PROPOSAL Multi-Level Approach to Managing AI Crawler Behavior and Content Protection

by Gianna Brachetti-Truskawa

Summary

This document proposes a comprehensive, multi-layered strategy to protect website content from unauthorized use in AI training, particularly by Large Language Models (LLMs). The approach leverages existing web standards and proposals, and introduces new methods to communicate content usage restrictions effectively. We are not reinventing the wheel - but suggesting to combine methods for better protection.

Objectives

1. Establish new standards for the Robots Exclusion Protocol (RFC 9309) to control AI-oriented crawlers.
2. Implement multiple further layers of protection to ensure content creators' rights are respected.
3. Utilize existing directives where possible and introduce new ones where necessary.
4. Encourage widespread adoption through major platforms and content management systems.

Proposed Multi-Level Approach

Layers

Crawling directives alone typically rely on the voluntary compliance of web crawlers, which is not always guaranteed. Therefore, it is essential to implement a mix of methods to safeguard intellectual property effectively.
The suggested methods in this approach correspond to the following layers:

1. Crawling and Indexation Directives
2. Licensing and Legal Information
3. Copyright Protection and Monitoring
4. Access Request Management

Detailed Methods To Manage Access by LLMs

1. Enhanced Robots.txt Directives

Standard Implementation:
The standard robots.txt file allows website administrators to specify which parts of their site should be allowed or disallowed for crawling. This requires knowledge of the exact user agents used by generative AI bots to scrape websites.

```
# Crawling directives for all user agents unless specified separately in
this file
User-agent: *
Disallow: /private/

# Disallowing access for specific user agents associated with generative AI
User-agent: GPTBot
Disallow: /

User-agent: Google-Extended
Disallow: /
```

Maintaining the robots.txt file is straightforward and quick; however, it comes with certain disadvantages:

1. Exposure of Sensitive Information: The file is publicly accessible and readable by humans, which could inadvertently reveal paths containing sensitive information. This might encourage malicious activities such as DDoS attacks or other cybersecurity threats targeting those paths.
2. Non-Compliance by Bots: Not all bots obey robots.txt directives. There have been instances where bots, such as Perplexity AI, have crawled websites regardless of the directives.
3. Some bots might not have separate user agents for crawling or training their LLM, and current robots.txt directives do not allow to distinguish between the purpose of a bot's access requests to a site.

Therefore, it is strongly recommended to implement additional layers of protection.

User Agent-Specific Implementation:
Dynamic serving of different robots.txt files based on user agents, potentially excluding all AI crawlers requires server-side handling but adds an additional layer. For instance, Reddit recently implemented a strategy to block all search engines except Google by serving a different robots.txt files for Googlebot. Other user agents - including all other search engines - see their classic robots.txt file, disallowing all access and referencing their public content policy to inform about how they allow or disallow content to be accessed and used (Ryan Siddle, Merj Blog on Reddit's robots.txt Cloaking Strategy [1]).

Dynamic user-agent-based serving of robots.txt files requires server-side handling, which adds complexity. If a domain does not want to expose their strategy of which user agents they wish to block, there are some advantages.

Example robots.txt cloaking strategy:

1. Identify user agents or IP ranges of bots associated with generative AI.
2. Serve these bots a robots.txt with strict exclusion rules, protecting content you do not wish to be scraped.
3. For all other user agents or IP addresses, serve a "vanilla" version of your robots.txt, listing all rules for benevolent bots like search engines.

To cloak a robots.txt file for specific user agents, website administrators using Cloudflare can leverage Cloudflare Workers. If dynamic serving relies on user agents, validate user-agent strings to prevent spoofing.

2. Licensing Information in Robots.txt

Combining licensing information with other directives in robots.txt adds an additional layer of protection and complements other methods like HTTP headers and meta tags, creating a multi-faceted approach to safeguarding a website's content.

Instead of going with Reddit's approach of adding licensing information to human-readable pages, we suggest adding this information to an machine-readable format like JSON-LD or XML. This file can then be referenced in robots.txt, providing a clear and accessible way to communicate the terms under which a website's content can be used.
We suggest establishing a standard for licensing JSON-LD or XML files.

