL Anthology, PubMed, and Open PMC. To help identifying sources that exist but are not proper academic papers, we also implemented logic to issue search engine queries through SearxNG. HALLUCINATOR marks the papers returns in this manner as "found on the Web," highlighting that this form of vetting is weaker than having a fully-verified reference on an academic repository.

DB coverage. No single online reference covers all published papers. For computer science venues, DBLP is the most comprehensive DB, but technical reports are often not found in any of the venues. Things become even trickier when HALLUCINATOR encounters hallucinations from other fields (e.g., law or social sciences), because papers from certain publishers are not listed in any online reference. Note that while Google Scholar is by far the most comprehensive database for academic papers, we are unable to query it in an automated fashion because of their aggressive anti-bot measures. Instead, we include a pre-populated Google Scholar link with any suspected hallucinated reference, so that the user can visit the site manually and verify potential false positives.

Ground truth. To develop HALLUCINATOR, we started with a set of papers authored by ourselves for which we both had a compiled PDF and the source bibtex file for the bibliography. We then tasked Claude Code to improve the system's parsing until the extracted titles matched the entries in the bibtex files.

While testing on papers that we found in the wild, it soon became clear that this is not sufficient, because new corner cases and exceptions in PDF encoding and reference handling keep appearing. We therefore developed a daily crawler that collects the latest submissions from

arXiv. Our crawler collects the last 100 submissions from several arXiv repositories (e.g., cs.CR for Cryptography and Security), downloading both the PDF and its LaTeX source. This allows the Claude Code agent to perform the same optimization discussed above, matching extracted reference data with bibtex information. The agent refines the set of regular expressions to handle the corner cases it encounters, and checks them against old data to make sure that introducing them does not break the old parsers. The updated regular expressions are then incorporated in the analysis engine.

3. Implementing HALLUCINATOR

While the high level methodology that HALLUCINATOR uses is relatively straight forward, as often is the case, implementation is a different story. Moreover, for a system like HALLUCINATOR, there are a surprising amount of non-trivial decisions to make in terms of architecture. In this section we present the design and implementation of HALLUCINATOR, with a particular focus on the development strategy we took to go from an initial commit on January 21, 2026 until the feature rich system it is as of April 3, 2026.

The use of generative AI. From the first commit, nearly every line of code in HALLUCINATOR was written by an AI code assistant (almost exclusively Claude Code Opus 4.5/4.6). To that end, we have decided to keep a full record of every commit pushed to our repo to provide a full and accurate history of what, exactly, the code assistant did at any given time. Quite frankly, it is very unlikely that HALLUCINATOR would exist at all if not for our reliance on AI.

That said, we *did* observe a commonly reported issue with AI code assistants: while it is easy to get *something* up and running and produce copious amounts of code, as things move further and further out of distribution, more direct supervision (and more importantly, *planning*) must take place. As a concrete example, while it took maybe half a day to get HALLUCINATOR's new Terminal User Interface running (something that would likely have taken weeks unassisted, if it happened at all), it took several days (most of which was spent supervising the AI code assistant) to get the rate limiting system in place (something we have written ourselves numerous times in the past). As another example, the original concurrency model the code assistant came up with was, to put it bluntly, bad, making use of shared state (almost exclusively), which overly complicated the code and made it much more difficult to add new databases to check, especially considering the overwhelming majority of HALLUCINATOR's functionality is I/O bound (i.e., async).

Rewrite it in Rust. The original proof-of-concept for HALLUCINATOR was written in Python. A Web UI was eventually built on top of the script, and some more databases added. However, it quickly became apparent that while Python was suitable for running HALLUCINATOR ourselves, the fact that you needed to setup a Python environment was a serious barrier to other people using it. We made a decision: HALLUCINATOR should be distributed as a single binary. While there are several ways to package Python scripts into self contained binaries, none of them are particularly well maintained. Moreover, Python was just not meant to fill this use case.

