Towards Efficient Construction of a Traceable, Multimodal, and Heterogeneous Data Warehouse

Antoine Gauquier
antoine.gauquier@ens.psl.eu
DI ENS, ENS, CNRS, PSL University & Inria
Supervised by: Ioana Manolescu and Pierre Senellart
Paris, France

ABSTRACT

This paper describes my PhD project and its advancement in the first months of its first year. The PhD aims to study holistic methods for building, populating, and exploiting warehouses of heterogeneous content. Each warehouse is characterized by a specification of the types of content we search for; a set of websites in which to search for the content; a set of dedicated methods to analyze and understand the content, including to establish or find links that connect different pieces of content. AI and uncertainty are naturally involved in these steps. We present the overall thesis aims, as well as encouraging preliminary results for one use case: the acquisition of statistical data resources from French government websites, leveraging reinforcement learning.

VLDB Workshop Reference Format:
Antoine Gauquier. Towards Efficient Construction of a Traceable, Multimodal, and Heterogeneous Data Warehouse. VLDB 2024 Workshop: VLDB Ph.D. Workshop.

1 INTRODUCTION

The availability of ever-increasing amounts of human-authored data, particularly published on the Web, opens up countless opportunities for users to find, understand, analyze, and repurpose this data. Doing so requires automatic methods for gathering and understanding this data. These data resources are built with a variety of formats, including unstructured data (text), semi-structured data (JSON, HTML, XML) or structured data (CSV, tabulated data). There is a wealth of standalone methods focused on acquiring, analyzing, and understanding such data resources. Unfortunately, most of these only address the processing of homogeneous data, usually in a single format. Moreover, there is (to our knowledge) no work that treats each of these stages in a single pipeline, where each from acquisition to exploitation benefits from the others. This end-to-end integration makes the information traceable, enabling us to estimate its trustworthiness. Seeing these data resources in a multimodal manner (for instance, by seeing a PDF simultaneously as raw text and rendered bitmap images) should minimize information loss throughout the processing phase, resulting in better quality information. The goal of this thesiss research is the *efficient construction of a traceable, multimodal, and heterogeneous data warehouse*. In particular, we aim to apply our developed solutions to *statistical data journalism*, to build a system that automatically verifies online claims on the basis of trustworthy data resources.

We first present the core problem targeted in this thesis. Then we focus on the *data acquisition* part which is our first focus: to devise efficient and focused Web crawling for retrieving data from human-produced resources. We outline our methodology and the first, promising results, obtained from a set of websites proposed to us by application domain experts.

2 PROBLEM

The thesis aims to develop models, algorithms, and a system capable of *automatically building rich multimodal data warehouses*. Such a warehouse could contain documents that are *unstructured* (mainly in textual form), *semi-structured* (PDF documents, pages or websites; content from social networks, etc.) and *structured* (datasets that are tabulated or presented as *aggregation cubes*, in CSV, OpenDocument formats, etc.). This work is divided in several subtasks, each of which relies on modern artificial intelligence methods such as: reinforcement learning for content acquisition (especially for Web content acquisition), deep learning for text or image content analysis, knowledge representation, etc.

Data acquisition. We aim at building data warehouses coming from multiple, heterogeneous data resources, in particular made available through the Web. Thus, we start by devising *crawling* methods, which should enable us to obtain the kind of data we are looking for, while being as efficient as possible. In this context, *efficiency* corresponds to *minimizing the cost of the crawling task*, whether in terms of running time, number of visited webpages, requests sent to the server, volumes of exchanged data, etc., all the while aiming for a maximal amount of *high-quality data* retrieved. The quality of a data resource is defined depending on the application: for instance, in our data journalism scenario, the quality could be proportional to the size (number of data points), or we could consider well-structured, CSV data resources as of better quality than tables that have to be extracted from PDF files, etc. It is also important to be able to *estimate* the quality of a data resource before actually acquiring that data.

