AI, Robots.txt

Jaime Jiménez
jaime.jimenez@ericsson.com
Ericsson
Jorvas, Finland

Jari Arkko
jari.arkko@ericsson.com
Ericsson
Jorvas, Finland

ABSTRACT

Large Language Models (LLMs) and their use of Internet-sourced material present numerous technical, commercial, legal, societal, and ethical challenges. An emerging practice proposes extending the `robots.txt` file to enable website owners to declare if they wish to "opt-out" from having their site's content used in training AI models.

This paper explores the topic. We argue that the problem is much broader than the simple opt-out mechanism, given the coming new applications, the many different ways to access training material, different AI techniques, and the need to both facilitate access to training material and enable opting out from it.

KEYWORDS

AI, LLMs, learning, training, training material access

Reference:

Jaime Jiménez and Jari Arkko. 2024. AI, Robots.txt. In *submissions to the IAB AI-CONTROL workshop, September, 2024.* Washington DC, USA, 5 pages.

Internet Architecture Board (IAB) Workshop, September 2024, Washington, DC, USA

1 INTRODUCTION

The challenges posed by LLMs and their use of material from the Internet include preserving content creator rights, user awareness of what material has been used in training the LLMs, monetization models for accessing material for training purposes, and so on. Significant commercial and other values depend on proper treatment of these challenges.

The IAB "AI-CONTROL" workshop has called for contributions on one aspect of this, specifically about how website owners can express their preferences for their material to be used in training. A potential solution is emerging for expressing such preferences in the form of re-using the `robots.txt` file [13] for this new purpose.

The workshop call for papers requests a critical evaluation of whether such a solution is an appropriate mechanism for the task, whether content creators can realistically employ it for opting out from the use of their material, and what the use cases, requirements and other considerations in this space are.

Our contribution, this paper, stems from the authors' experience in working with various AI applications and technologies. We have also worked with Internet technologies beyond those relating to the open web, such as various APIs and other services.

This paper tries to convey the complexity of the topic. In particular, we believe that the needs for identifying AI robots and material they are allowed to use extend far beyond the case of a big, commercial and generic AI model training.

We review the different aspects of the problem space in Section 2. The state of today's practices in crawling for AI is discussed in Section 3. The interaction between crawlers and APIs is analyzed in Section 4. The emergence and capabilities of AI agents are covered in Section 5. An analysis of what could be feasible to do is detailed in Section 6.

2 THE LANDSCAPE

In particular, we want to draw attention to:

Ecosystem and Diversity

AI applications and the training they need are very diverse. Many are very different from the well-known general purpose models, such GPT-4 [15] or Mistral [10].

It is also important that we recognize the role of content creators as separate from content providers, such as users that provide content that is then later further added to website. The original content creator is often not the one offering the content on the web, in some cases they may not even have directly interacted with each other. There may or may not be agreements between them detailing the appropriate use of the material, and even when such agreements exist they can be rather one-sided and done begore GenAI became prominent.

The diversity of 'content' also extends beyond websites, e.g., material reached via web APIs, private data sets that may or may not be reachable on the web, user actions, network traffic, etc. are also all content.

Facilitating Training Data Retrieval

The opt-out angle is of course important, but perhaps it would be useful to view it as a part of a broader context. We wish to enable access to material that can be used in training and control that access based on preferences and legal and business constraints.

For instance, various APIs to information play a crucial role, but can their results be used in training, under what conditions, and how is this signalled?

Dynamic Behavior

The timing of changes in the ecosystem also creates an impact, e.g., policy changes after crawling are not easily taken into account. This is particularly significant as large AI models take a lot of time and effort to create and tend to be rather long-lived, whereas policies have changed recently quite rapidly. Separately created pools of training data (such as [5] or [2]) also take time to evolve.

There is often also a desire to base the AI applications on dynamic information, e.g., to use the up-to-date version of the information at AI inference time. One could also imagine the possibility of requesting removal of content from a trained AI model.

More generally, the use of different AI technologies impacts the way that they may use training data. For instance, Retrieval Augmented Generation (RAG) [14] techniques go beyond ingesting content in a model, as they simply allow for indexing and later retrieval of relevant material.