On February 7th, 2026, we began porting over the original Python code base to Rust. There were several, complimentary reasons we made this change.

First, Rust is statically compiled and ends up in a single binary, so it is easy to distribute; no custom environment needed. Next, Rust seems to mitigate (to some extent at least) issues that AI assistants tend to make: Rust code simply will not compile if it is incorrect (i.e., type checks, exhaustive branches, borrow checker, etc.). As a nice bonus, Rust tooling is known for having *excellent* error messages, which often tell a programmer (or AI assistant) exactly how to fix whatever compiler check their code failed. Third, Rust is performant and its so called "fearless concurrency" gives us primitives and a host of libraries that enable us to easily handle thousands of reference checks lightning fast. Finally, Rust allows us to (pretty easily) expose Python bindings, giving us and others the ability to use various HALLUCINATOR components in a variety of different applications.

System Architecture. In a nutshell, HALLUCINATOR is organized as a set of crates (libraries) that expose functionality related to extracting and validating references.

The `hallucinator-core` crate includes types like `Reference` (which represents a reference), trait definitions like `DatabaseBackend` and some concrete implementations (e.g., CrossRef), our rate limiting system, concurrency, etc.

Around this core, there are crates designed for streaming archive extraction and file type detection (`hallucinator-ingest`), to handle BBL and BIB files (`hallucinator-bbl`), reporting (`hallucinator-reporting`), and special crates for building and using offline databases (`hallucinator-dblp` and `hallucinator-dblp`), as well as a crate for parsing PDFs (`hallucinator-pdf`).

One benefit of this design is that it allows, e.g., the back end PDF engine to be swapped out. For our initial release, we use `MuPDF`, which is AGPL licensed. While we have no particular issue with the AGPL, it *is* a contagious license. I.e., any program that it is statically compiled into (e.g., a typical Rust program) must also be AGPL. Our design, however, allows for MuPDF to be completely swapped out, or not even compiled in at all.

This occurs naturally because the `halucinator-core` crate exposes traits which define a common interface for `PdfBackend`s that other crates (e.g., `hallucinator-pdf-mupdf`) can implement for the specific PDF parsing tool they use. We can then *feature gate* the use of MuPDF (or any other PDF engine) at compile time using Rust's standard "feature" compilation strategy. Swapping in another `PdfBackend` is as simple as selecting a different feature when defining dependencies. Although

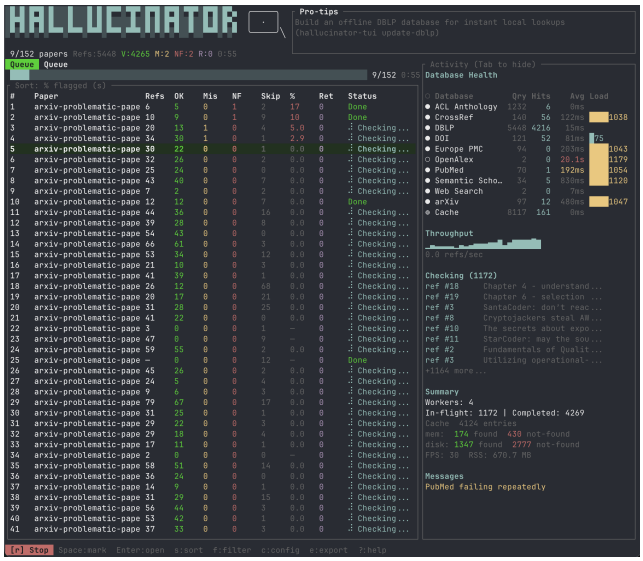

Figure 1: HALLUCINATOR's queue screen showing a batch of papers being processed. The TUI is designed to give feedback on what might be batches of hundreds of papers being processed.

Figure 2: Example summary of a run of HALLUCINATOR's Web interface.