In our data journalism target scenario, the goal is to gather *statistics data from multiple, heterogeneous websites* where French public administrations have compiled and published them. While all these statistics are legally speaking Open Data, they are overwhelmingly not structured in RDF, but rather in CSV, Office formats, PDF, etc. The statistics are useful to journalists for their *fact-checking work*,

Proceedings of the VLDB Endowment. ISSN 2150-8097.

using semi-automated tools [3, 16]. Journalists proposed a list of websites of interest, belonging to various French ministries, government offices, statistical services, etc. The methodology and the first results obtained through our work are presented in Sections 3 and 4. Before concluding, we discuss relevant work in Section 5.

Data extraction and analysis. We aim to populate our target warehouses with *interconnected* data. For instance, in the data journalism scenario, some files contains statistical data, e.g., the unemployment rates over all French departments in 2023, while other files contain *dimension dictionaries*. Specifically, statisticians who author (publish) datasets sometimes use abridged names (codes) for values of the data dimensions, e.g., F13 may designate Female unemployed people in the age group numbered 13, corresponding to 42 to 46 years old. In such cases, a separate file (or a separate sheet of the same file) contains the mapping (decoding) of codes such as F13, into a natural-language description. For each statistics file, we should also identify the corresponding dictionary, and store the connection between the two. As the above examples show, the extraction and analysis task is made more complex by the fact that the data collected is heterogeneous: both in format (HTML, PDF, CSV, XLS, RDF, etc.) and in content. The challenge is therefore to come up with general methods that can handle the differences in formats, by still extracting information that can be put in relation in a common, comprehensive format.

Operating the data warehouse. The last subtask consists in storing and querying all of this data in its variety, while taking benefit from the relations identified. A crucial aspect is to keep track of the *origin (or provenance)* of each piece of data. Maintaining provenance and uncertainty throughout the AI processes involved in building and operating the warehouse contributes to the explainability and confidence one can have in results extracted from the warehouse. Last but not least, we aim to develop interfaces friendly to non-technical users, to support their work with our content warehouse.

3 METHODOLOGY

As this PhD work is still in its early stages, in this section, we present our methodology for the first task: data acquisition, or more specifically, efficient extraction of human-authored data resources from the Web. To this end, we are using reinforcement learning methods for efficient Web crawling, with a broad definition of efficiency as explained in Section 2.

Our current goal is: given a website known to contain data resources of interest, retrieve all (or a large proportion of) these resources within a limited *budget*. Here, budget can be interpreted in different ways; currently we consider both the number of requests sent to the server and the volume of data exchanged with it. These two aspects complement each other, as the first takes into account the fact that we have to pay a minimum waiting time between two requests, while the second integrates the amount of information carried by the resources. Exhaustively crawling all the pages of a website to identify the data resources it hosts may be: (i) a significant waste of resources, given that websites may contain numerous pages lacking data resources of interest; (ii) prohibitively slow; and (iii) simply unfeasible due to crawling limits set by the website. To efficiently direct a partial search yet recover all the data resources we want, we use *reinforcement learning (RL)*. We choose this method because on one hand, we have no a priori knowledge of the cartography or content of the website, and on the other hand, the structure of the sites can change drastically from one site to another, thus preventing us from implementing a fixed generic method applicable to all websites.

Reinforcement learning, in its most classical form, is based on a *Markov Decision Process (MDP)*, modeling the *environment*, *states* and *actions* that the *agent* (the reinforcement learning algorithm) has to take into account. The natural way to model this MDP in the context of Web crawling is to consider each set of pages crawled as a state, and for such a state, each available link leading to another page an action. However, in the general case, a website hosts many pages and many links, making the MDP model too large. Therefore, usage of RL requires that we model our problem more compactly, as follows. First, we approximate the website as a *single-state* environment, where *all pages are the same from the agent perspective*. Of course, the set of links (and their number) that can be followed change after each action, since, in reality, new links are progressively discovered and since we do not want to crawl a resource twice. Such an environment can be handled using *multi-armed bandit* algorithms [2]. This family of algorithms gets its name from real-world slot machines, and consists in choosing an arm (among a set of several arms) to get a *reward*, whose value is not known in advance. The goal is therefore to determine which of the arms should be pulled, in order to maximize the obtained rewards in the long run. In its classical form, the multi-armed bandit problem has a fixed set of arms (or actions) at each *episode*, and even assuming a finite horizon, the number of episodes has to be quite high for the method to enjoy formal success guarantees. Unfortunately, in our context, the action set changes from one episode to another. To adapt the methods to our environment, we consider a *sleeping bandit* approach, that supports changing the set of available arms during the crawl at a finite horizon. Specifically, we rely on the *Awake Upper-Estimated Reward (AUER)* algorithm [10], which is an adaptation of the *Upper Confidence Bound (UCB)* of [2].