There are policy questions that regard to how content is ingested in a model vs. how it is used later at inference time.

The Needs of the AI's Users

The training process is obviously not just about the LLMs and the training material they ingest. The users of the AI matter as well. And for the users, it often matters more knowing what type of material is used than whether it can be used. For instance, what code licenses were used in the training data for an LLM that provides programming assistance? And does this have any implications on how the output from the model can be used?

This need is of course not just a matter for the content provider (and the original authors) and the AI model owner. It is a very much also an issue for the users of the AI model. They need to understand what type of material and from where has been used to train the model.

Practical Issues

There are difficulties in distinguishing different uses of crawling due to how crawling software is often structured, lack of good identification mechanisms, and many practical consent-related issues [9].

There are also inherent limitations of the `robots.txt` file, which is not designed for complex policy expressions, or even expressing policy based on actual material use rather than the name of the user agent.

Where APIs are used for information access, a lot of the API access configuration requires manual effort for setting up application keys and other context. Could this be done in-band in the API? Would there be ways to ensure better machine-readable definition of APIs as well as the acquisition and contracting of the relevant credentials?

There are also issues with how massive data sets can be efficiently and accurately crawled, e.g., can streams of updates be used instead of repeated crawling?

3 AI CRAWLING TODAY

AI crawling today is at least twofold: crawling for training AI models and crawling for real-time inference.

A significant portion of the training data for LLMs comes from large datasets like "The Pile" and "Common Crawl," which are aggregates of smaller datasets [5, 17]. These datasets, often hundreds of GBs in size, are filtered and processed into plaintext. Additional datasets are available on platforms such as Huggingface [7].

These datasets are created by web crawling and fetching entire websites. The crawled pages were then processed and cleaned using tools like "jusText" [11], which removes boilerplate content like navigation links, headers, and footers from HTML pages. This is because text containing full sentences is better suited for LLM tokenization and consumption.

The crawlers used while creating The Pile were pretty much the same as those employed for search indexing. They for example build on top of "Scrapy" [19] or the already mentioned "Common Crawl's Crawler" [3]. This similarity means that a content creator or host may not easily distinguish between a crawler used for search indexing and one used for LLM data ingestion - some crawlers serve both functions like those used by Internet search engines, social networking sites or some chat applications. Unauthorized AI crawlers can be already be identified like other bot traffic is; they typically exhibit high bounce rates and low session durations, with the caveat that their traffic often originates from a subset of common IP addresses associated with LLM vendors.

We propose that a significant portion of AI-related crawling today is likely for real-time **inference** . This is because training costs are prohibitively high, and obtaining current information requires fetching and integrating content into the LLM context. This integration is often done either through Retrieval-Augmented Generation (RAG) or its variations [4], with cached information, or directly within the context prompt as a real-time background task during user interaction with the LLM.

4 CRAWLERS AND APIS

LLMs are limited by their training data, which can quickly become outdated. In Generative AI, **tools** are used to provide LLMs with current information *at inference time*, primarily through APIs rather than web crawling [18], [16]. APIs are the standard interfaces for accessing third-party services and systems.

For instance, if an LLM is asked to list the top 10 movies of 2023, it would use the IMDb API to fetch this data instead of crawling the IMDb website. APIs deliver information in a structured format, making it easier for LLMs to process compared to raw HTML. APIs also allow retrieval of only the necessary information, improving efficiency.

APIs offer fine-grained control through mechanisms like API keys and OAuth tokens, making them ideal for real-time data retrieval due to their speed and efficiency. They are more reliable than websites, as they are designed for system-to-system interaction and are less likely to change frequently.

5 AI AGENTS ARE COMING

Originally, `robots.txt` was designed to help automatic agents find the right content on a website, long before its use for Search Engine Optimizations (SEO). Modern AI agents are similar in function but come equipped with advanced **reasoning capabilities**. These agents can decompose a high-level goal into smaller, actionable tasks and then execute those tasks based on the input, context, and available tools. The primary difference between these new agents and traditional web crawlers is their purpose: while traditional crawlers are primarily used for data training, AI agents are designed for real-time inference. In a sense, AI agents' "browsing" patterns more closely resemble a human user browsing the web or a client application retrieving data from a service, than a crawler performing massive data retrieval.