Source	Offline	Queries	Matches
DBLP	✓	2,641	1,941
arXiv	✗	661	83
CrossRef	✗	474	158
Semantic Scholar	✗	345	104
ACL Anthology	✓	700	0
Europe PMC	✗	578	84
PubMed	✗	474	19
OpenAlex	✓	0	0
SearxNG (Web)	✗	169	66

TABLE 2: List of sources currently supported by HALLUCINATOR with sample lookup rates. OpenAlex was down the day of the deadline.

we currently only support MuPDF, we are actively investigating the use of other PDF engines.

In the spirit of keeping HALLUCINATOR modular and generic, we allow the user to bypass the PDF extraction step by providing bibtex files to the tool directly (.bib and .bbl). This could be used by publishers that require document sources while compiling proceedings (e.g., the ACM) or by authors to make sure that their papers do not contain hallucinations.

Make beautiful software. HALLUCINATOR is *not* designed (nor expected) to be a perfect classifier. It is designed to be a tool that humans use to help them find problematic papers. Thus, we set out to build a Terminal User Interface (TUI) that was easy to use, scalable enough to handle hundreds (or even thousands) of papers, and looked good while doing it. HALLUCINATOR's TUI (see Figure 1) provides a full, interactive experience that not only lets users run checks themselves, but also serves as an annotation tool where results can be imported, allowing, e.g., results to be distributed to Vice Chairs for closer scrutiny without requiring them to run checks on the papers themselves. Figure 3 shows an example of the HALLUCINATOR's paper detail screen, providing information about not found references and allowing the user manually to check those results.

From the screenshot in Figure 1, you can see most of HALLUCINATOR's features in effect. E.g., its rate limiting system is in full effect, showing quite a bit of back pressure for many of the online databases we query. The screenshot also displays the effects of offline DBLP and ACL databases (over 4,000 references were found in DBLP alone, with an average response time of 15ms), our two layer caching system that has both an in-memory and optional persistent cache to improve processing speed across different runs of the program, etc. Anecdotally, the people we have shown HALLUCINATOR to have found it surprisingly fun to use; as it turns out, blinkenlights are both useful and aesthetically pleasing.

We also include a Web interface where users can direct their browser to HALLUCINATOR, drag and drop PDFs in the dialog for analysis, and receive nicely formatted reports (see Figure 2).

4. Results

We first evaluate HALLUCINATOR's throughput. We then evaluate HALLUCINATOR in two settings: 1) as Program Chairs of prominent conferences to help us identify papers with hallucinated references and desk rejected them; and 2) on the dataset of 50 papers accepted at NeurIPS 2025, which were found to contain 100 hallucinated references by GPTZero [2].

4.1. Performance analysis

Using the latest version of the TUI, HALLUCINATOR was able to parse the 187 camera ready papers that were presented at the NDSS 2026 conference in 22 minutes. These papers contained a total of 9,777 citations, setting the system's throughput to 7.4 citations per second. The main bottleneck are rate limits imposed by third party repositories (e.g., CrossRef, SemanticScholar).

4.2. Running HALLUCINATOR as Program Chairs

We ran HALLUCINATOR as Program Chairs of two prominent academic conferences, one in the cybersecurity

Venue	Submissions	Papers with Halluc. Refs
ACSAC 2025	408	8
ICWSM 2026 (Jan Cycle)	298	11

TABLE 3: Prevalence of hallucinated references across venues.

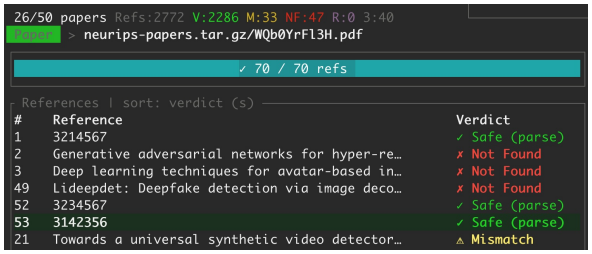

Figure 3: Example summary of a run of HALLUCINATOR's TUI interface. The user can browse a reference's details by pressing enter and marking the reference as safe by pressing tab.