Our next issue is to limit the number of actions. Observe that in the primitive modeling where an action (arm to be activated) is "crawling along one link", no agent learning can take place, since this arm is only available once (we do not want to crawl the same page twice). As mentioned in Section 5, we want to take advantage of the structure of webpages. This is why we work with *sets of links* leading to the pages not visited yet, that we group by the structural (*Data Object Model or DOM*) path leading to the links, within the HTML webpage where each link is discovered. Intuitively, this leverages the assumption that links sharing similar paths in the HTML DOM structure are likely to lead to the same kind of pages. By doing so, we hope to separate sets of links leading to pages that contain data resources, from sets that do not.

Any RL algorithm follows a *goal*, which defines its *reward*. Since we want to identify data resources within different pages on a given website, we first set the reward to be the number of data resources discovered by following a given link. This simple initial choice gave good results in our first experiments (see Section 4).

For an ethical use of the websites we experiment on, *for our experiments*, we have fully acquired and replicated locally (on our computer) the websites on which we try our methods. Thus, we crawl each website only once, while running (successive versions)

of our algorithms with different parameters, strategies, etc., on our local copy. Also, having the exhaustive crawl enables to correctly assess the recall of our crawl, as we discuss next.

4 PRELIMINARY RESULTS

Baselines. As baseline, we use a *naive* (random) crawler. At each new *episode* of the crawling phase, the crawler randomly selects one of the links identified by crawling all previous pages, in what we call its *frontier* (the set of all links it discovered but not followed yet). We also implement a more standard algorithm for exhaustive crawling of websites, which is *Breadth-First Search (BFS)*, where we crawl pages in their order of discovery, i.e., the frontier is a page queue. This method is less naive than the random crawler, yet does not benefit from any kind of learning.

Visual and numerical evaluation methods. Since our goal is to retrieve all data resources of a given website crawling as little as possible, we use two main metrics: *the number of retrieved data resources vs the number of requests made to the server*, and *the volume of retrieved data resources vs the volume of requests on everything except data resources.* Plotting both metrics yields a visual evaluation of the quality of a crawling approach. For the first, the closer the curve is to what we call a (utopian) *"omniscient" crawler* (which knows the location of all data resources in advance), the better. For the second, the closer the curve is to the top left corner of the graph (i.e., as much data as possible is discovered in as little volume of crawled HTML pages as possible), the better. Let W, A be two crawling algorithms, with W being a baseline, e.g., random. To A we associate: $r_W(A) = \frac{\mathcal{A}(A)}{\mathcal{A}(W)}$ where $\mathcal{A}(x)$ is the area under the curve of x. If $r(A) < 1$, algorithm A gives results that are worse than algorithm W, i.e., worse than a random crawler.

We also track the number of pages/volume of data an algorithm visits/retrieves before crawling $Y\%$ of the total (volume of) data resources on a Web site, with $x \in \{50, 90\}$. Smaller values mean fewer pages/less volume had to be crawled, thus are better.

Selected data and results. Among two dozen websites suggested by journalists, we selected for our tests four varied ones. https://www.collectivites-locales.gouv.fr/ has many data resources compared to the number of distinct pages it contains (more than two data resources for each HTML webpage), and relatively few HTML webpages (around 1 700). In https://www.cnis.fr/, the ratio is 1.3, but it has three times more pages (around 5 300). https://www.justice.gouv.fr/ has a ratio close to 0.36, for approximately 42 000 pages. Finally, https://www.education.gouv.fr/ has a very small ratio (around 0.12) and a large amount of HTML pages (around 82 000).