Example JSON-LD Licensing Information:

```
{
  "@context": "https://schema.org",
  "@type": "CreativeWork",
  "license": {
    "@type": "CreativeWork",
    "name": "Custom AI Usage License",
    "url": "https://www.example.com/ai-usage-license"
  },
  "usageInfo": {
    "aiTraining": "disallowed",
    "summarization": "allowed",
    "reproduction": "disallowed"
  }
}
```

Example reference to licensing information in robots.txt:

```
# Crawling directives for all user agents unless specified separately in
this file
User-agent: *
Disallow: /private/

# Disallowing access for specific user agents associated with generative AI
User-agent: GPTBot
Disallow: /

User-agent: Google-Extended
Disallow: /

# Licensing information
License-info: https://www.example.com/license-info.json
```

Advantages:

1. Legal Clarity: While robots.txt directives are not legally binding, adding licensing information can serve as a clear notice of a website's content usage rights and strengthen website owners' position in case of disputes over unauthorized use of their content.
2. Ease of Implementation: Updating robots.txt to include a link to a licensing information file is straightforward and does not require significant technical changes.

3. HTTP Headers

Using already established directives via HTTP headers such as X-Robots-Tag and Cache-Control allows to manage crawling and caching behavior. Additionally, we suggest to add new headers for custom legal notices and licensing information to provide clear and enforceable instructions to web crawlers.

HTTP Headers already in use
The use of HTTP headers like X-Robots-Tag to manage crawling behavior is already in practice. For example, Bing respects nocache or noarchive directives for AI crawling (Christopher Evans, How to Block AIs From Crawling Your Content [2]).

X-Robots-Tag:

```
X-Robots-Tag: noindex, noarchive, nosnippet
```

Cache-Control:

```
Cache-Control: no-store
```

We suggest bots associated with generative AI handle no-store as a directive to not cache or store any information retrieved from websites with this HTTP header.

Custom HTTP Headers for Legal Notices and Licensing Information
Establishing additional headers for legal and licensing information allows informing crawlers about the use of web content, even when a robots.txt file does not reference this information or is not present, or disobeyed.

Custom Legal Notice:

```
X-Legal-Notice: Unauthorized use of this content for AI training is
prohibited. See /terms-of-service for details.
```

License Information:

```
X-License-Info: CC BY-NC-ND
```

While these headers can provide more granular control, they still rely on the good faith of crawlers to respect them, as they do not provide technical enforcement. Restrictive creative common licenses such as CC BY-NC-4.0 or CC BY-NC-ND might not be pragmatic for LLMs (as discussed by Fili Wiese, Robots.txt is not the answer: Proposing a new meta tag for LLM/AI [3]). We would encourage LLMs to accept licensing details regardless.

In order to ensure that legal and licensing information is accessible (eg. in case a website owner wishes to make a legal case if their information has been used in an unauthorized manner), this information needs to referenced in more places, such as the footer (see more below). We would also encourage website administrators to establish further protective layers.

4. HTML Meta Tags

Meta tags are already used to handle indexing of web documents, and there are a few proposals to change them to be more specific for LLMs.
We would suggest that LLMs obey classic directives that are already established, such as: - which would, however, require that website administrators serve different HTML versions with these meta tags for specific user agents (see also: *User Agent-Specific Implementation:* above), which is more complex.

Some platforms like Deviantart have therefore suggested a straight-forward addition, , to specifically manage if LLMs are allowed to crawl and use web content (Deviantart, UPDATE

All Deviations Are Opted Out of AI Datasets [4]). This allows website administrators to allow indexing of a page for search engines, while disallowing the use of said content by LLMs.

In addition, we suggest adding licensing information to meta tags as well, as suggested by Fili Wiese [3], or referencing the location of license information (as suggested for *robots.txt* above):

Example Meta Tags:

```
<head>
  <meta name="robots" content="noindex, noarchive, nosnippet, noai">
  <meta name="license" content="https://www.example.com/license-info">
</head>
```

5. XML Sitemaps

XML sitemaps provide another way of reusing already established annotations to inform bots about whether access of a URL is allowed.
Website administrators could create a specific XML sitemap for LLMs - either serving it dynamically per user agent or IP address, or by applying a new standard for sitemaps specifically designed to handle crawling by bots associated with generative AI.

and can be reused to let LLMs know if the content is allowed to be crawled and used:

Example Restricted XML Sitemap:

```
<?xml version="1.0" encoding="UTF-8"?>
<urlset xmlns="http://www.sitemaps.org/schemas/sitemap/0.9">
  <url>
    <loc>https://www.example.com/restricted-page</loc>
    <lastmod>2024-07-24</lastmod>
    <changefreq>never</changefreq>
    <priority>0.0</priority>
  </url>
</urlset>
```