field (ACSAC 2025) and another one in social computing (ICWSM 2026). For each reference flagged, we checked manually for its correctness and existence using Google Scholar. While this process can sound painstakingly tedious, HALLUCINATOR's interface makes it easy to quickly sift through false positives, identify real hallucinations, and generate detailed reports (see Figure 3 for an example of the TUI's paper detail interface). For ACSAC, we found 8 papers out of 408 submitted ones containing multiple fabricated references, and we proceeded with desk rejecting them. This is 1.9% of all submitted papers. For ICWSM's January submission cycle, out of the approximately 300 submissions we checked, 11 (3.6%) had hallucinated citations. One of those hallucinated citations was so plausible that another PC chair (who the citation was attributed to) had to check their own publication record to verify it was just a very similar title to one of their real papers.

One interesting phenomenon that we discovered during this process was that some hallucinated references *do* sometimes appear on Google Scholar. When digging deeper, said hallucinated references do not seem to have PDFs available, and the papers that cite them are themselves generated by AI and wound up indexed by Google Scholar due to being published on arXiv.

4.3. Testing HALLUCINATOR on the GPTZero NeurIPS dataset

Testing on GPTzero's NeurIPS dataset [2] is an interesting benchmark for HALLUCINATOR. First, this dataset contains known hallucinated references and allows us to benchmark our system against a commercial baseline. Second, HALLUCINATOR was designed and tuned to parse references and detect hallucinations in cybersecurity papers, and testing it on a different field can give confidence of the reliability of the system.

HALLUCINATOR extracted a total of 2,649 references from 50 papers. Of these references, 2,469 were automatically verified as existing, 8 were skipped as non academic (e.g., blog posts or news articles), 103 were marked as non-existing papers, and 69 matched the title of the reference but returned a different set of authors.

We manually checked the references with the help of HALLUCINATOR's TUI. As mentioned, this allows to cross-check flagged entries with Google Scholar, and allowed us to sort through them in a matter of minutes. We found three main reasons for false positives. For 10 entries, HALLUCINATOR correctly extracted reference data for existing papers, but those papers were not present in the sources that our system queries, and failed to return exact Web searches. These include references to old papers that are only available on Google Scholar as citations and not PDFs (but recall that we cannot query Google Scholar due to rate limits) or links to datasets.

Another category of errors is due to parsing failures. While HALLUCINATOR is able to correctly extract the vast majority of references from PDFs, authors compile their PDFs in various ways and with different settings, and this leads to errors either extracting paper details or matching them against third party sources. These errors include hyphenation problems, authors mis-parsed as part of the title, or journal names spilling into the title. In total, we observed 32 issues of this kind.

Finally, we sometimes observe a mismatch between the author list extracted by the PDF and the one returned by the online sources. We identified three reasons for this: first, when referring to chapters in edited books the editor of the volume is often listed instead of the authors of the chapter. Second, when dealing with reports released by companies or agencies sometimes the name of the company/agency is listed instead of the set of authors. This is for example the case of the GPT4 and DeepSeek whitepapers, which we observed 4 times and 2 times in our dataset respectively. Third, we found that some online references like CrossRef sometimes return an empty set of authors for the references they found, causing false positives in HALLUCINATOR. We observe this happen for 8 references, and we are in the process of fixing this bug as we write. In total, we observed 23 errors of this kind.

While HALLUCINATOR's design allows the user to quickly swift through potential hallucinated references and clear false positives, we acknowledge that lowering false positives is a priority. We are continuously improving out system. The detection performance has improved greatly from a month ago, and it will likely improve greatly by the time we will present this work at the workshop in May.