Figure 1 plots the number of retrieved data resources against the number of crawled resources, and the volume of requests dedicated to data resources retrieving against the volume of requests on everything else on https://www.cnis.fr/, for each algorithm. The corresponding ratios of areas under the curve using the random crawler as point of comparison, as well as the number of pages we have to visit to respectively retrieve 50%, and 90% of data resources (volume), are presented in Table 1, for the four websites.

These different evaluations show that our approach is significantly more efficient in comparison to the baselines we consider. First, visually, we see that, for each website, the curves describing our approach (in blue for the two plots) are always above the baselines curves. Regarding numerical features, we see that our approach is always better than the other two, since it always gets the highest area ratio for the four websites, and the lowest number of pages to visit to respectively reach 50%, and 90% of data resources and data resources volume for each website.

5 STATE OF THE ART AND LIMITATIONS

In the area of data acquisition from the Web, prior works address the question of *focused Web crawling* for specific subjects of interest [4, 9], in particular, taking advantage of the structure of the pages in a website [3, 6], while other crawl the content of databases hidden behind forms [8, 14]. Some works consider retrieval data resources from social networks through their APIs [3], or propose methods for assessing the quality of the retrieved data [7]. These works do not answer the question of minimal-effort effort data resources retrieval. [4, 9] exploit topic-specific focused crawling, while we need methods that are topic-independent, efficient in all contexts. [8, 14] offer efficient methods for retrieving data behind forms, while our resources can be linked from any type of HTML page, according to the sites involved in the data journalism application.

In information extraction and analysis of heterogeneous data resources, *multimodal machine learning* has been used to extract information from different kinds of content, e.g., text, images, etc. [17] uses documents' textual content, and their layout information, to understand document images. Other works [12, 18] propose datasets to be used by multimodal models to infer the composition of documents on specific topics. To link different fragments of extracted information, *Open Information Extraction (OIE)* [11] has been used to identify named entities [1] and relations [15] from text, and from textual fields in (semi-)structured data.

Data heterogeneity is a major challenge, rarely addressed. Information extraction methods in the literature, even multimodal ones, tend to focus on a single type of data resource: [12, 17, 18] only deal with PDF documents for instance. Our work targets much more varied content than can be handled by current-days warehouses, and has more ambitious goals understanding it, e.g., tables.

Data warehousing [5] has led to industrial-strength methods and platforms for storing and processing huge amounts of data, typically relational (tabular) data. In [1], heterogeneous data resources (text, structured, or semi-structured, i.e., JSON, RDF, etc.) are loaded into a single graph-structured warehouse. Such systems are not concerned with the acquisition and extraction of information from various resources; instead, the data is considered given. Also, our work targets (also) more complex content, such as PDFs comprising tables or scientific theorems, etc.

6 CONCLUSIONS AND FUTURE WORK

This PhD thesis is about efficiently building a traceable, multimodal and heterogeneous data warehouse. This involves three tasks: acquisition of heterogeneous data resources, extraction and analysis of information retrieved from these resources, and finally building and exploiting the warehouse. The use of AI introduces uncertainty; capturing and storing provenance in the warehouse will enable debugging, repairs, and better confidence in the warehouse content. We presented results of our *heterogeneous data resources acquisition* task. Specifically, we trained an efficient *crawler* through the use of