The reasoning aspects for LLM-enhanced agents is a current field of AI research, many techniques exists [1], [21], [23], [22] and they are being integrated into AI agents. They significantly improve the ability of a program to perform complex tasks that require strategic decision-making and problem-solving abilities.

For example a high level task could be: *"Send a message to Jaime telling him the top 10 movies in 2023"*; the AI agent would decompose this high level goal into smaller actionable items like looking through the user contact list, crafting the appropriate message and sending a

GET/POST with the message to the right API endpoint (e.g., Vonage Service or IMDB for example) with the appropriate payload. Agents can indeed interact with existing infrastructure via APIs but they can also crawl the web.

This is a major change, few years ago clients lacked semantic understanding of web content [12]. Today's LLM-enhanced hypermedia clients, or "smart clients," can *"understand"* web links and payload contents. These clients can infer that a link with `rel=comments` leads to comments about the current resource and know they should POST there to create a new comment. Such agents may only need the YAML OpenAPI [8] specification and a few examples to interact effectively with any well-constructed REST API without additional documentation.

As these new AI applications are deployed, we should recognize their distinction from traditional crawlers and develop guidelines that enable responsible use.

6 WHAT CAN WE DO?

Existing publications propose a standardized mechanism for explicit consent to fetch website content for training data [9]. The use of this data at inference time is intended for tooling and providing real-time responses based on current data to users, not necessarily for storage.

Currently, `robots.txt` files can exclude certain website paths if the crawler's user-agent identifies itself as an AI company without the proper copyright permissions. It is not uncommon to see the `Disallow: /` directive used to block AI robots on major websites.

Improving Filtering of AI Crawlers

To better filter AI crawlers, we could implement the following improvements:

- *Introduce a Principle-Based Approach:* Instead of a "user-agent name" based approach, `robots.txt` could adopt a principle-based approach. Policies could be applied based on the intended use of the data (e.g., storage, indexing, training, inference), the type of content, and other relevant criteria.
- *Extend the Syntax:* The current `robots.txt` syntax is too limited for complex policy settings. It is necessary to express which intended uses are appropriate. Other more complex policies might indicate specific user agents who, for instance, have full access per an agreement. Policies might also indicate the type of license that applies to the content.

Training Policies Beyond Crawling

As is clear from previous sections, data fed to the LLMs is not limited to traditional crawling. Expressing policies about intended use of data would be useful in other contexts as well:

- *Policy objects in APIs:* It would be useful to be able to express policies within APIs, for instance as part of a machine-readable API definition or on a case-by-case basis upon making an API call. This would be particularly useful for inference-time AI applications.
- *Policy as a user choice:* Users should control how their content is used, including specifying appropriate intended uses. This choice can be part of their overall service settings.

Facilitating AI Crawling

We may also consider a different direction, that of simplifying the crawling for AI crawlers. For instance, we could propose the inclusion of standardized metadata in the `robots.txt` file, applicable for both inference and training.

In its simplest form, this could involve a redirect to a `/norm` path for allowed user-agents, where readily formatted text files with the content would be provided. Agents would simply ignore other paths and use that one only.

Contents under the `/norm` path would be normalized in plaintext for AI systems. This includes providing clean text (stripping out HTML tags, scripts, styles, and other non-content elements) and removing duplicate content and URLs. The server may also present content differently based on policy or other criteria.

In order to incentivize content owners to share their content, methods for incorporating standardized payment requests within HTTP 402 responses or other similar mechanisms could be revisited for this purpose [6] and the `robots.txt` file could be used to signal its availability.

Standardizing API Interactions for AI Agents

As introduced in Sections 4 and 5 as APIs are commonly used by AI systems, one problem that arises in

this space seems to be the lack of standardized in-band configuration interfaces.

- Automatically setting webhooks via the API interface instead of through a configuration web portal.
- Reliably setting up security settings to interact with the API, such as obtaining web access tokens via the same in-band API interface.
- Enabling API entry point discovery via a standard API catalog to allow AI agents to autodiscover and interact with relevant entry points. The HTTP API Working Group is already addressing some of these issues, as seen in [20].

