Boosting Big Brother: Attacking Search Engines with Encodings

Nicholas Boucher
University of Cambridge
nicholas.boucher@cl.cam.ac.uk

Luca Pajola
University of Padua
luca.pajola@phd.unipd.it

Ilia Shumailov
University of Oxford
ilia.shumailov@chch.ox.ac.uk

Ross Anderson
University of Cambridge & Edinburgh
ross.anderson@cl.cam.ac.uk

Mauro Conti
University of Padua
conti@math.unipd.it

ABSTRACT

Search engines are vulnerable to attacks against indexing and searching via text encoding manipulation. By imperceptibly perturbing text using uncommon encoded representations, adversaries can control results across search engines for specific search queries. We demonstrate that this attack is successful against two major commercial search engines - Google and Bing - and one open source search engine - Elasticsearch. We further demonstrate that this attack is successful against LLM chat search including Bing's GPT-4 chatbot and Google's Bard chatbot. We also present a variant of the attack targeting text summarization and plagiarism detection models, two ML tasks closely tied to search. We provide a set of defenses against these techniques and warn that adversaries can leverage these attacks to launch disinformation campaigns against unsuspecting users, motivating the need for search engine maintainers to patch deployed systems.

CCS CONCEPTS

• **Security and privacy** → *Systems security*; • **Information systems** → **Information retrieval**; • **Computing methodologies** → *Machine learning*;

KEYWORDS

search engines, attacks, text encodings, indexing, disinformation

ACM Reference Format:
Nicholas Boucher, Luca Pajola, Ilia Shumailov, Ross Anderson, and Mauro Conti. 2023. Boosting Big Brother: Attacking Search Engines with Encodings. In *The 26th International Symposium on Research in Attacks, Intrusions and Defenses (RAID '23), October 16–18, 2023, Hong Kong, China.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3607199.3607220

1 INTRODUCTION

Disinformation is a recurring threat to society exacerbated by the internet. While global internet connectivity democratizes knowledge by giving broad access to information, it also builds an easily scalable platform that can be used to provide purposefully manipulated content for promoting an adversarial goal. When adversaries coordinate the promotion of knowingly false information as truth via online platforms, we refer to such efforts as disinformation campaigns.

Search engines play a critical role in mitigating disinformation campaigns. Ethically, it would be dubious for search engines to identify and suppress disinformation; this quickly begins to take the form of censorship. However, users do expect search engines to produce representative results. From this assumption, it should be difficult for an adversary to severely manipulate search results without controlling the majority of relevant online content.

In this paper, we will show that this assumption is false. Adversaries can leverage atypical text encodings to manipulate search results in a targeted fashion.

Consider the dystopian world of George Orwell's *1984*: in the novel, the enemy and ally states in an ongoing war switch roles partway through the narrative. When this occurs, the relevant government rewrites wartime propaganda to state that the previous ally had always been the enemy [23]. In such a world, we would expect that search engines would surface at least some of the plethora of historical documents representing factual history rather than surfacing an exclusive subset of documents aligned with the advertised political narrative. Using the techniques presented in this paper, though, that expectation may not hold.

Search engines, like most text processing systems, understand text according to its binary encoding. For every instance of encoded text, there exist alternative representations that render the same glyphs encoded differently. Search engines that fail to correlate different representations of the same rendered text are subject to adversarial manipulation against both indexing and searching.

Using these techniques, an adversary can provide search terms that return targeted results across search engines and prevent content from being indexed as expected. Adversaries can leverage these attacks to boost disinformation campaigns by deceiving users into believing false claims are broadly supported by search results. These techniques could also be used to adversarially limit the discoverability of legal documents such as court disclosures or patents.

In this work, we study how malicious actors can manipulate the indexing and retrieval of web information through text encoding manipulations. We first focus on how search engines can be manipulated by analyzing the ability of attackers to perform: (1) *hiding*, *i.e.* the ability to hide adversarial content from benign query results; and (2) *surfacing*, *i.e.* the ability to yield adversarial results

RAID '23, October 16–18, 2023, Hong Kong, China
© 2023 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0765-0/23/10.
https://doi.org/10.1145/3607199.3607220

for perturbed queries. Using our techniques, an attacker may be able to publish content indexed by search engines that only appears in the search results of imperceptibly perturbed, adversarial queries. An example is shown in Figure 1, where malicious content appears only when users search using a poisoned query. Our experiments analyze how distinct search engines – commercial (e.g., Bing and Google) and open source (Elasticsearch) – behave under this novel threat. Furthermore, we assess how different machine learning systems that commonly support search engines can be affected by our proposed attack. In particular, we analyze the effect of our attacks on Bing's integration with GPT-4, Google's Bard model, text summarization models, and plagiarism detection models.

Our contributions can be summarized as follows:

- We introduce a novel attack against search engine indexing and querying through adversarial text encodings.
- We conduct experiments demonstrating that these attacks are successful against three major commercial and open source search engines.
- We demonstrate that these attacks also extend to chatbot search assistants, text summarization models, and plagiarism detection models, adding further attacks relevant to modern search engine ecosystems.

2 BACKGROUND

2.1 Encoding Text

Within computing systems, text must be represented in some binary fashion. To accomplish this, software designers first select a text specification such as Unicode [31][1]. This specification maps characters to numbers known as code points[2]. These code points are then represented in binary according to a chosen encoding, such as UTF-8.

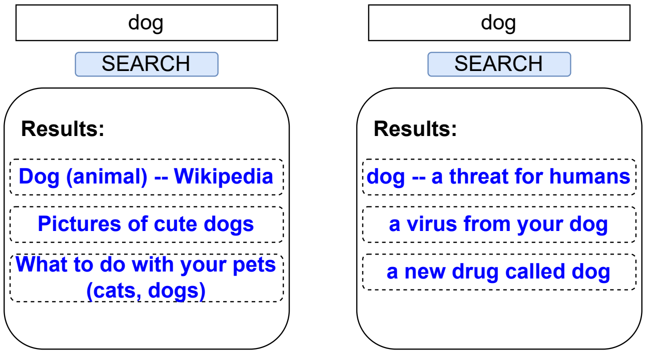
When text is rendered, binary values interpreted as code points according to the chosen encoding code points are resolved to characters via the text specification, then any control characters implementing special behavior are resolved, and finally characters are rendered as glyphs using the chosen font.

2.2 Imperceptible Perturbations

Rendered text is not injective with encoded representations; that is, identical rendered text can be represented by many different encoded values within the same text specification.

The following categories represent variations of imperceptible perturbations, also known as bad characters, which can be used to encode the same visual text with different values [6]:

(1) **Invisible Characters**
Invisible characters are characters that render to the absence of a glyph. They often exist for formatting purposes in specific contexts, and have no effect on text outside of those contexts. One example in Unicode is the Zero Width Space (ZWSP). Invisible characters can be injected arbitrarily to perturb text encodings without visual effect.



(a) Benign query.

(b) Malicious query.

Figure 1: An example of the attacker's goal. The figures show the search engine's top results for two "dog" queries, where one is benign (left) and the other adversarial (right). The queries appear identical, but the adversarial query is written with homoglyphs (`U+501` & `U+3BF`).

(2) **Homoglyphs**
Homoglyphs are characters that render to the same or nearly the same glyph. An example in Unicode is the Latin H and the Cyrillic H. Homoglyph characters can be substituted with each other to generate imperceptible perturbations.

(3) **Reorderings**
Reorderings exploit bidirectional text support by changing the display order of text while also changing the encoded order of text to offset the effect. An example in Unicode can be implemented using the Right-to-Left Override (RLO) control character. Since encoded values are manipulated without changing the rendered text, reorderings are a form of imperceptible perturbation.

(4) **Deletions**
Deletions exploit control characters designed to remove adjacent text by injecting arbitrary characters followed by an equivalent number of deletion control characters. An example in Unicode is the Delete (DEL) character. Deletions are powerful because they allow injecting characters into text without rendering them, although deletions are not imperceptible in all settings.

3 THE ATTACK

3.1 Threat Model

We propose a threat model in which an adversary seeks to insert web pages as highly ranked results for a specific search engine query executed by a victim user, where highly ranked is defined as appearing on the first default-sized page of results. The adversary does not have the ability to modify the search engine, nor does the adversary have knowledge of which search engine will be used by the victim. The adversary can create public websites that will be indexed by the search engine, but does not have the ability to

[1] Unicode is perhaps the most common modern text specification. According to w3techs.com/technologies/details/en-utf8, 97.9% of observed websites use the UTF-8 Unicode encoding.

[2] When written, Unicode code points are typically formatted as preceding with `U+`.

Table 1: Perturbation Techniques Used in Bad Search Wiki

Perturbation Name	Category	Description
base	Unperturbed	Text without perturbation to serve as a control.
zwsp	Invisible Character	Injects a Zero Width Space between all adjacent characters.
zwnj	Invisible Character	Injects a Zero Width Non-Joiner between all adjacent characters.
zwj	Invisible Character	Injects a Zero Width Joiner between all adjacent characters.
homo	Homoglyph	Substitutes each character with a randomly chosen homoglyph.
rlo	Reordering	Wraps text with a Right-to-Left Override and reverses logical order.
bksp	Deletion	Injects an `X` followed by a backspace character (`U+8`).
del	Deletion	Injects an `X` followed by a backspace character (`U+7F`).

promote those sites within the search engine index above other similar sites which they do not control.

A practical realization of this adversary is an actor conducting a disinformation campaign. This actor wishes for their search results to be prioritized over other results such that evidence conflicting the disinformation is crowded out by content under the actor's control.

3.2 Attack Technique

Within our threat model, an attacker can perform this attack by leveraging imperceptible perturbations.