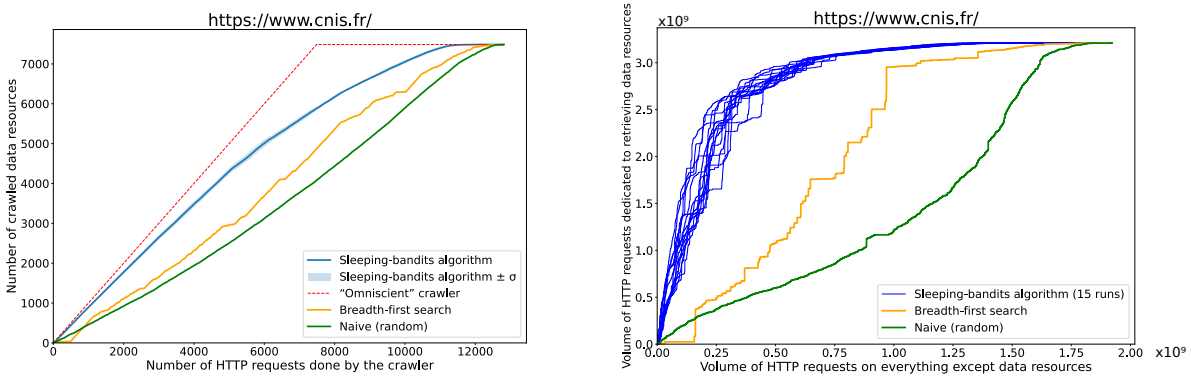

Figure 1: Number of crawled data resources in function of all crawled resources and of the volume dedicated for data resources retrieving in function of data volume dedicated to the rest, for different algorithms.

Table 1: Performance of our RL approach (mean over 15 runs) compared with baselines (random crawler and BFS). Four values are shown for each metrics, respectively computed on the websites: https://www.collectivites-locales.gouv.fr/, https://www.justice.gouv.fr/, https://www.cnis.fr/, and https://www.education.gouv.fr/.

			% of X to identify a fraction Y of the data of the Web site			
	Area ratio ($r_{\text{Naive}}(\cdot)$) for …		X = "HTTP requests"		X = "Volume of HTML requests"	
Algorithm	… number of requests	… volume of requests	Y = 50% (resources)	Y = 90% (resources)	Y = 50% (volume)	Y = 90% (volume)
Naive crawler	1.00 - 1.00 - 1.00 - 1.00	1.00 - 1.00 - 1.00 - 1.00	53.3 - 47.7 - 54.3 - 63.4	89.2 - 83.2 - 87.0 - 95.8	62.3 - 22.5 - 63.8 - 46.6	89.3 - 70.1 - 82.7 - 92.6
BFS	1.01 - 1.01 - 1.11 - 1.37	1.04 - 1.00 - 1.45 - 1.33	51.9 - 47.0 - 47.3 - 34.5	87.5 - 80.8 - 81.7 - 95.5	49.8 - 20.7 - 33.2 - 16.4	80.8 - 68.0 - 50.4 - 92.8
Our approach	1.12 - 1.35 - 1.32 - 1.58	1.72 - 1.14 - 1.89 - 1.48	41.2 - 23.5 - 33.8 - 17.8	79.1 - 45.5 - 72.3 - 69.4	14.7 - 8.7 - 9.0 - 5.7	57.5 - 35.4 - 28.5 - 68.0

sleeping bandits, aiming to identify as many statistical datasets as possible while keeping the crawling effort to a minimum. We show that our method is much more effective than naive baselines.

Future refinements of our data acquisition method include testing another *sleeping bandit* algorithm, especially one that is better adapted to the distribution of our *reward* system. Also, we plan to improve our way to assess the quality of a data resource, since at the moment, we give the same importance to all data resources, irrespective of their content. We also work on a third baseline that is taking advantage of the *DOM* paths as described above, but without using any learning techniques; we believe that both the use of *DOM* paths and reinforcement learning are actually useful. We will experiment on a wider range of websites, hopefully showing that our method generalizes well. We also aim to handle the three tasks in a unified framework, since each can impact the others: for example, extracted elements can be used to feed the way in which new elements are acquired; queries carried out on the warehouse can lead to new data being added and new acquisitions being launched, etc. Finally, we want to address applications other than statistical data journalism, in particular the automatic extraction of theorems and proofs from scientific papers to build a queryable knowledge base of mathematical results, building upon [13].

