Efficient Crawling for Scalable Web Data Acquisition

29th International Conference on Extending Database Technology (EDBT 2026)

Antoine Gauquier[1] Ioana Manolescu[2] Pierre Senellart[1]

[1]DI ENS, ENS, CNRS, PSL & Inria
[2]Inria & Institut Polytechnique de Paris

27 March 2026

Context (1/3)

Ever increasing volume of content of many forms on the Web:

- Corpora of text and video $\longrightarrow$ Training of ML or DL models
- Structured HTML (especially containing tables) $\longrightarrow$ Knowledge Bases
- Open-access statistic dataset and studies $\longrightarrow$ Analyzing how societies function and doing fact-checking.
- ...

First step for building applications based on Web-published data is to find those data files.

Context (2/3)

Data files come in a wide variety of formats:

- Text and video $\longrightarrow$ `.doc`, `.txt`, `.mp4`, `.flv`...
- Structured HTML $\longrightarrow$ `.html`
- Statistic dataset and studies $\longrightarrow$ `.pdf`, `.xls`, `.csv`...

They are published by different providers:

- *National agencies*: INSEE (France), Labor/Census Bureaus (US), ABS (AU)
- *International bodies*: Eurostat, IMF, WB, UN Agencies $\longrightarrow$ some provide APIs or portals
- *Smaller entities*: Trade unions, NGOs $\longrightarrow$ publish highly-specialized data

Insee Mesurer pour comprendre

United States Census Bureau

International Monetary Fund

World Health Organization

Open Knowledge Foundation

Context (3/3)

Websites can be huge with only a small fraction of published data files

Exhaustively crawling those websites takes a lot of space and most importantly time:

- Taken by the crawl of a lot of "uninteresting" resources
- Crawling ethics requires to wait (typically $\sim$ 1 sec.) between two HTTP requests $\longrightarrow$ For a website of 1M pages, it would take roughly 11 days of waiting...

Generic crawler for any given website (selected beforehand) $\longrightarrow$ we cannot make any assumption about the website's structure/content.

Problem Statement

Problem: Given the starting URL of a website, we want to retrieve as many targets as possible, while minimizing the consumption of resources. We define:

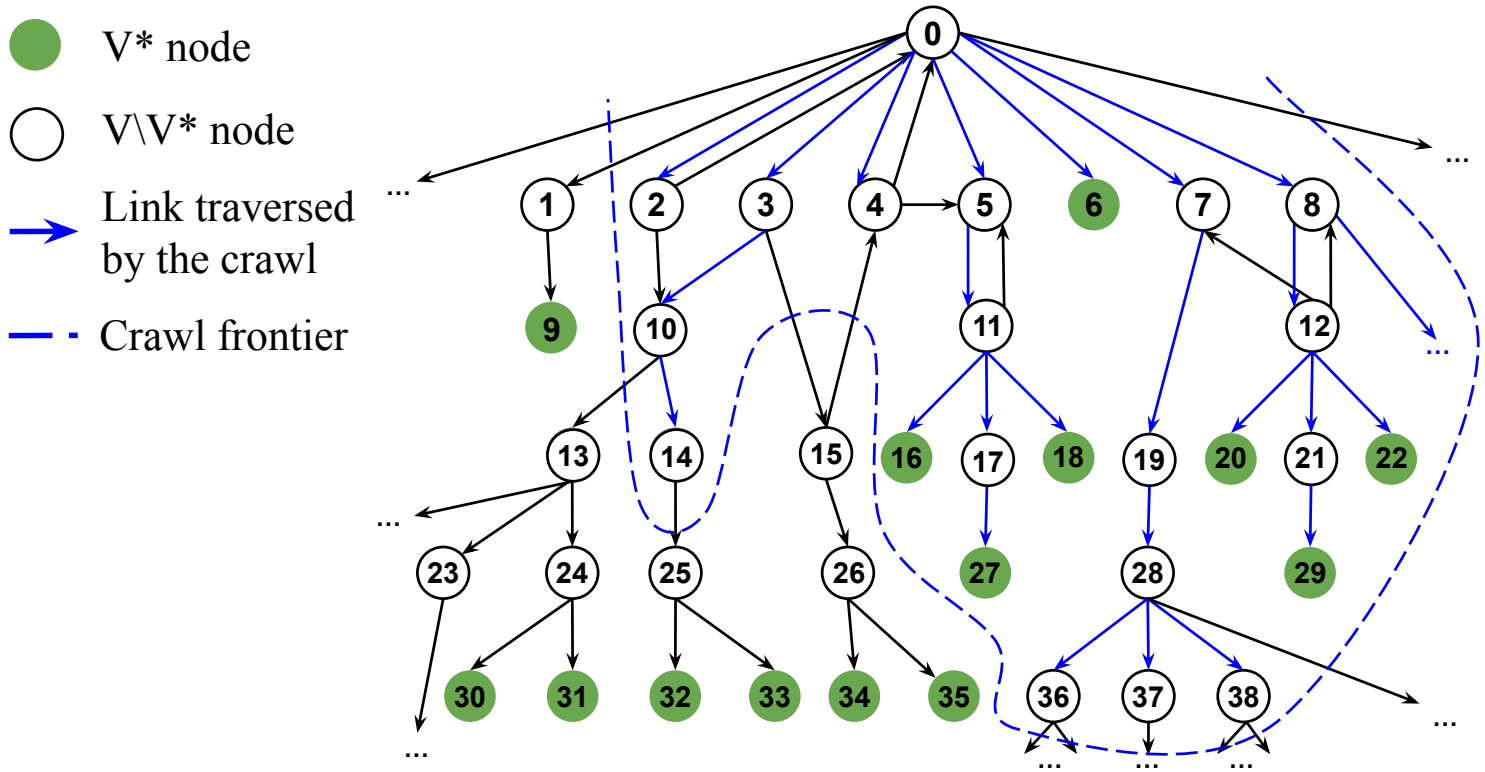
- A target: file uploaded on the website, satisfying some constraints specified by the user (e.g., be a data file: CSV, Excel, PDF...).
- Resources: number of HTTP requests sent to the server, and data volume exchanged with the server.