Barring defenses, search engines understand both indexed content and search queries according to their encoded values. That is, visually identical text with and without imperceptible perturbations will be interpreted by search engines as distinct values. Adversaries can leverage this to plant poisoned content within the search engine index, and then surface that content to victim users who search using the imperceptibly perturbed string. This content will also be unlikely to show up in unperturbed queries, meaning that poisoned content is shown primarily to targeted users.

To illustrate this attack, consider the following example:

(1) Eve is running a disinformation campaign to deceive victims into believing that an unproven drug is an effective treatment for a certain ailment. Eve creates multiple fake websites attesting to the efficacy of the drug.
(2) Eve then modifies these sites such that each occurrence of the drug's name is imperceptibly perturbed with the same perturbation.
(3) Eve submits her sites for indexing in multiple major commercial search engines.
(4) Once Eve validates that the sites are indexed, she publicizes the drug on social media platforms using the perturbed version of the name.
(5) Alice, a victim, sees Eve's social media post and searches her favorite search engine to learn more about the drug. She copies the name of the drug from the social media post into the search bar rather than retyping the long name.
(6) Without realizing, Alice has searched for the imperceptibly perturbed version of the drug's name. The search engine returns Eve's fake websites as the top results, since they are the only indexed sites containing the search term imperceptibly perturbed in that manner.
(7) Alice is now deceived into believing that most internet results support Eve's disinformation claims.

By leveraging such techniques at scale, an adversary can significantly promote search engine results to support a broader disinformation campaign.

We note that in this setting, it is not necessary to deceive all potential victims. Some users may retype search queries thereby removing imperceptible perturbations, and yet others may go directly to trusted sources of information rather than search engines. So long as a subset of users leverage copy+paste or click-to-search functionality and review only "top" ranked sources, this attack will have victims.

4 EVALUATION

4.1 Methodology

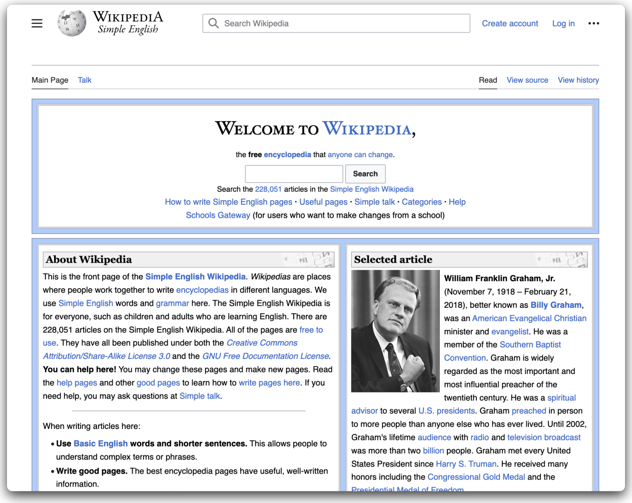
Our experiments test whether search engines can be affected by the presence of imperceptible perturbations both in indexing, *i.e.* parsing crawled content, and querying, *i.e.* performing searches. We define three distinct measures for evaluating the impact of imperceptible perturbations on search engines:

- *Disruption Potential*, measuring the SERP mismatch between benign and perturbed queries;
- *Hiding Potential*, measuring the ability of perturbed pages being discovered through benign queries;
- *Surfacing Potential*, measuring the ability of perturbed pages being discovered through perturbed queries.

Disruption is a broad measurement of whether a search engine is affected by imperceptible perturbations. Hiding is a more specific metric that determines whether indexed content can be withheld from search results for typical users of a search engine, while surfacing determines whether targeted content can be surfaced in search results for a targeted users. An fully vulnerable platforms has all of these properties. In the following paragraphs, we define these measures formally.

Disruption Potential. We analyze the discrepancy in Search Engine Results Pages (SERPs) between benign queries and their imperceptibly perturbed counterparts. In particular, SERP S from engine e is nothing more than a list of URLs u representing the highest ranked results for the given query x_i:

$$S_e(x_i) = [u_0, u_1, ..., u_n], \qquad (1)$$



Welcome to Wikipedia,

the free encyclopedia that anyone can change.

Search the 228,051 articles in the Simple English Wikipedia
How to write Simple English pages · Useful pages · Simple talk · Categories · Help
Schools Gateway (for users who want to make changes from a school)

About Wikipedia

This is the front page of the **Simple English Wikipedia**. *Wikipedias* are places where people work together to write encyclopedias in different languages. We use Simple English words and grammar here. The Simple English Wikipedia is for everyone, such as children and adults who are learning English. There are 228,051 articles on the Simple English Wikipedia. All of the pages are free to use. They have all been published under both the *Creative Commons Attribution/Share-Alike License 3.0* and the *GNU Free Documentation License*. **You can help here!** You may change these pages and make new pages. Read the help pages and other good pages to learn how to write pages here. If you need help, you may ask questions at Simple talk.

When writing articles here:

- **Use Basic English words and shorter sentences.** This allows people to understand complex terms or phrases.
- **Write good pages.** The best encyclopedia pages have useful, well-written information.

Selected article

William Franklin Graham, Jr. (November 7, 1918 – February 21, 2018), better known as **Billy Graham**, was an American Evangelical Christian minister and evangelist. He was a member of the Southern Baptist Convention. Graham is widely regarded as the most important and most influential preacher of the twentieth century. He was a spiritual advisor to several U.S. presidents. Graham preached in person to more people than anyone else who has ever lived. Until 2002, Graham's lifetime audience with radio and television broadcast was more than two billion people. Graham met every United States President since Harry S. Truman. He received many honors including the Congressional Gold Medal and the Presidential Medal of Freedom

Figure 2: Simple Wikipedia.

Therefore, we compare the SERP of x_i and x_i^{adv} as follows:

$$M_d(S_e, x_i, x_i^{adv}) = 1 - \frac{|S_e(x_i) \cap S_e(x_i^{adv})|}{|S_e(x_i)|}, \tag{2}$$

where $|\cdot|$ is the cardinality of the set. In other words, we attempt to measure how many correct results $S_e(x_i^{adv})$ contains. M_d is defined in $[0, 1]$, where 1 indicates that $S_e(x_i) \cap S_e(x_i^{adv}) = \emptyset$, *i.e.* there is a total mismatch between $S_e(x_i)$ and $S_e(x_i^{adv})$; conversely, when $M_d = 0$, the search engine S_e is not affected by the perturbation, *i.e.* $S_e(x_i) = S_e(x_i^{adv})$.

Hiding Potential. We analyze the ability of an attacker to hide content from a search engine's index using the hiding score. For this metric, we define u_i^{adv} as a URL containing imperceptibly perturbed content relevant to the unperturbed query x_i. We, therefore, define the hiding metric as follows:

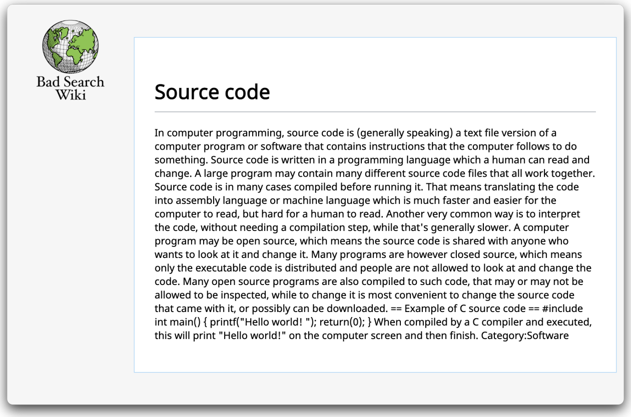
$$M_h(S_e, x_i, u_i^{adv}) = \begin{cases} 0 & \text{if } u_i^{adv} \in S_e(x_i), \\ 1 & \text{otherwise.} \end{cases} \tag{3}$$

Intuitively, this means that a high M_h score implies an attacker can prevent content from appearing in typical search results by adding imperceptible perturbations.

Surfacing Potential. Similarly, we analyze the ability of an attacker to surface a specific page in search engine results for a query of their choice using the surfacing score. For this metric, we define u_i^{adv} as a URL containing imperceptibly perturbed content relevant to the perturbed query x_i^{adv}. We therefore define the surfacing metric as follows:

$$M_s(S_e, x_i^{adv}, u_i^{adv}) = \begin{cases} 1 & \text{if } u_i^{adv} \in S_e(x_i^{adv}), \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

Intuitively, this means that a high M_s score implies an attacker can surface imperceptibly perturbed content with high confidence for a given perturbed query.



Source code

In computer programming, source code is (generally speaking) a text file version of a computer program or software that contains instructions that the computer follows to do something. Source code is written in a programming language which a human can read and change. A large program may contain many different source code files that all work together. Source code is in many cases compiled before running it. That means translating the code into assembly language or machine language which is much faster and easier for the computer to read, but hard for a human to read. Another very common way is to interpret the code, without needing a compilation step, while that's generally slower. A computer program may be open source, which means the source code is shared with anyone who wants to look at it and change it. Many programs are however closed source, which means only the executable code is distributed and people are not allowed to look at and change the code. Many open source programs are also compiled to such code, that may or may not be allowed to be inspected, while to change it is most convenient to change the source code that came with it, or possibly can be downloaded. == Example of C source code == #include int main() { printf("Hello world! "); return(0); } When compiled by a C compiler and executed, this will print "Hello world!" on the computer screen and then finish. Category:Software

Figure 3: Bad Search Wiki.

4.2 Experimental Setup

We evaluate our attack on three common search engines: Google, Bing, and Elasticsearch. Of these, Google and Bing, the two most common commercial search engines [30], are both black-box systems, while Elasticsearch is an open source search engine implementation.

