Creating Searchable Web Page Snapshots using Semantic Technologies

Radek Burget[1][0000−0001−5233−0456] and Hamza Salem[1][0000−0002−9143−5231]

Brno University of Technology, Faculty of Information Technology, Bozetechova 2, 61266 Brno, Czechia
{burgetr,salem}@fit.vut.cz

Abstract. For many applications, it is necessary to create snapshots of web pages that accurately describe how the page appeared in a browser at a given point in time. Storing the original code (even when including all referenced resources) and creating bitmap screenshots have many drawbacks when it comes to searching, viewing and manipulating such snapshots. In this paper, we demonstrate a different approach that uses a remotely controlled web browser for rendering web pages. We capture the complete information about the rendered page and all pieces of its content, transform it to an explicit RDF-based model representation stored in a repository. Then, the stored page models may be examined using an interactive web-based tools, exported in different formats, linked with other data sources, and queried using SPARQL.

Keywords: First keyword · Second keyword · Another keyword.

1 Introduction

In many situations related to working with information published on the web, we need to create snapshots of web pages that capture the web page as it was rendered by the web browser at that particular moment as accurately as possible. The typical motivation for this is to document the source of information obtained from the web for different purposes, that include

- Monitoring web content that changes over time (such as product prices, real estates, etc.)
- Creating datasets for the evaluation different web content processing algorithms (such as data extraction, page segmentation, etc.)[1, 2]
- Preparing training data for machine learning-based algorithms that analyze the visual presentation of the page [4], and many more.

In many of these scenarios, the visual presentation of the content is as important as the content itself. In product monitoring, for example, we want to answer questions retrospectively, such as: Was the product price visually marked as discounted? Was the information presented in the right context? Also, many automated data extraction methods depend on the visual presentation of the

content and for their evaluation, it is necessary to create the corresponding datasets [4, 6, 7]. In such scenarios, it is not sufficient to store the text content or the HTML code of the source page, as we lose the visual context.

Traditional approaches to web page archiving are typically based on saving the HTML code of the web page together with other resources that are necessary to re-render the page again. This is how the Internet Archive[1] works, for example. However, in order to reconstruct the visual appearance of the page from such a snapshot, we need to get a web browser to "replay" the archived content. This is a time-consuming task, and the result may not fully match the original appearance of the page because browsers evolve over time, they may run in different environments with different fonts available, and the pages may contain links to third-party dynamic content (such as advertisements) that cannot be efficiently stored [5]. The other option is to take a bitmap screenshot of the rendered page, which is easy to save and view later, but its content cannot be efficiently searched and/or annotated.

In this demo, we present a complementary approach that is based on capturing the complete information about a rendered page and each piece of its content from the web browser, transforming it into an explicit RDF model, and storing it in a central repository. We show, how the snapshots can be created, inspected via an interactive web interface, exported in a variety of formats ranging from XML and Turtle to plain bitmap screenshots, and possibly linked to other data such as structured data (JSON-LD) annotations.

2 Related Work

The traditional formats for archiving web pages include mainly the standardized WARC[2] (ISO 28500) that stores the original HTML code together with the additional resources such as style sheets, scripts, images and other content referenced from HTML. The MHTML file format [3] and the HAR file format[3] are based on similar principles. Published datasets of web pages typically use one of these formats. For example, SWDE [2] or WEIR [1] consist of HTML files only, Kiesel et al. [5] use the WARC format, and Hotti et al. [4] use MHTML files. To annotate the captured pages with additional information, separate files in specific formats (such as text files or bitmap screenshots) must be used [2, 1, 6] or specific attributes can be added to the original HTML files [4].

The tool presented in this paper does not aim to replace the existing possibilities mentioned above, but it provides an additional way to create an "interactive screenshot" of the page: A representation of the rendered page that is explicitly described, can be browsed, linked to other data, and queried using the standard SPARQL language, allowing important information to be extracted from the collected set of pages.

[1] http://archive.org/
[2] http://bibnum.bnf.fr/WARC/
[3] A W3C proposal that has never become a standard but is supported by software tools such as Playwright.

3 Creating an RDF Model of a Rendered Page

4 Implementation and Applications

5 Applications

6 Conclusions

Acknowledgements The work is supported by the Brno University of Technology project "Application of AI methods to cyber security and control systems", no. FIT-S-20-6293.

References

1. Bronzi, M., Crescenzi, V., Merialdo, P., Papotti, P.: Extraction and integration of partially overlapping web sources. Proc. VLDB Endow. **6**(10), 805–816 (aug 2013). https://doi.org/10.14778/2536206.2536209, https://doi.org/10.14778/2536206.2536209
2. Hao, Q., Cai, R., Pang, Y., Zhang, L.: From one tree to a forest: A unified solution for structured web data extraction. p. 775–784. SIGIR '11, Association for Computing Machinery, New York, NY, USA (2011). https://doi.org/10.1145/2009916.2010020, https://doi.org/10.1145/2009916.2010020
3. Hopmann, A., Palme, J., Shelness, N.H.: MIME Encapsulation of Aggregate Documents, such as HTML (MHTML). RFC 2557 (Mar 1999). https://doi.org/10.17487/RFC2557, https://www.rfc-editor.org/info/rfc2557
4. Hotti, A., Risuleo, R.S., Magureanu, S., Moradi, A., Lagergren, J.: Graph neural networks for nomination and representation learning of web elements (2021). https://doi.org/10.48550/ARXIV.2111.02168
5. Kiesel, J., Kneist, F., Alshomary, M., Stein, B., Hagen, M., Potthast, M.: Reproducible web corpora: Interactive archiving with automatic quality assessment. J. Data and Information Quality **10**(4) (oct 2018). https://doi.org/10.1145/3239574
6. Kiesel, J., Kneist, F., Meyer, L., Komlossy, K., Stein, B., Potthast, M.: Web page segmentation revisited: Evaluation framework and dataset. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management. p. 3047–3054. CIKM '20, Association for Computing Machinery, New York, NY, USA (2020). https://doi.org/10.1145/3340531.3412782
7. Milička, M., Burget, R.: Information extraction from web sources based on multi-aspect content analysis. In: Semantic Web Evaluation Challenges, SemWebEval 2015 at ESWC 2015. pp. 81–92. No. 548 in Communications in Computer and Information Science, Springer International Publishing, Portorož, SI (2015)