We want to retrieve the maximum of targets while minimizing the effort to do so.

Formalization: Graph Crawling Problem (1/2)

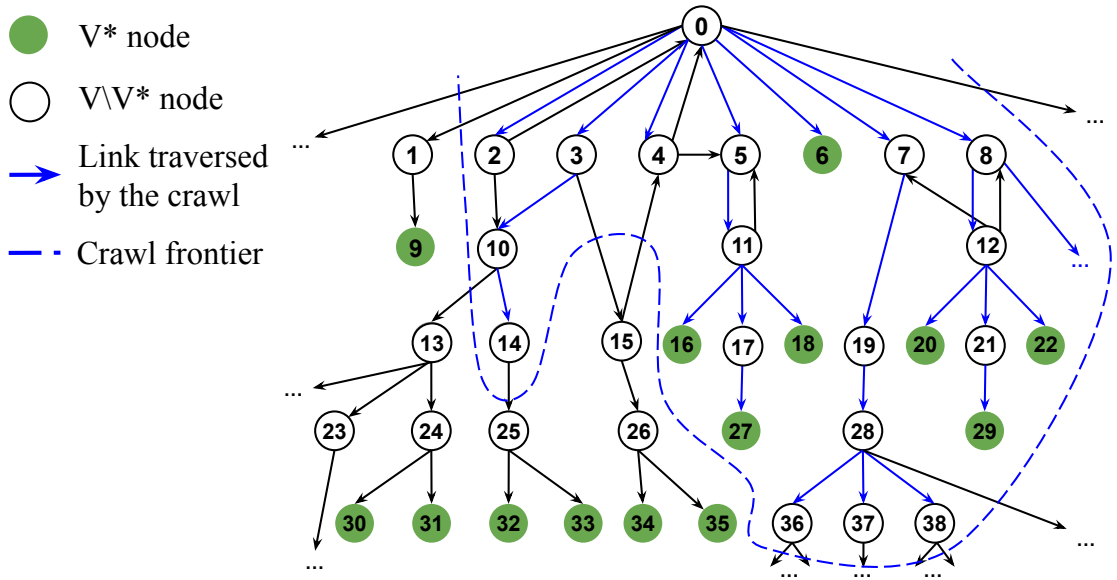
Website graph: $G = (V, E, r, \omega, \lambda)$, with:

- V a finite set of *vertices* (webpages)
- $E \subseteq V^2$ a set of *edges* (hyperlinks)
- $r \in V$ the *root* of the graph (input webpage; 0 here)
- $\omega : V \to \mathbb{R}^+$ a *cost* function (cost of retrieving that page)
- ($\lambda : E \to \mathcal{L}$ a *labeling* function).