Our evaluation analyzes search results on an imperceptibly perturbed version of Simple Wikipedia[3]. Pictured in Figure 2, Simple Wikipedia is an English-language instance of Wikipedia aimed at children and adults learning the language. At the time of writing, it contained 224,219 articles, making it more conducive for experimentation than the significantly larger primary Wikipedia instance. We leveraged eight perturbations for our experiments representing all four categories of imperceptible perturbations: invisible characters, homoglyphs, reorderings, and deletions. These perturbations are described in Table 1. We note that from our testing the deletion techniques produce visual artifacts in most web browsers; we include these techniques for robustness, but they would likely be avoided in practice. We also note that *base* is the name given to the control setting in all of our experiments for which no perturbations are applied.

Experiments against Elasticsearch involved running a local instance of the search engine and indexing the entirety of Simple Wikipedia for each perturbation technique.

Experiments against Google and Bing were more complicated, as we could not programmatically specify free-form data for indexing. To get around this, we deployed a mirror of Simple Wikipedia with added perturbations to a web server under our control[4]. We then requested indexing of the site with both Google and Bing, and leveraged each search engine's API for querying the index of our site only. It was rather challenging to get these sites into the index, requiring properly formatted sitemaps, robot files, crawl requests, and a friendly domain name before a sufficient portion of the site was indexed by either search engine. To accommodate indexing constraints, we randomly selected 100 articles for perturbations

[3] simple.wikipedia.org
[4] badsearch.soc.srcf.net

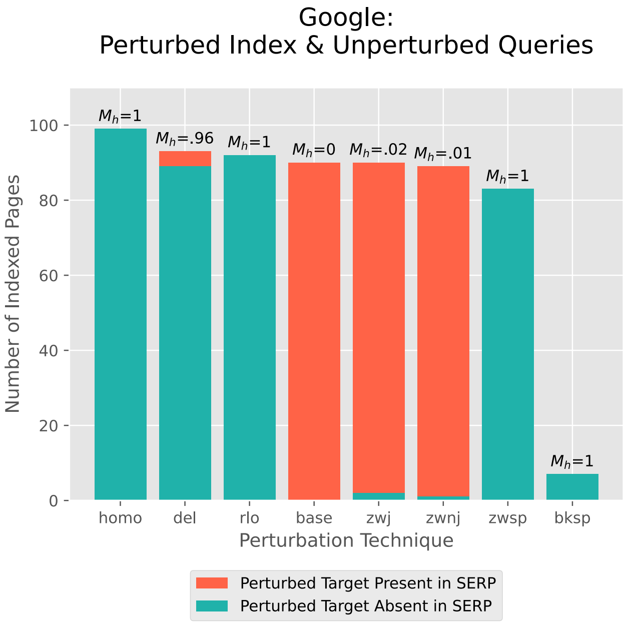


Figure 4: Google Hiding Experiment. The higher M_h, the stronger the attack. Green represents attack success.

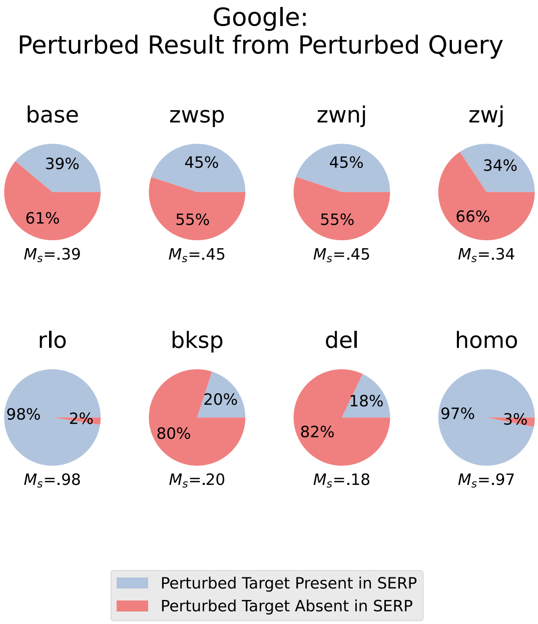


Figure 5: Google Surfacing Experiment. The higher M_s, the stronger the attack. Blue represents attack success.

across our eight techniques for experiments with Google and Bing rather than using the entirety of Simple Wikipedia.

Our experimental online wiki – dubbed the "Bad Search Wiki" – for which we depict a sample article in Figure 3 displays the title and article text taken from an export of Simple Wikipedia. We remove all formatting and embedded media from each article. Each article is repeated 8 times within the site, with each instance having a different perturbation applied. URLs do not contain any article-specific information not already in the page to prevent any effect on the indexing process.

Following our evaluations of Google, Bing, and Elasticsearch, we provide one additional set of experiments for Google and Bing that query the open internet rather than the Bad Search Wiki alone. This set of experiments aims to complement the Bad Search Wiki evaluations to show that the results presented throughout this paper also apply to web properties outside of our control.

The source code for our Bad Search Wiki and each experiment is available on GitHub[5].

4.3 Google

Google offers a Programmable Search Engine product[6] that allows automated querying of the Google search index. The API allows specifying individual URL patterns for inclusion in each search query, allowing us to select which perturbation technique pages to query with each search. Google also offers a Search Console[7] that makes is easy to detect if pages that you own are included in Google's index.

We conducted two different experiments against Google using the Bad Search Wiki which we will describe below.

The first experiment, which we call the *hiding experiment*, tests whether Google's search engine displays different behavior between the same query in perturbed and unperturbed form. To do this, we query the subset of Google's index that contains only articles perturbed using a single perturbation technique, such as zwsp, on our experimental site. For the search query, we use the unperturbed name of the target article. If Google returns the target article in its perturbed form, we can conclude that Google removes this form of perturbation during indexing.

The results of the hiding experiment are shown in Figure 4. We note that despite spending a full year attempting to get Google to index the entire Bad Search Wiki, there were some pages that were never included in the index. Missing pages are represented by shorter bars in this visualization. Likewise, pages that were indexed and returned the target article in its perturbed form in the first SERP, *i.e.* the top 10 URL hits, are represented in red; these represent pages that were not successfully "hidden" from the search engine through perturbations. Indexed target pages that were not returned in the first results page are represented in green, as these represent pages that were successfully "hidden". Mean M_h values are reported for each technique, where $S_e(x_i)$ is defined as the singleton set including only the target page perturbed using the selected technique.

Unsurprisingly, we see that unperturbed queries against an unperturbed index always include the target result. More surprising, however, is that ZWJ and ZWNJ results are also correctly included the majority of the time. This result suggests that Google is robust

[5] github.com/nickboucher/search-engine-attacks
[6] developers.google.com/custom-search
[7] search.google.com/search-console

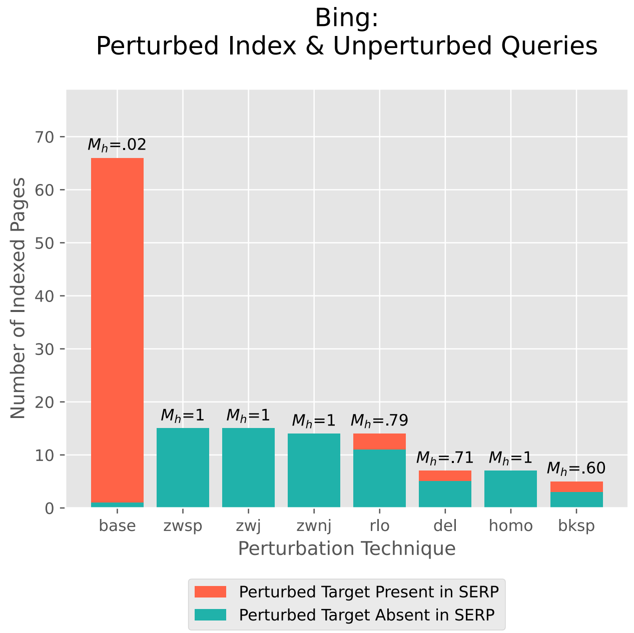


Figure 6: Bing Hiding Experiment. The higher M_h, the stronger the attack. Green represents attack success.

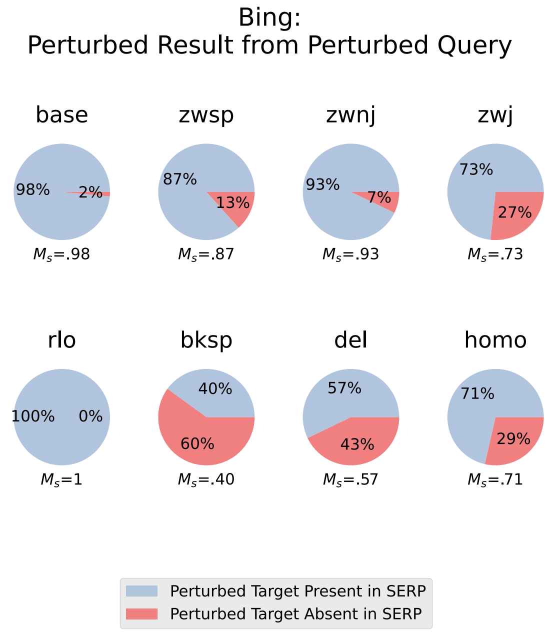


Figure 7: Bing Surfacing Experiment. The higher M_s, the stronger the attack. Blue represents attack success.

against ZWJ and ZWNJ, but not against ZWSP, homoglyphs, DELs, BKSPs, and RLOs.