The XML sitemap could then be referenced in robots.txt, eg.:

```
# Crawling directives for all user agents unless specified separately in
this file
User-agent: *
Disallow: /private/

# Disallowing access for specific user agents associated with generative AI
User-agent: GPTBot
```

```
Disallow: /

User-agent: Google-Extended
Disallow: /

# Licensing information
License-info: https://www.example.com/license-info.json

# Generic sitemap for all bots
Sitemap: https://www.example.com/sitemap.xml

# Restricted sitemap for LLMs
Sitemap-restricted: https://www.example.com/restricted.xml
```

6. Terms of Service and Legal Notices

Update Terms of Serice (ToS) and display clear legal notices on the website to communicate content usage policies explicitly, providing a legally enforceable framework to protect against unauthorized AI training. This ensures that visitors and automated systems alike are aware of any restrictions and might help strengthen website owners' legal position in case of disputes.

Update ToS Example:

```
Prohibited Uses: The use of any content from this website for the purpose of training, developing, or operating artificial intelligence (AI) models, machine learning (ML) systems, or any other automated systems is strictly prohibited.
```

Displaying a visible notice or licensing information in the footer will make this information retrievable by every crawler.

Footer Notice Example:

```
<footer>
  <p>© 2024 [Name/Company]. All Rights Reserved. Unauthorized use of this content for AI training or data mining is strictly prohibited. See our <a href="/terms-of-service">Terms of Service</a> for more details.</p>
</footer>
```

Should content creators wish to specify licensing information instead, eg. CC BY-NC-ND, "All Rights Reserved" is contradictory and obsolete.

7. HTTP Status Codes

The most straight-forward way to enforce how a crawler can access a website is by HTTP status codes. We suggest to either use already established status codes, or establish new, LLM-specific HTTP status codes. The latter would offer a granular approach but would require widespread adoption and standardization.
Serving different HTTP status codes to specific user agents requires serverside handling, so it is not an easily accessible method.
The advantage is that it enables access management at a large scale, eg. for large organizations and web platforms.

Established HTTP status codes:

```
HTTP/1.1 403 Forbidden
HTTP/1.1 451 Unavailable For Legal Reasons
```

451 ([5] would serve as a hint for LLMs that there may be licensing restrictions.

New HTTP status code:

```
HTTP/1.1 452 LLM access denied
```

Aside from being challenging to establish as a new standard, another disadvantage of a dedicated LLM-related status code is that it would inform crawlers of the reason for access being denied. This could encourage to change IP addresses or user agents to avoid running into the error, while already established and widely used status codes remain unclear why they are being restricted.

Non-Technical Implementation Recommendations

Aside from establishing new web standards to allow to manage and monitor unauthorized use of website content in AI training, we should also try to achieve:

1. **Platform Integration:** We would like to encourage major, commonly used platforms (e.g., Cloudflare, WordPress, Google Search Console) to implement controls in their user interfaces.
2. **Education:** We should provide resources and guidance for content creators on implementing these protections, as well as how to protect their rights by filing DMCA notices for copyright infringement claims.
3. **Legal Framework:** We need to advocate for clear legal guidelines regarding the use of copyrighted material in AI training.

Conclusion

This multi-level approach provides a comprehensive strategy for protecting content from unauthorized use in AI training. By leveraging existing web standards and introducing new methods, content creators can effectively communicate their usage restrictions and maintain control over their intellectual property in the age of AI.

While robots.txt and other directives might seem like a straight-forward approach, they might not provide sufficient protection as they require LLMs to obey these rules.
Serverside monitoring and access management, eg. based on IP ranges, would be a more efficient approach to fully control if, how, and by whom web content is accessed and used.

References:

[1] https://merj.com/blog/investigating-reddits-robots-txt-cloaking-strategy
[2] https://www.semetrical.com/how-to-block-ai-from-crawling-your-content/
[3] https://searchengineland.com/robots-txt-new-meta-tag-llm-ai-429510
[4] https://www.deviantart.com/team/journal/UPDATE-All-Deviations-Are-Opted-Out-of-AI-Datasets-934500371
[5] https://developer.mozilla.org/en-US/docs/Web/HTTP/Status/451