7 ACKNOWLEDGMENTS

We'd like to thank Ericsson for their support of this work. We also appreciate Mark Nottingham, Suresh Krishnan, Mirja Kühlewind, Zeljka Sotra, Elif Ustundag Soykan, Carsten Bormann and Lorenzo Corneo for their valuable discussions on this topic.

REFERENCES

[1] Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michał Podstawski, and et al Gianinazzi. [n. d.]. Graph of Thoughts: Solving Elaborate Problems with Large Language Models. ([n. d.]).

[2] Vadim Borisov and Richard H. Schreiber. 2024. Open Artificial Knowledge. arXiv:2407.14371 [cs.CL] https://arxiv.org/abs/2407.14371

[3] Commoncrawl. 2024. Commoncrawl Crawler. (2024). https://github.com/commoncrawl/commoncrawl-crawler

[4] Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, and Jonathan Larson. 2024. From Local to Global: A Graph RAG Approach to Query-Focused Summarization. arXiv:2404.16130 [cs.CL] https://arxiv.org/abs/2404.16130

[5] Leo Gao, Stella Biderman, Sid Black, and et al. 2020. The Pile: An 800GB Dataset of Diverse Text for Language Modeling. (2020). https://arxiv.org/abs/2101.00027

[6] Adrian Hope-Bailie. 2017. HTTP-Payments. Internet-Draft draft-hope-bailie-http-payments-00, Internet Engineering Task Force. https://datatracker.ietf.org/doc/html/draft-hope-bailie-http-payments-00 Expired May 4, 2018.

[7] Huggingface. 2024. Huggingface Datasets. (2024). https://huggingface.co/datasets

[8] OpenAPI Initiative. 2023. OpenAPI Specification. https://www.openapis.org/

[9] D. Ippolito and Y.W. Yu. 2023. DONOTTRAIN: A Metadata Standard for Indicating Consent for Machine Learning. (2023). https://genlaw.github.io/CameraReady/42.pdf

[10] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7B. arXiv:2310.06825 [cs.CL] https://arxiv.org/abs/2310.06825

[11] jusText. 2024. jusText. (2024). https://pypi.org/project/jusText/

[12] Jeff Knupp. 2014. Why I hate HATEOAS. (2014). https://jeffknupp.com/blog/2014/06/03/why-i-hate-hateoas

[13] M. Koster, G. Illyes, H. Zeller, and L. Sassman. 2022. Robots Exclusion Protocol. RFC 9309. https://doi.org/10.17487/RFC9309

[14] Patrick Lewis, Ethan Perez, Aleksandra Piktus, and Fabio Petroni. 2020. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 9459–9474. https://proceedings.neurips.cc/paper_files/paper/2020/file/6b493230205f780e1bc26945df7481e5-Paper.pdf

[15] OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, and et al. 2024. GPT-4 Technical Report. arXiv:2303.08774 [cs.CL]

[16] A. Parisi, Y. Zhao, and N. Fiedel. 2022. TALM: Tool Augmented Language Models. (2022). https://doi.org/10.48550/arXiv.2205.12255

[17] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. arXiv:1910.10683 [cs, stat]

[18] T. Schick and et al. 2023. Toolformer: Language Models Can Teach Themselves to Use Tools. (2023). https://doi.org/10.48550/arXiv.2302.04761

[19] Scrapy. 2024. Scrapy. (2024). https://github.com/scrapy/scrapy

[20] Kevin Smith. 2024. api-catalog: a well-known URI and link relation to help discovery of APIs. Internet draft draft-ietf-httpapi-api-catalog-03 (Work In Progress). arXiv:2404.16130 https://datatracker.ietf.org/doc/html/draft-ietf-httpapi-api-catalog-03

[21] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. https://doi.org/10.48550/arXiv.2201.11903 arXiv:2201.11903 [cs]

[22] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. https://doi.org/10.48550/arXiv.2305.10601 arXiv:2305.10601 [cs]

[23] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing Reasoning and Acting in Language Models. https://doi.org/10.48550/arXiv.2210.03629 arXiv:2210.03629 [cs]