The second experiment, which we call the *surfacing experiment*, queries the entire Bad Search Wiki site (all perturbations) with the search query being the perturbed form of an article title. If the same perturbed form of the article is present in the first page of query results, we can conclude that the perturbation technique is a good candidate for our attack when used against Google.

From the results, we can see that RLOs and homoglyphs are particularly good techniques for targeted content poisoning on Google. It is somewhat surprising that unperturbed (base) queries aren't 100% present, but we suspect that, following the results from the hiding experiment, Google views base, ZWJ, and ZWNJ as duplicative content, and randomly selects only one of these pages to show in the top search results.

From these results, we can conclude that reordering and homoglyph perturbations are highly effective attack techniques against Google. We can also conclude that Google has existing mitigations in place preventing ZWJ and ZWNJ-based perturbation attacks.

4.4 Bing

We conducted the same set of experiments against Bing that we conducted against Google and represent results equivalently. Bing also provides programmatic search engine querying via its Custom Search API product[8]. Similarly, web property owners can request and validate Bing indexing using Webmaster Tools[9].

The first experiment conducted against Bing was likewise the hiding experiment. As with Google, in this experiment we searched the index only including the target perturbation using the unperturbed article title as the search query. We note that compared to Google, we found it very difficult to index a large number of pages in Bing. The number of indexed pages is lower, despite the fact that we launched a second instance of the Bad Search Wiki site and combined the Bing results data for both.

The results shown in Figure 6 suggest that Bing views each perturbation technique distinctly; results and queries are not correctly associated between perturbed/unperturbed version, with the exception of the unperturbed control which was highly discoverable as expected. This data implies that there are not likely to be defenses built into Bing for any of the measured perturbation techniques.

The second experiment conducted against Bing likewise was the surfacing experiment. As with Google, in this experiment we searched the entire Bad Search Wiki index (all perturbations) using perturbed article titles.

The results shown in Figure 7 further suggest that Bing has little to no mitigations in place for perturbation attacks. The data are not as binary for each technique, but we suspect that the noise is higher in this experiment due to the smaller sample size per technique from indexing limitations. From this data, though, we can conclude that RLOs, ZWNJs, and ZWSPs are highly effective attack techniques against Bing. The remaining techniques are also likely to represent effective attacks with slightly lower attack success rate (ASR).

4.5 Elasticsearch

Since Elasticsearch is an open-source search engine, there is no need to deploy websites and request indexing to run experiments; rather, we can simply directly index our perturbed article titles and

[8] customsearch.ai
[9] bing.com/webmasters

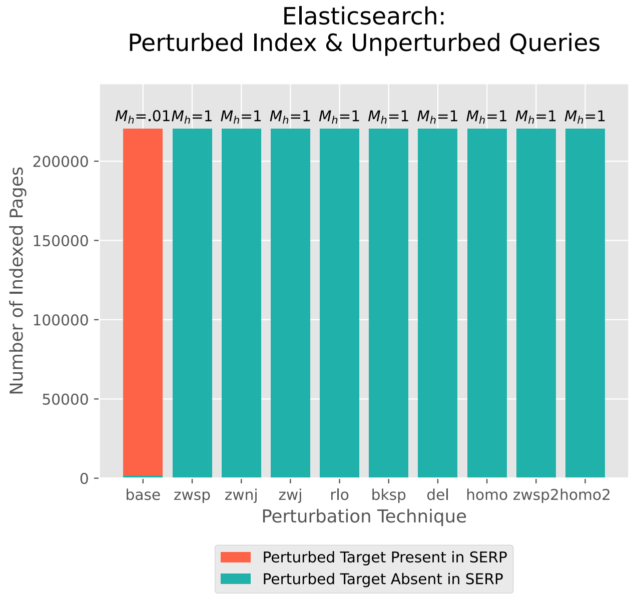


Figure 8: Elasticsearch Hiding Experiment. The higher M_h, the stronger the attack. Green represents attack success.

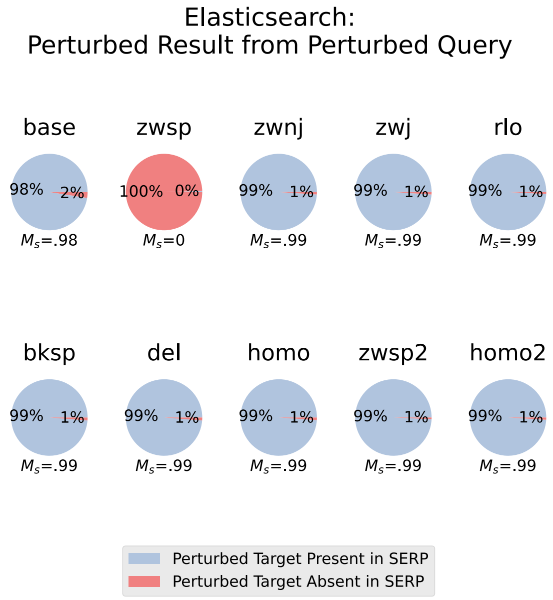


Figure 9: Elasticsearch Surfacing Experiment. The higher M_s, the stronger the attack. Blue represents attack success.

content. Indexing ability is thus not a limiting factor, and we therefore indexed all pages in Simple Wikipedia for each perturbation technique. We also added two additional perturbation techniques not seen in our previous experiments: zwsp2, for which article titles are alone perturbed with a random number of ZWSPs held constant for repeated words, and homo2 in which homoglyphs were manually selected to minimize visual perturbation artifacts rather than randomly selecting homoglyph substitutions.

Our experiments are conducted against Elasticsearch 8.5.3 run via Docker[10]. We performed queries via the Python Elasticsearch client.

Although the experimental setup was slightly different, we performed the same two evaluations as those conducted against Google and Bing and represent results equivalently.

The first experiment was therefore the hiding experiment, for which the results can be found in Figure 8. The results suggest that Elasticsearch interprets each perturbation technique as a distinct value from the unperturbed equivalent.

The second experiment was thus the surfacing experiment, and the results can be found in Figure 9. The results indicate that all techniques other than zwsp are highly successful in surfacing targeted, perturbed content using similarly perturbed search queries. This implies that ZWSPs are ignored in search queries, but all other perturbation techniques are treated distinctly from their unperturbed counterparts.

4.6 Open-Internet Measurement

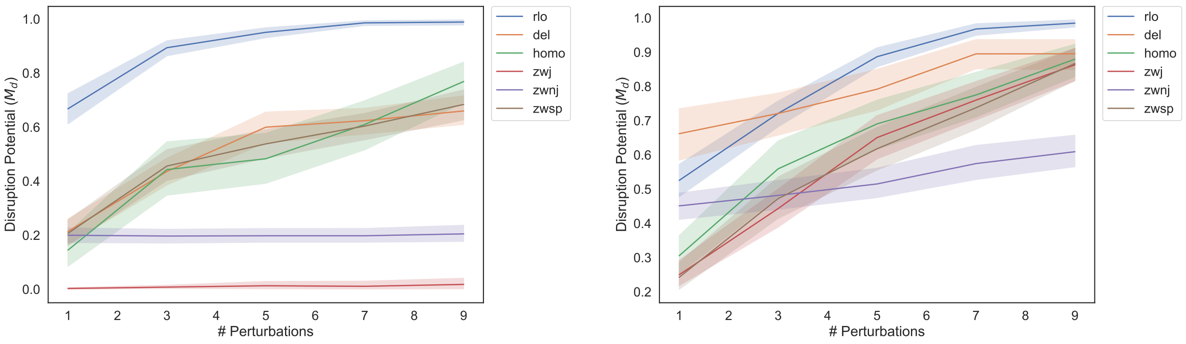
In a final set of experiments, we evaluated the performance of Google and Bing over the general internet with perturbed queries. Each engine was queried using the official API [12, 20] with queries formed from the question/answer dataset provided by the *boolq* version of the *super-glue* dataset [8, 33]. From this dataset, we randomly selected 100 questions and injected imperceptible perturbations in random positions. In our experiments, we vary the number of injected characters as follows: $\{1, 3, 5, 7, 9\}$. We then measure the *Disruption Potential* as defined in Equation 2. This set of experiments seeks only to validate the performance of search engines against imperceptible perturbations in the most general, open-internet setting, and does not seek to determine whether the URLs returned are related to malicious campaigns.

Figure 10 shows the open-internet experimental results for both Bing and Google. As expected, the results show that, in general, search engines are negatively affected by imperceptible perturbations, and as perturbations increase, performance drops significantly. Our injections' randomness can explain this phenomenon: when the perturbation is small – *e.g.* one character – its positioning

Table 2: Top 10 web pages occurring among perturbed URLs.

Website	Occurrences
en.wikipedia.org	1351
www.researchgate.net	1132
www.quora.com	449
www.imdb.com	366
www.ncbi.nlm.nih.gov	353
www.coursehero.com	293
www.youtube.com	292
screenrant.com	198
archive.org	196
www.nytimes.com	196

[10] hub.docker.com/_/elasticsearch



(a) Google.

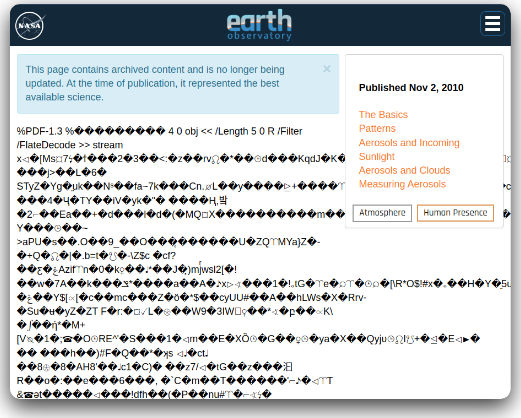
(b) Bing.