After vetting, we confirmed that HALLUCINATOR correctly identified 107 hallucinated references in the GPTzero dataset. We manually compared our detections with those of GPTzero to evaluate the accuracy of HALLUCINATOR and potential improvements over the state of the art. HALLUCINATOR did not correctly parse the

title of 3 references among the 100 flagged by GPTZero. For 26 references flagged by GPTZero, by the time we ran HALLUCINATOR on those papers their proceedings version had been updated and the references corrected; HALLUCINATOR correctly verified those references. The remaining 71 references were correctly flagged as hallucinated by HALLUCINATOR. On top of that, our system flagged 36 additional hallucinated references compared to GPTZero. This shows HALLUCINATOR's potential in helping the community identify and handle fabricated references at scale.

5. Discussion and Conclusion

We believe that the stance of the research community should be clear: hallucinated references are academic misconduct and papers that contain them cannot be trusted and must be desk rejected. Hallucinated references can be a symptom of a deeper problem with the paper, and nothing reported in it can be trusted. It is also unfair to reviewers to expect them to painstakingly validate a paper that has been generated by AI. Desk rejecting these papers helps preventing submission pools from being overwhelmed with slop submissions, and will make the review load of Program Committees more manageable. It also sends a signal to authors that misconduct has consequences. Simply having authors fix non-existing references when caught is not enough, as the AI generated claims that were linked to hallucinated references remain in the paper and can be erroneous.

Any AI slop papers that slips through the cracks and gets published poisons the respective research field, introducing potentially bogus results and claims. Hallucinated references are a type of AI slop, which can introduce citation distortion and make false claims about previous work spill into future research. We have seen this happening in the machine learning community, is the security community next? Is this already happening and we have not noticed?

It might come as a surprise to the reader who got to this point in this paper, but we are not luddites who reject AI in science and research. We believe that AI can greatly improve the human ability of finding information, conducting experiments, and producing results, but this must follow a *sustainable path*. Slop research results negatively affect AI models themselves in the long run, as they train on information that is out there. There was a time (until the release of ChatGPT) when AI models would be trained with almost only human-generated data. This time has long gone, and the lower the quality of AI-generated content that is littering the Internet, the lower will be the quality of future models.

We believe that human-AI symbiosis is key to future research progress, but this is based on trust. If AI models become known for producing bogus results, humans will stop trusting them, and miss on the multiplier that AI-assisted research can be. HALLUCINATOR helps the research community automate some of the checks needed to ensure the accuracy of research. It is written with the help of AI, but does not use AI in making its decisions. Humans are still the ultimate arbiters, and should verify any reference that is flagged by the tool.

While making it significantly easier to find hallucinated references, HALLUCINATOR only scratches the surface of the problem. Beyond incorrect bibliographic entries, a more insidious problem lies in references that do exist, but that do not back up the claims made in the paper. This is a longstanding problem in research, but AI has increased its magnitude. We are planning to look at this next – a careful use of AI can most likely confirm if a certain claim is present in a target paper or not.

An important question is how long will hallucinated references remain a problem for. Will next generation AI models get rid of them? We have been observing a pattern in the past years: while models get better and problems become less prevalent, they do not go away. For example, the issue that were facing LLMs when trying to detect software vulnerabilities in source code in 2023 [12] are still present, they just manifest themselves less often. We expect the same to happen with hallucinated citations. We tried to have two of the most advanced models available at the time of writing correct their own hallucinated references, OpenAI Codex 5.3 and Claude Opus 4.5. When asked to produce a survey paper on social network abuse as discussed in Section 1, Codex 5.3 produced 6 hallucinated references out of a total of 50 references, while Opus 4.5 produced 3 hallucinations out of 53 total citations. When asked to double check its output and fix any non-existing references, Opus 4.5 managed to fix all but one, leaving one hallucination in the final PDF. Codex 5.3, on the other hand, still produced a paper with 6 hallucinated citations. Interestingly, however, it fixed one of the hallucinated references in the original PDF, but introduced another hallucination in the process. This shows that the problem is far from being solved.