Formalization: Graph Crawling Problem (2/2)

Given a website graph $G = (V, E, r, \omega, \lambda)$ and a subset V^* of targets of V, the graph crawling problem is to find a crawl $T = (V', E')$ of G with $V^* \subseteq V'$, of minimal total cost.



This problem turns out to be intractable, as it it NP-hard (proof via reduction from set cover problem).

This is why we need a heuristic yet effective solution to solve our crawling problem: an Efficient Crawler for Scalable Web Data Acquisition.

Existing Web Crawlers (1/3)

What are existing Web crawlers from the literature? Among most simple and classic methods:

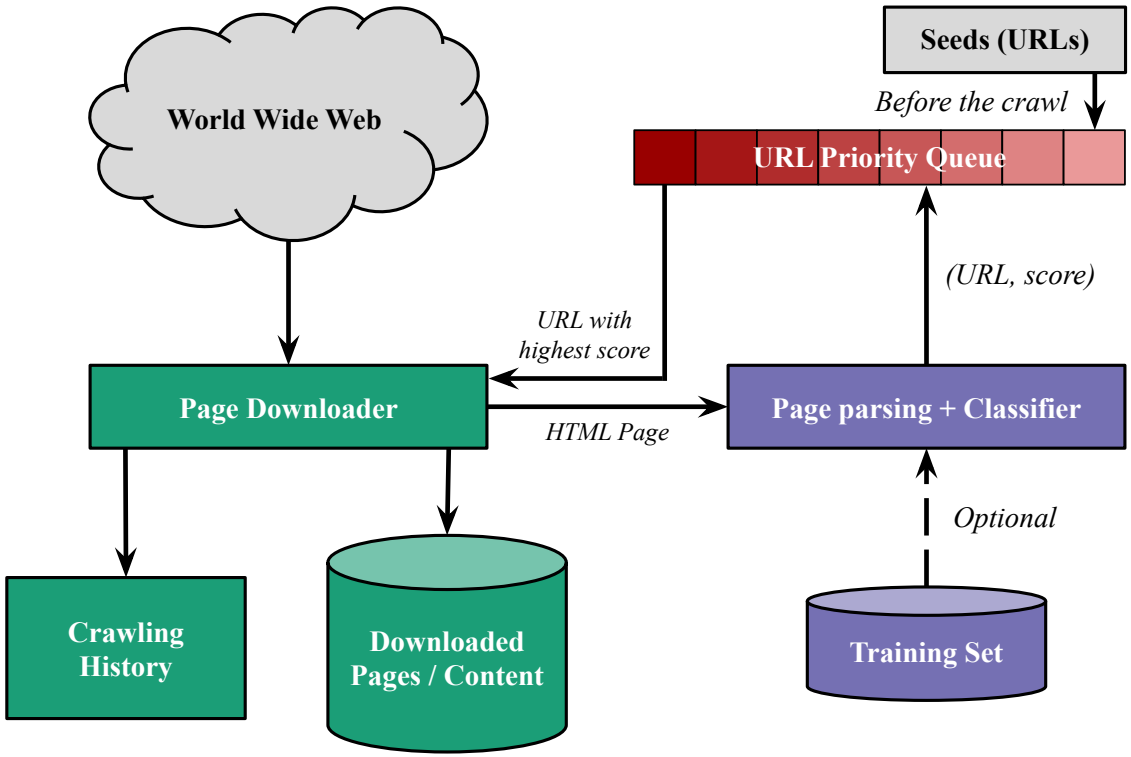
- Random Crawler: selects the next hyperlink to visit uniformly at random from the frontier. Most naive technique.
- Breadth-First Search (BFS): frontier as a First-In-First-Out queue data structure. Efficient when targets are not too deep.
- Depth-First Search (DFS): frontier in a Last-In-First-Out stack.

None of them is effective without prior knowledge of the website's structure, which we don't have

Existing Web Crawlers (2/3)

Focused Crawlers:

- Prioritize some pages during a crawl, following a focus (an objective).
- Usually, a classifier assesses the likelihood that an hyperlink leads to a page part of the focus.
- Frontier: a *priority-queue*.