Figure 10: Effect of imperceptible perturbations on different search engines at the varying amount of perturbation. The higher M_d, the stronger the attack.

might not undermine the quality of the queries. However, when inserting many characters, it is more likely that these injections destroy the semantics of victim sentences.

These results also validate that Google is resistant to ZWJ and ZWNJ injections, with a stable performance at increasing perturbation. Comparatively, deletion, homoglyphs, and ZWSP characters have a moderately increasing negative effect. Bidi injections display an immediate and large effect, guaranteeing a significant drop in performance with a SERP similarity close to 0.1 from the injection of only three characters. Bing, on the other hand, appears vulnerable to all the classes of evaluated attacks. Similar to Google, the Bidi attack appears the most effective.

We also analyzed the discrepancies between SERP from perturbed and unperturbed queries. We aimed to answer the following question: *what is the nature of web pages contained in imperceptible perturbed SERPs but not in their unperturbed counterparts?* In this analysis, we combined this set of URLs returned by Google and Bing from imperceptible perturbed queries for collective analysis.



This page contains archived content and is no longer being updated. At the time of publication, it represented the best available science.

Published Nov 2, 2010

The Basics
Patterns
Aerosols and Incoming Sunlight
Aerosols and Clouds
Measuring Aerosols

Atmosphere Human Presence

```
%PDF-1.3 %���������� 4 0 obj << /Length 5 0 R /Filter
/FlateDecode >> stream
x◁�[Ms¤7ϟ�†���2�3��<:�z��rv♌�*��◷d���KqdJ�K�
���j>��L�6�
STyZ�Yg�ˎuk��Nˢ��fa~7k���Cn.⌀L��y����⊵+����♈
���4�Ч�TY��iV�yk�"� ����Ḩ,밝
�2⌐��Ea��+�d���l�d�(�MQ¤X�����������m��
Y���◷��~
>aPU�s��.O��9_��O���������U�ZQ♈MYa}Z�-
�+Q�♌�|�.b=t�♉�-\Z$c �cf?
��ƹ�ټAzif♈n�0�k♀��♩*��J�)mjwsl2[�!
��w�7A��k���ⵣ*����a��A�♪x▷⊀���1�!ₒtG�♈e�⌕♈�◷⌕�[\R*O$!#x�ₒ��H�Y�5u
�ټ��Y$[∝[�c��mc���Z�ð�*$��cyUU#��A��hLWs�X�Rrv-
�Su�ʉ�yZ�ZT F�r:�¤✓L�⊛��W9�3IW□♀��*⊀�ƥ��∝K\
�ʃ��ή*�M+
[V⊗�1�;☎�O◷RE^'�S���1�◁m��E�XÖ◷�G��♀◷�ya�X��Qyjʋ◷♌l♉+�⊴�E◁►�
�� ���h��)#F�Q��*�ʞs ◁♩�ct♩
��8⊛�8�AH8'��♩c1�C)� ��z7/◁�tG��z���汨
R��o�:��e���6���, �`C�m��T������'⌐♪�◁♈T
&☎ət������◁���!dfh��(�P��nu#♈�⌐⊀ϟ�
```

Figure 11: Example of a perturbed URL appearing in many distinct perturbed Bing queries.

We found 15,324 and 11,080 URLs for Bing and Google, respectively, for a total of 26,404 entries. We processed such URLs with *urlib*, a python library, and we extracted the following information: scheme (*e.g.* HTTPS), network location (*e.g.* google.com), URL path, URL parameters, and query. Among the 26404 URLs, we found that 507 use insecure communications (*i.e.* HTTP), while the other adopt HTTPS. A surprising outcome is the repetition of some network locations: in particular, we found 6,567 unique websites. Table 2 shows the top 10 popular websites. Such results appear in the 73% of Bing and 27% of Google results. Similarly, the top 10 webpages are not distributed uniformly among the various attack: 36% Bidi, 14% deletion, 5% homoglyph, 13% ZWJ, 13% ZWNJ, 19% ZWSP.

We found some perturbed URLs repeated across distinct queries, and analyzed who the most common perturbed URL publishers were. The top 5 perturbed URLs appear in Bing responses, and all of them belong to the "Bidi" attack. In order, docz.net[11] appears 121 times in 74 distinct queries, contains a URL fragmented by hyphens, and resolves to content similarly fragmented with whitespace. Researchgate.com[12] appears 118 times in 79 distinct queries; this web page exhibits similar fragmentation in its URL and title. Earthobservatory.nasa.com[13] appears 80 times in 59 distinct queries, and contains a broken PDF, as show in Figure 11. Next is Researchgate.com[14] again, appearing 75 times in 58 distinct queries; this webpage exhibits a fragmented URL, title, and content. Finally, text-id.123dok.com[15], appears 66 times among 51 distinct queries, and exhibits a fragmented URL and content.

5 SEARCH-ADJACENT MACHINE LEARNING

Beyond traditional search, there are a variety of machine learning powered systems that augment modern search engine products. In this section, we evaluate the performance of these search-adjacent machine learning tasks against imperceptible perturbations.

[11] doczz.net/doc/7707974/e-s-q-u-e-m-a-s-d-e-s-e-n-t-i-d-o—u-n-o-q-u-e-\%E2...
[12] www.researchgate.net/publication/301747842_R_e_l_a_t_i_o_n_s_h_i_p_b_e_t_w...
[13] earthobservatory.nasa.gov/features/Aerosols/what_are_aerosols_1999.pdf
[14] www.researchgate.net/publication/360221630_I_M_P_A_C_T_O_D_E_L_A_S...
[15] text-id.123dok.com/document/zpv7334z-h-s-i-l-g-n-el-a-i-r-e-t-a-m-g-n-i-t-i-r...

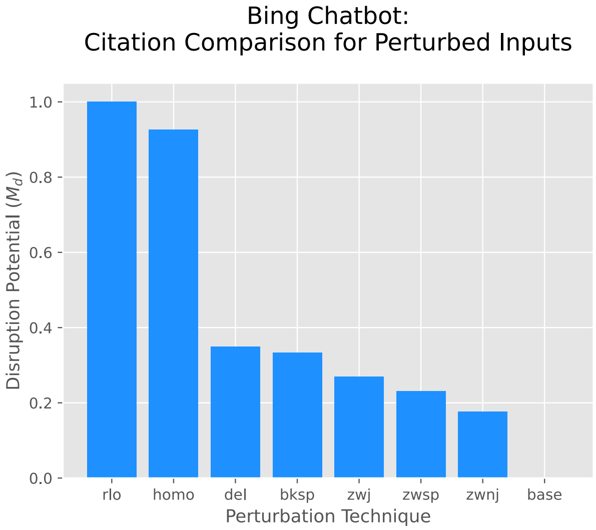


Figure 12: Comparison of Bing Chatbot web sources cited between perturbed and unperturbed inputs. The higher M_d, the stronger the attack.

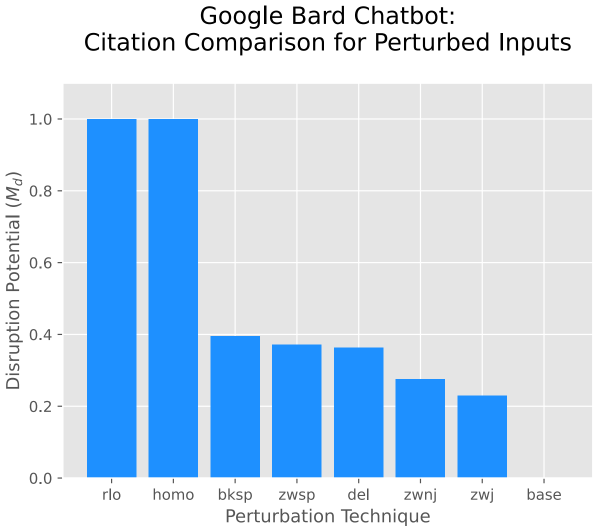


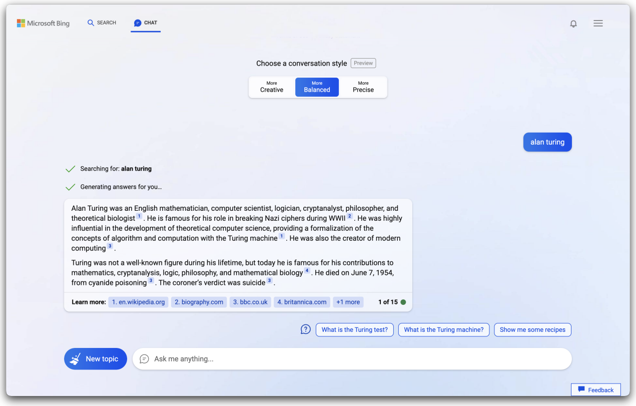
Figure 14: Comparison of Google Bard chatbot web sources cited between perturbed and unperturbed inputs. The higher M_d, the stronger the attack.

5.1 Chatbot Search

Recent strides in large language models (LLMs) have led some of the largest commercial search providers to claim that the future of search will be closely coupled with LLM-driven chatbots [18, 26]. The first global-scale product to market leveraging this technology for search was Microsoft’s update to Bing introducing a chatbot driven by OpenAI’s GPT-4 [17, 22], followed shortly thereafter by Google’s release of its competitor model Bard [14].

Given the relationship to traditional search, we were curious if both Bing’s GPT-4 and Google’s Bard chatbot were affected by imperceptible perturbations.

To test this, we provided the titles of the articles on our "Bad Search Wiki" as inputs to both model, repeating each inference for the different perturbation techniques listed in Table 1. To accomplish this, we wrote a script which directly queries each chatbot’s API and captures the results. Each input was provided in the context of a fresh chat session, such that previous inputs would not affect the models’ outputs.