REFERENCES

[1] Angelos Christos Anadiotis, Oana Balalau, Catarina Conceição, Helena Galhardas, Mhd Yamen Haddad, Ioana Manolescu, Tayeb Merabti, and Jingmao You. 2022. Graph integration of structured, semistructured and unstructured data for data journalism. *Information Systems* 104 (2022), 101846.

[2] Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. 2002. Finite-time Analysis of the Multiarmed Bandit Problem. *Machine Learning* 47, 2 (01 May 2002), 235–256.

[3] Oana Balalau, Simon Ebel, Théo Galizzi, Ioana Manolescu, Quentin Massonnat, Antoine Deiana, Emilie Gautreau, Antoine Krempf, Thomas Pontillon, Gérald Roux, and Joanna Yakin. 2022. Fact-checking Multidimensional Statistic Claims in French. In *TTO 2022 - Truth and Trust Online*.

[4] Soumen Chakrabarti, Martin van den Berg, and Byron Dom. 1999. Focused crawling: a new approach to topic-specific Web resource discovery. *Computer Networks* 31, 11 (1999), 1623–1640.

[5] Pravin Chandra and Manoj K. Gupta. 2018. Comprehensive survey on data warehousing research. *International Journal of Information Technology* 10, 2 (01 June 2018), 217–224.

[6] Muhammad Faheem and Pierre Senellart. 2015. Adaptive Web Crawling through Structure-Based Link Classification. In *ICADL*. 39–51.

[7] Alban Galland, Serge Abiteboul, Amélie Marian, and Pierre Senellart. 2010. Corroborating Information from Disagreeing Views. In *WSDM*. 131–140.

[8] Inma Hernández, Carlos R. Rivero, and David Ruiz. 2019. Deep Web crawling: a survey. *World Wide Web* 22, 4 (2019), 1577–1610.

[9] Diligenti, Michelangelo and Coetzee, Frans and Lawrence, Steve and Giles, C Lee and Gori, Marco and others. 2000. Focused Crawling Using Context Graphs.. In *VLDB*. 527–534.

[10] Robert Kleinberg, Alexandru Niculescu-Mizil, and Yogeshwer Sharma. 2010. Regret bounds for sleeping experts and bandits. *Machine Learning* 80, 2 (01 Sept. 2010), 245–272.

[11] Keshav Kolluru, Vaibhav Adlakha, Samarth Aggarwal, Mausam, and Soumen Chakrabarti. 2020. OpenIE6: Iterative Grid Labeling and Coordination Analysis for Open Information Extraction. In *EMNLP*. ACL, 3748–3761.

[12] Minghao Li, Yiheng Xu, Lei Cui, Shaohan Huang, Furu Wei, Zhoujun Li, and Ming Zhou. 2020. DocBank: A benchmark dataset for document layout analysis. *arXiv preprint arXiv:2006.01038* (2020).

[13] Shrey Mishra, Lucas Pluvinage, and Pierre Senellart. 2021. Towards Extraction of Theorems and Proofs in Scholarly Articles. In *DocEng'21*.

[14] Cheng Sheng, Nan Zhang, Yufei Tao, and Xin Jin. 2012. Optimal algorithms for crawling a hidden database in the web. *Proc. VLDB Endow.* 5, 11, 11121123.

[15] Prajna Upadhyay, Oana Balalau, and Ioana Manolescu. 2023. Open Information Extraction with Entity Focused Constraints. In *EACL*. Dubrovnik, Croatia.

[16] You Wu, Pankaj K. Agarwal, Chengkai Li, Jun Yang, and Cong Yu. 2017. Computational Fact Checking through Query Perturbations. *ACM Trans. Database Syst.* 42, 1 (2017).

[17] Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. 2020. LayoutLM: Pre-Training of Text and Layout for Document Image Understanding. In *KDD '20*. 11921200.

[18] Xu Zhong, Jianbin Tang, and Antonio Jimeno Yepes. 2019. PubLayNet: largest dataset ever for document layout analysis. In *ICDAR*. IEEE, 1015–1022.