Another concern is that by releasing HALLUCINATOR we make it easier for people to cheat. An AI could use our tool to make sure that the references it produces are existing. This would, however, solve the problem of non-existing bibliographic references. While incorrect information might still be present in papers, having an existing reference to compare claims against would enable further analysis, opening the door to more robust citation and claim checking. In the end, if the goal is to eliminate hallucinated citations, we would prefer they are eliminated *before* they are uploaded to submission systems.

HALLUCINATOR is open source and publicly available, with pre-built binaries and Python wheels.[1] We welcome feedback by the research community and encourage Program Committees to use it in their own reviewing pipeline.

1. https://github.com/gianlucasb/hallucinator

Acknowledgments. We would like to thank Lujo Bauer, Cristian Cantoro, Emiliano De Cristofaro, and Ben Weinshel for their valuable feedback and help, including Github issues and pull requests.

References

[1] Y. Sakai, H. Kamigaito, and T. Watanabe, "Hallucitation matters: Revealing the impact of hallucinated references with 300 hallucinated papers in ACL conferences," *arXiv preprint arXiv:2601.18724*, 2026.

[2] N. Shmatko, A. Adam, and P. Esau, "GPTZero finds 100 new hallucinations in NeurIPS 2025 accepted papers," https://gptzero.me/news/neurips/, Jan 2026.

[3] E. Redmiles and B. Stock, "USENIX Security '26 Between-Cycle Transparency Report," Google Docs, Published online, 2026. [Online]. Available: https://docs.google.com/document/d/e/2PACX-1vSnHpeTFReelMZ124SzmgObXrHDN_B-UgxhOaE2RKfnfHn_qWuobI7kPg4TKBMhhZy_qxYtggCo7vjK/pub

[4] W. McCurdy, "Who Is Reviewing the Latest AI Research? It's AI, Apparently," Nov 2025. [Online]. Available: https://www.pcmag.com/news/who-is-reviewing-the-latest-ai-research-its-ai-apparently

[5] "Call for Papers: 33rd ACM Conference on Computer and Communications Security (CCS 2026)," ACM SIGSAC Website, 2025. [Online]. Available: https://www.sigsac.org/ccs/CCS2026/call-for/call-for-papers.html

[6] B. Emi, "Pangram Predicts 21% of ICLR Reviews are AI-Generated," Pangram Blog, Nov 2025. [Online]. Available: https://www.pangram.com/blog/pangram-predicts-21-of-iclr-reviews-are-ai-generated

[7] S. A. Greenberg, "How citation distortions create unfounded authority: analysis of a citation network," *BMJ*, vol. 339, 2009.

[8] G. Hine, J. Onaolapo, E. De Cristofaro, N. Kourtellis, I. Leontiadis, R. Samaras, G. Stringhini, and J. Blackburn, "Kek, cucks, and god emperor trump: A measurement study of 4chan's politically incorrect forum and its effects on the web," in *International AAAI Conference on Web and Social Media (ICWSM)*, 2017.

[9] C. Ling, U. Balcı, J. Blackburn, and G. Stringhini, "A first look at zoombombing," in *IEEE symposium on security and privacy (SP)*, 2021.

[10] M. H. Saeed, S. Ali, J. Blackburn, E. De Cristofaro, S. Zannettou, and G. Stringhini, "Trollmagnifier: Detecting state-sponsored troll accounts on reddit," in *IEEE symposium on security and privacy (SP)*, 2022.

[11] S. Zannettou, T. Caulfield, J. Blackburn, E. De Cristofaro, M. Sirivianos, G. Stringhini, and G. Suarez-Tangil, "On the origins of memes by means of fringe web communities," in *Internet Measurement Conference (IMC)*, 2018.

[12] S. Ullah, M. Han, S. Pujar, H. Pearce, A. Coskun, and G. Stringhini, "LLMs cannot reliably identify and reason about security vulnerabilities (yet?): A comprehensive evaluation, framework, and benchmarks," in *IEEE symposium on security and privacy (SP)*, 2024.