Choose a conversation style

alan turing

Searching for: **alan turing**

Generating answers for you...

Alan Turing was an English mathematician, computer scientist, logician, cryptanalyst, philosopher, and theoretical biologist [1] . He is famous for his role in breaking Nazi ciphers during WWII [2] . He was highly influential in the development of theoretical computer science, providing a formalization of the concepts of algorithm and computation with the Turing machine [1] . He was also the creator of modern computing [3] .

Turing was not a well-known figure during his lifetime, but today he is famous for his contributions to mathematics, cryptanalysis, logic, philosophy, and mathematical biology [4] . He died on June 7, 1954, from cyanide poisoning [3] . The coroner’s verdict was suicide [3] .

Figure 13: Bing Chatbot UI.

Since the outputs of a search chatbot are different than the SERP outputs of search engines, we had to adjust the manner in which we interpret experimental results. Both the Bing and Bard conveniently provide a set of web sources with each query to serve as citations in responses generated. We compared the set of URLs returned as citations for perturbed inputs with those returned for unperturbed inputs using the disruption score M_d defined in Equation 2. We show this evaluation for Bing’s GPT-4 model in Figure 12. From these results, we observe that RLO and homo are the most effective perturbation techniques for disrupting chatbot response citations and are almost always successful. Each other technique, other than the control (base), also disrupted the results but had a success rate less than half that of the strongest techniques. We show the same evaluation for Google’s Bard in Figure 14 and observe that the results for each technique follow the same pattern as Bing but with a slightly higher attack success rate for nearly every perturbation technique.

In addition, we wanted to evaluate the non-URL text emitted by the chatbot in the presence of imperceptible perturbations. To accomplish this, we calculated the chrF score [27] for the perturbed model output with the unperturbed model output as the ground truth reference. In this metric, we compare only the text output and do not consider the web sources analyzed in the previous experiment. These results for Bing can be seen in Figure 15. From these results, we see again that the Bing GPT-4 model was most affected by rlo and homo perturbation techniques, with a notable but less powerful affect on each other perturbation technique. The same evaluation for Google’s Bard is show in Figure 16, and once again the trends are nearly identical to Bing’s GPT-4 but with a higher attack success rate.

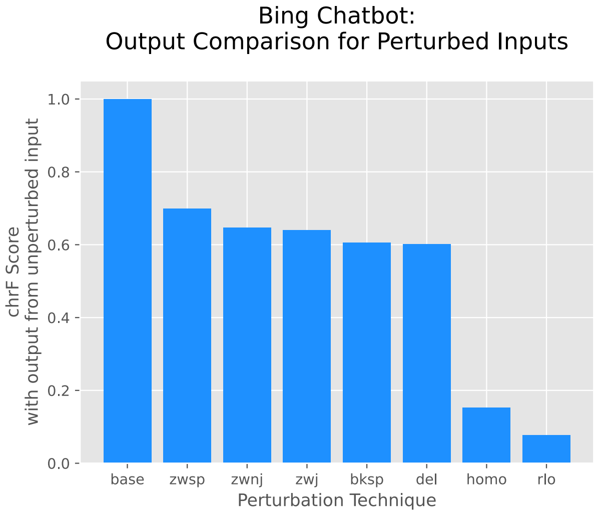


Figure 15: Comparison of Bing Chatbot text outputs between perturbed and unperturbed inputs. The lower chrF Score, the stronger the attack.

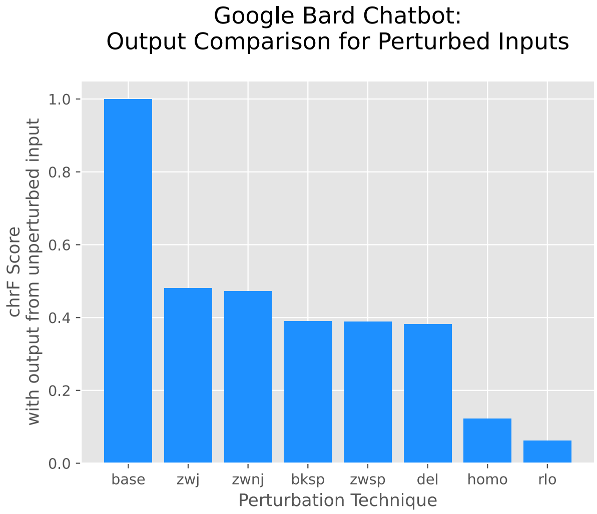


Figure 16: Comparison of Google Bard chatbot text outputs between perturbed and unperturbed inputs. The lower chrF Score, the stronger the attack.

From these results, we can conclude that both Bing's GPT-4 chatbot and Google's Bard chatbot is highly vulnerable to manipulation via bidirectional control characters and homoglyphs, and at least somewhat vulnerable to each other perturbation technique examined.

5.2 Summarization

Automatic Text Summarization (ATS) is the process of reducing the length of a source of text without losing salient aspects or essential information. ATS are critical tools, especially in the Web context, where users seeking information might need to deal with many distinct sources containing enormous quantities of data [11]. Following the outcomes shown so far, we now demonstrate that imperceptible perturbations can affect ATS. The reasons why the attack might succeed are similar to the conjecture described in the ZeW-attack [24]: the introduction of imperceptible perturbations affects the indexing stage of NLP pipelines; therefore, since such characters are likely not present in a regular vocabulary, obfuscated sentences will be translated into a sequence of meaningless representations.

In this work, we proposed a novel generative adversarial procedure for ATS. To the best of our knowledge, no prior works discuss adversarial attacks in ATS. Therefore, we now formalize attackers' goals and capabilities. Conceptually, attacking an ATS is similar to the more traditional *evasion attack* [3], where adversarial examples are carefully crafted to produce a misclassification in the victims model. Similarly, in adversarial ATS, an attacker might aim to produce the following types of attack:

- *Targeted attack*, where the summarized text reflects only targeted properties. For instance, an attacker might aim to produce a summarization that avoids specifc concepts or, similarly, to a summarization that avoids targeted emotions (*e.g.* the output does not contain negative sentences);
- *Untargeted attack*, where the unique purpose is to maximize the output degradation. This scenario can be seen as a *Denial-of-Service*.

In this work, we focus on the latter scenario, *i.e.* untargeted attack, and we design a novel adversarial procedure by considering the following aspects:

1. *Passive attack*: as perturbed samples are likely to appear in webpages, attackers do not know what summarizers will be applied, nor when the page will be summarized.
2. *Gradient-free optimization*: attackers might prefer educated guessing or heuristics to produce perturbed samples since gradient-based attack might require knowledge of victims' model [2].
3. *Imperceptible*: imperceptible perturbations maximize the stealthiness of the attack since they are by definition imperceptible [6, 24].

5.2.1 *The Mikado Algorithm.*

We now describe the proposed attack we named "Mikado Algorithm". Following the observations made in Section 5.2 and what is described by Pajola and Conti [24], for the untargeted scenario, the maximum expectation of attack is found when the adversarial samples are entirely obfuscated. This means that, for Bidi, deletion, ZWJ, ZWNJ, and ZWSP, perturbed characters are inserted between each character of the given sentence; similarly, in a homoglyph attack, each sentence's character is replaced with its homoglyph counterpart. We refer to O as the obfuscation function that produces the strongest adversarial sample for a given input x and technique t, with $t \in$ [bidi, deletion, homoglyph, ZWJ, ZWNJ, ZWSP]. Therefore,

$$x^{adv} = O(x, t). \tag{5}$$

The attacker might evaluate the quality of the produced adversarial sample by leveraging an external ATS $\mathcal{A}$, and an evaluation

Algorithm 1 Mikado generation process

```
Require: $x$                                   ▷ Original sample
Require: $t$                                   ▷ Type of obfuscation
Require: $p$                                   ▷ Size of population
Require: $n$                                   ▷ Step of sanitization
Require: $b$                                   ▷ Budget
  $x^{adv} = \mathcal{O}(x, t)$
  $m = \mathcal{M}\Big(\mathcal{A}(x), \mathcal{A}(x^{adv})\Big)$
  while $m \leq b$ do
      $\mathbf{x}^{cand} = \mathcal{S}(x^{adv}, p, n)$
      $i = \min_i \mathcal{M}\Big(\mathcal{A}(x), \mathcal{A}(x_i^{cand})\Big)$
      $m = \mathcal{M}\Big(\mathcal{A}(x), \mathcal{A}(x_i^{cand})\Big)$
      if $m \leq b$ then
          $x^{adv} = x_i^{cand}$
      end if
  end while
```

metric $\mathcal{M}$. For instance, $\mathcal{A}$ can be an online free tool or a python pre-trained library. $\mathcal{M}$ needs to measure the similarity between the summarization between x and x_adv; in our implementation, we use the BLEU score:

$$m = \mathcal{M}\Big(\mathcal{A}(x), \mathcal{A}(x^{adv})\Big), \tag{6}$$

with the similarity $m \in [0, 1]$, where 0 indicates that the two summarized text are completely different, and 1 are identical. Note that, for a fair comparison between $\mathcal{A}(x)$ and $\mathcal{A}(x^{adv})$, the adversarially summarized text needs to be sanitized of imperceptible perturbation. This is necessary since, otherwise, two visually identical words (one with imperceptible perturbations and the other without) will be treated as two different words from the BLEU score.

The produced x^{adv} might contain unnecessary imperceptible perturbations that, if removed, will not alter the overall quality of the attack. We recall that while the homoglyph attack produces a sentence with the same amount of characters, this ratio is ×2 when considering deletion, ZWNJ, ZWJ, and ZWSP; the Bidi algorithm, instead, produces ×5 characters. To reduce the number of perturbed characters without impacting the attack's quality, we designed a genetic algorithm, given their success in NLP adversarial generations [1, 6]. The procedure is based on a sanitization function $\mathcal{S}$ that, given an adversarial sample x^{adv}, generates p versions of the adversarial samples by randomly removing n perturbed characters. We call this list of candidates $\mathbf{x}^{cand}$:

$$\mathbf{x}^{cand} = \mathcal{S}(x^{adv}, p, n). \tag{7}$$

The best candidate in the population is found as follows:

$$\min_i \mathcal{M}\Big(\mathcal{A}(x), \mathcal{A}(x_i^{cand})\Big), i \in [0, p-1]. \tag{8}$$

Note that it is likely that the best candidates x_i^{cand} produce an attack weaker than x^{adv}. Therefore, we introduce a budget parameter $b \in [0, 1]$ that indicates the maximum similarity we are willing to accept among the original and adversarial summarization:

$$x^{adv} = \begin{cases} x_i^{cand}, & \text{if } \mathcal{M}\Big(\mathcal{A}(x), \mathcal{A}(x_i^{cand})\Big) \leq b \\ x^{adv}, & \text{otherwise} \end{cases}. \tag{9}$$

If x^{adv} is updated, we repeat the procedure by generating a new population of sanitized candidates; on the opposite, we return x^{adv}. Algorithm 1 shows Mikado's pseudo-algorithm. Attackers not willing to minimize the amount of perturbation, or simply without access to any ATS, can skip the optimization procedure, and use the maximum perturbed adversarial sample.

5.2.2 Summarization Attack Evaluation. In our experiment, we aim not only to prove the ferocity of the Mikado algorithm toward a target model $\mathcal{A}$ used during the optimization process in a black-box manner but to show further that the generated adversarial samples can *transfer* among different ATS. We recall that with transferability, we refer to the ability of an attack to produce adversarial samples effective in any unknown model, and not only the one used during their generation [25]. In our experiment, we used the "cnn_dailymail" dataset (v3.0.0) [13, 28], consisting in a corpus of news articles written by journalists at CNN and the Daily Mail. For the attack, a random subsample of 500 instances is utilized. We considered three popular ATS trained on the "cnn_dailymail" dataset, and available on *Hugging Face*[16]:

- `facebook/bart-large-cnn`;
- `sshleifer/distilbart-cnn-12-6`;
- `sshleifer/distilbart-xsum-12-6`.

We use the former model in the Mikado optimization process, while the remaining models evaluate the attack transferability. We set $p = 20$, $n = 250$, and $b = 0.1$. In our implementation, we do not test the "Bidi attack" since we could not guarantee a proper output sanitization; we recall that the Bidi algorithm inserts, for each transformation, multiple sequences of characters. The order of such characters might not be respected in the summarized version.

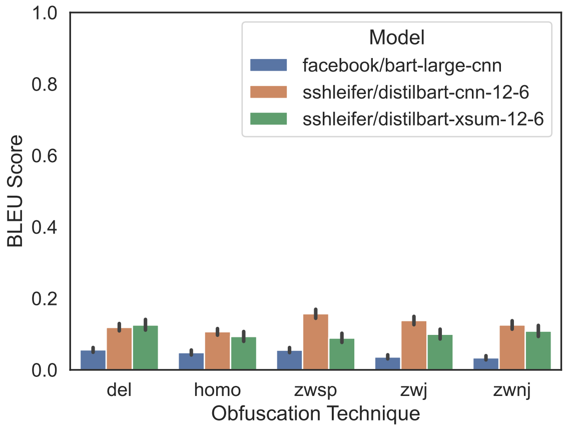


Figure 17: Attack performance varying of ATS and obfuscation technique according to Equation 6. The lower the score, the stronger the attack.

[16] https://huggingface.co/

Table 3: Examples of summarization failures.

True Summarization	Adversarial Summarization	Score
CNN confirms that Michelle MacLaren is leaving the upcoming "Wonder Woman" movie. The movie, starring Gal Gadot in the title role of the Amazon princess, is still set for release on June 23, 2017. It's the first theatrical movie centering around the most popular female superhero.	Wanted to shoot footage of golden lassos and invisible jets. MacLaren was announced as the director of the movie in November. Warner Bros. and Michelle MacLaren have decided not to move forward with the movie.	0.01
Daniel Boykin, 33, entered the woman's home multiple times, where he took videos, photos and other data. Police found more than 90 videos and 1,500 photos of the victim on Boykin's phone and computer. The victim filed a complaint after seeing images of herself on his phone last year.	Daniel Boykin, 33, videotaped a female co-worker in a TSA-on-ly restroom. The woman filed a complaint after seeing images of herself on his phone. Boykin worked in an administrative capacity and did n't engage in public security screening.	0.21
The Americans" is one of the best series on TV. "Fresh Off the Boat" is the first sitcom with an Asian-American cast since the 1990s. "Scorpion," about a ragtag band of geniuses sent on secret missions, got a lot of hype.	The Americans is one of the best series on TV, critics say. The finale is titled March 8, 1983, when President Reagan called the Soviet Union. The first History Channel scripted series wraps up on Thursday.	0.15

The result shown in Figure 17 confirm our intuition: ATS are affected by imperceptible perturbations. This is true not only on the model used during the Mikado optimization but also while transferring to unknown models. The attack is effective, with, on average, a score equal to 0.1 for the Facebook model, corresponding to the budget *b* limit we imposed. Similarly, the efficacy is preserved in the unknown models, with all the cases below 0.2.

Table 3 shows examples of adversarial summarized text. The reader might notice that when the score is close to zero, the adversarial summarization appears utterly different from the correct version. When the score is higher (*e.g.* 0.2), adversarial summarizations appear to discuss the same topic as their correct versions; however, they fail to preserve the target insights.

5.3 Plagiarism Detection

Another family of tools that can be affected by imperceptible perturbations is plagiarism detection. Plagiarism is a fraudulent appropriation of another party's intellectual property. The goal of such a tool is, given a source of text, to spot potential third party sources that are exactly or nearly identical. Plagiarism checkers are essential in many fields to help verify the ownership of new content. Common use cases include the journalism and the scientific communities. In this setting, we consider a plagiarism checker as a classifier that returns 1 if the input content is plagiarized, and 0 if plagiarism is not detected. The attacker's goal is to evade the detection of such classifiers when using plagiarized content.

We performed experiments against Grammarly[17], a popular tool for grammatical corrections widely used in both academia and industry with integration into many web applications (*e.g.* Overleaf). Among the tools offered within Grammarly is a plagiarism checker. Since the tool does not provide APIs to interact directly with the service, we conducted the experiments manually through a web interface. For this experiment, we used 50 samples of Kaggle's Quote dataset[18]. For each sample, we injected the maximum amount of perturbation for all the attacks we describe in Table 1, except deletion. The deletion attack cannot be performed since deletion control characters are actioned by the web browser before ingestion into the Grammarly interface. We measure the attack using *attack success rate*, which we define in this setting as the percentage of misclassifications.

The results of the experiment demonstrate the ferocity of imperceptible perturbations. In particular, all the analyzed attacks appear to be effective, producing a perfect attack success rate. In other words, Grammarly did not detect any of our imperceptibly perturbed plagiarized content as plagiarism.

6 DISCUSSION

6.1 Ethics

Whenever novel attacks are proposed, ethics must be closely considered. We found that two commercial search engines were vulnerable to our proposed attacks, and therefore notified the maintainers of these systems prior to public release of this paper. We also paid full price for all commercial search engine APIs used to conduct the experiments described within this paper.

6.2 Defense

Search engine usage is typically an interactive operation. This suggests a simple mitigation that can be easily retrofitted onto existing search engines: search engines should provide some visual indication when imperceptible perturbations are detected. This could take the form of a warning message, highlighting suspected-perturbed characters in the search box, or anything else that brings visual attention to encoding level perturbations. Attack detection is straightforward: a RegEx that detect invisible characters, bidirectional control characters, deletion control characters, and selected homoglyphs would suffice.

Search engine maintainers looking for a more robust solution can build a text sanitization phase into their search pipelines. This would need to take the form of removing invisible characters, resolving control characters, and replacing homoglyphs with common characters. Homoglyph replacement is not as straightforward as the other tasks, but there are ways in which a collection of explicitly defined homoglyphs can be defined for mapping to common characters during sanitization. A precedent is given by the Punycode standard for mapping strings containing Unicode characters, such as internationalized domain names, into the subset of ASCII favored by DNS [10].

Barring either of these defenses, each of the search engines examined, and likely future search engine implementations, will be vulnerable to this attack and provide an avenue that could support disinformation campaigns.

[17] grammarly.com

[18] kaggle.com/datasets/akmittal/quotes-dataset

**Table 4: Summary of all experimental results.
Larger numbers / greener cells are more successful attacks.**

Target	**Metric**	**Metric Change** (Relative to No Perturbation)						
		zwsp	**zwnj**	**zwj**	**homo**	**rlo**	**bksp**	**del**
Google Search	M_h	100%	1%	2%	100%	100%	100%	96%
Google Search	M_s	6%	6%	-5%	58%	59%	-19%	-21%
Google Search	M_d ($b = 9$)	68%	21%	2%	77%	99%	n/a	66%
Bing Search	M_h	98%	98%	98%	98%	77%	58%	69%
Bing Search	M_s	-11%	-5%	-25%	-27%	2%	-58%	-41%
Bing Search	M_d ($b = 9$)	87%	61%	86%	88%	98%	n/a	90%
Elasticsearch	M_h	99%	99%	99%	99%	99%	99%	99%
Elasticsearch	M_s	-98%	1%	1%	1%	1%	1%	1%
Bing Chatbot (GPT-4)	M_d	23%	18%	27%	93%	100%	33%	35%
Bing Chatbot (GPT-4)	-chrF	30%	35%	36%	85%	92%	39%	40%
Google Bard	M_d	37%	28%	23%	100%	100%	40%	36%
Google Bard	-chrF	61%	53%	52%	88%	94%	61%	62%
ATS: bart-large-cnn	1-BLEU	94%	97%	96%	95%	n/a	n/a	94%
ATS: distilbart-cnn-12-6	1-BLEU	84%	87%	86%	89%	n/a	n/a	88%
ATS: distilbart-xsum-12-6	1-BLEU	91%	89%	90%	91%	n/a	n/a	87%

7 RELATED WORK

7.1 Encoding Attacks

Text encoding manipulation has been used across many different domains to attack a myriad of systems. One such example is NLP systems, which have previously been targeted with adversarial examples formed using Unicode perturbations [6, 24]. Similar techniques were used to craft the Trojan Source attack, an attack which places vulnerabilities in source code that are not rendered to human code reviewers [5]. Control characters have also been used to disguise the filenames and filepaths of malware [7, 16, 19]. In other settings, Unicode has been leveraged to create steganographic communication channels for botnets [9]. Lastly, homoglyphs [29] and invisible characters [21] have long been used to conduct phishing campaigns.

7.2 Disinformation Campaigns

The internet enables individuals to connect globally by powering platforms such as social networks (*e.g.* Facebook), e-commerce businesses (*e.g.* Amazon), and online forums (*e.g.* Reddit). Despite the many benefits of a globally connected internet, it also offers many opportunities to malicious actors; in particular, it allows the rapid dissemination of potentially adversarial information. Conspiracy theories, rumors, and other forms of disinformation have the potential to affect public opinion [32], which poses a particularly potent threat to democratic societies. For instance, Bessi and Ferrara [4] observed that the presence of bots on social network platforms in the US 2016 political elections manipulated the political discussion. Later, during the COVID-19 pandemic, conspiracy campaigns were widely spread via social networks [15].

8 CONCLUSION

In this paper, we presented a novel attack on search engines that leverages imperceptible perturbations in text encodings to manipulate search engine results. We find that our attacks work on real-world, deployed commercial search engines as summarized in Table 4. We also found that this attack successfully extends to search-adjacent machine learning models, including search chatbots, text summarization, and plagiarism detection. When exploited, this vulnerability can be used to power disinformation campaigns. Simple defenses exist in the form of visual alerts and input sanitization, and it is necessary for search engine maintainers to adopt such defenses to mitigate this risk.

REFERENCES

[1] Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating Natural Language Adversarial Examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium, 2890–2896. https://doi.org/10.18653/v1/D18-1316

[2] Giovanni Apruzzese, Hyrum S. Anderson, Savino Dambra, David Freeman, Fabio Pierazzi, and Kevin Roundy. 2023. "Real Attackers Don't Compute Gradients": Bridging the Gap Between Adversarial ML Research and Practice. In *2023 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*. IEEE, Raleigh, NC, 339–364. https://doi.org/10.1109/SaTML54575.2023.00031

[3] Marco Barreno, Blaine Nelson, Russell Sears, Anthony D. Joseph, and J. D. Tygar. 2006. Can Machine Learning Be Secure?. In *Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security* (Taipei, Taiwan) *(ASIACCS '06)*. Association for Computing Machinery, New York, NY, USA, 16–25. https://doi.org/10.1145/1128817.1128824

[4] Alessandro Bessi and Emilio Ferrara. 2016. Social bots distort the 2016 US Presidential election online discussion. *First monday* 21, 11-7 (2016).

[5] Nicholas Boucher and Ross Anderson. 2023. Trojan Source: Invisible Vulnerabilities. In *32nd USENIX Security Symposium (USENIX Security 23)*. USENIX Association, Anaheim, CA. https://arxiv.org/abs/2111.00169

[6] Nicholas Boucher, Ilia Shumailov, Ross Anderson, and Nicolas Papernot. 2022. Bad Characters: Imperceptible NLP Attacks. In *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, San Francisco, CA, 1987–2004. https://doi.org/10.1109/SP46214.2022.9833641

[7] Brian Krebs. 2011. 'Right-to-Left Override' Aids Email Attacks. https://krebsonsecurity.com/2011/09/right-to-left-override-aids-email-attacks/

[8] Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the Surprising Difficulty of Natural Yes/No Questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota, 2924–2936. https://doi.org/10.18653/v1/N19-1300

[9] Alberto Compagno, Mauro Conti, Daniele Lain, Giulio Lovisotto, and Luigi Vincenzo Mancini. 2015. Boten ELISA: A novel approach for botnet C&C in Online Social Networks. In *2015 IEEE Conference on Communications and Network Security (CNS)*. IEEE, Florence, Italy, 74–82. https://doi.org/10.1109/CNS.2015.7346813

[10] A. Costello. 2003. *Punycode: A Bootstring encoding of Unicode for Internationalized Domain Names in Applications (IDNA)*. RFC 3492. Internet Requests for Comments. https://www.rfc-editor.org/rfc/rfc3492

[11] Wafaa S El-Kassas, Cherif R Salama, Ahmed A Rafea, and Hoda K Mohamed. 2021. Automatic text summarization: A comprehensive survey. *Expert Systems with Applications* 165 (2021), 113679.

[12] Google. 2023. Custom search JSON API. https://developers.google.com/custom-search/v1/overview

[13] Karl Moritz Hermann, Tomáš Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching Machines to Read and Comprehend. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1* (Montreal, Canada) *(NIPS'15)*. MIT Press, Cambridge, MA, USA, 1693–1701.

[14] Sissie Hsiao and Eli Collins. 2023. Try Bard and share your feedback. https://blog.google/technology/ai/try-bard

[15] Md Saiful Islam, Abu-Hena Mostofa Kamal, Alamgir Kabir, Dorothy L Southern, Sazzad Hossain Khan, SM Murshid Hasan, Tonmoy Sarkar, Shayla Sharmin, Shiuli Das, Tuhin Roy, et al. 2021. COVID-19 vaccine rumors and conspiracy theories: The need for cognitive inoculation against misinformation to improve vaccine adherence. *PloS one* 16, 5 (2021), e0251605.

[16] Jakob Lell. 2014. [Hacking-Contest] Invisible configuration file backdooring with Unicode homoglyphs. https://www.jakoblell.com/blog/2014/05/07/hacking-contest-invisible-configuration-file-backdooring-with-unicode-homoglyphs/

[17] Yusuf Mehdi. 2023. Confirmed: the new Bing runs on OpenAI's GPT-4. https://blogs.bing.com/search/march_2023/Confirmed-the-new-Bing-runs-on-OpenAI%E2%80%99s-GPT-4

[18] Yusuf Mehdi. 2023. Reinventing search with a new AI-powered Microsoft Bing and Edge, your copilot for the web. https://blogs.microsoft.com/blog/2023/02/07/reinventing-search-with-a-new-ai-powered-microsoft-bing-and-edge-your-copilot-for-the-web

[19] Microsoft. 2017. Win32/Sirefef. https://www.microsoft.com/en-us/wdsi/threats/malware-encyclopedia-description?Name=Win32/Sirefef

[20] Microsoft. 2023. Web search API: Microsoft bing. https://www.microsoft.com/en-us/bing/apis/bing-web-search-api

[21] Yoav Nathaniel. 2019. Z-wasp vulnerability used to Phish Office 365 and ATP. https://www.avanan.com/blog/zwasp-microsoft-office-365-phishing-vulnerability

[22] OpenAI. 2023. GPT-4 Technical Report. arXiv:2303.08774 [cs.CL]

[23] George Orwell. 1949. *1984*. Secker & Warburg, London.

[24] L. Pajola and M. Conti. 2021. Fall of Giants: How popular text-based MLaaS fall against a simple evasion attack. In *2021 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE Computer Society, Los Alamitos, CA, USA, 198–211. https://doi.org/10.1109/EuroSP51992.2021.00023

[25] Nicolas Papernot, Patrick McDaniel, and Ian Goodfellow. 2016. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples.

[26] Sundar Pichai. 2023. An important next step on our AI journey. https://blog.google/technology/ai/bard-google-ai-search-updates

[27] Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*. Association for Computational Linguistics, Lisbon, Portugal, 392–395. https://doi.org/10.18653/v1/W15-3049

[28] Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get To The Point: Summarization with Pointer-Generator Networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Vancouver, Canada, 1073–1083.

[29] Geoffrey Simpson, Tyler Moore, and Richard Clayton. 2020. Ten years of attacks on companies using visual impersonation of domain names. In *2020 APWG Symposium on Electronic Crime Research (eCrime)*. IEEE, Boston, MA, 1–12. https://doi.org/10.1109/eCrime51433.2020.9493251

[30] StatCounter. 2022. Global Search Engine Market Share 2022. https://www.statista.com/statistics/216573/worldwide-market-share-of-search-engines/

[31] The Unicode Consortium. 2022. The Unicode Standard, Version 15.0. https://www.unicode.org/versions/Unicode15.0.0

[32] Michela Del Vicario, Alessandro Bessi, Fabiana Zollo, Fabio Petroni, Antonio Scala, Guido Caldarelli, H. Eugene Stanley, and Walter Quattrociocchi. 2016. The spreading of misinformation online. *Proceedings of the National Academy of Sciences* 113, 3 (2016), 554–559.

[33] Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Curran Associates Inc., Red Hook, NY, USA, Article 294, 15 pages.