Existing Web Crawlers (3/3)

Two kind of focused crawlers:

- Topical: focus is a predefined topic $\longrightarrow$ keywords or sample documents. Most common ones.
- Non-Topical: rarer; focus is something else, e.g.:
  - ANTHELION [MMB14]: semantic annotations in Web pages
  - ACEBOT [FS15]: as much diverse textual content as possible

In our setting: topical crawlers are ill-suited

- We want targets about any possible subject covered by the website
- Independently, defining a topic $\Longrightarrow$ choosing one (or several) languages

Best approach: non-topical focused crawlers

Research Hypotheses

Our approach $\longrightarrow$ two main research hypotheses:

- Hyperlinks similarly structured in the HTML page were they were found lead to similar content.
- It is possible to learn which hyperlinks are most likely to lead to targets, given their link structure.

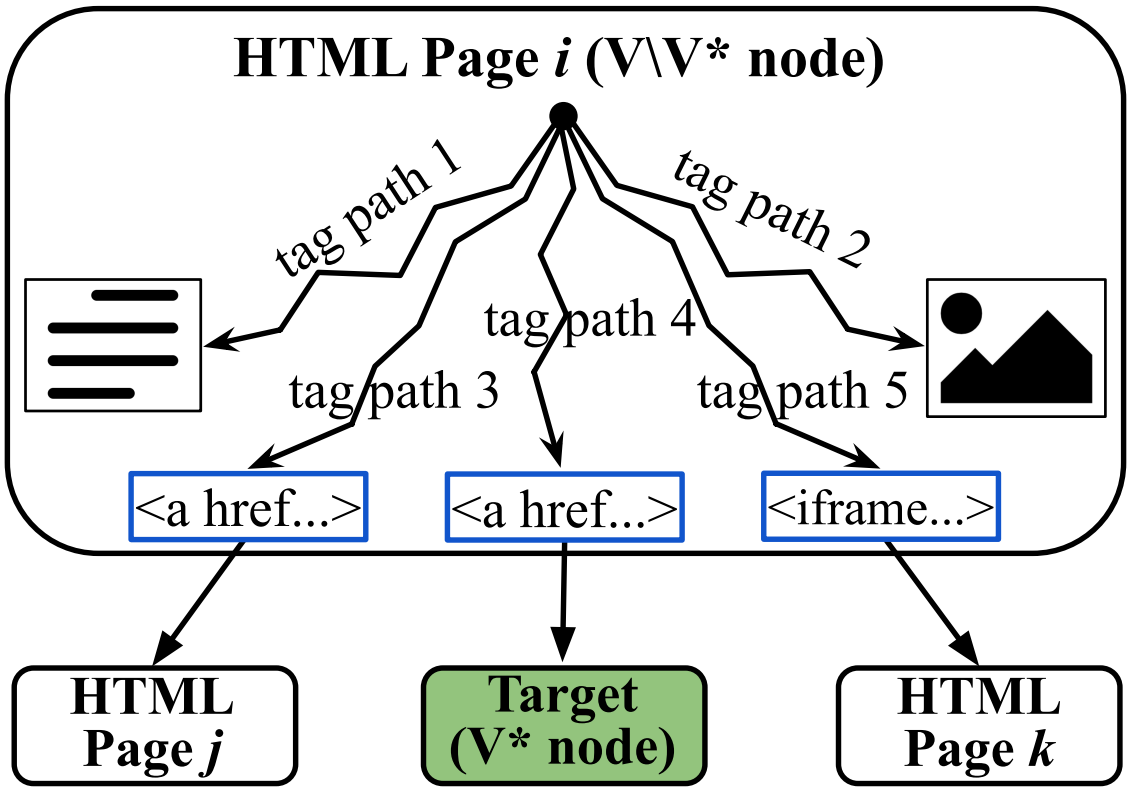
Idea: We can separate, for a given website, parts of it that are rich in targets, from ones that are not:

- without prior knowledge of the website's structure/content
- just assuming existence of some structure

Building Groups of Hyperlinks

How to do it: We represent each encountered hyperlink with its associated tag path (its link structure).

They are computed from the HTML page's Document Object Model:



Building Groups of Hyperlinks

We form groups of hyperlinks with a merging strategy based on similarity between the tag paths. . .

. . . to separate the fruitful groups from the less interesting ones, as we assume that similar hyperlinks will lead to similar kind of content.

https://opm.justice.gouv.fr/content/notice-dépôt-dune-candidature-au-concours-de-greffier-de-tribunal-de-commerce-annexe-1-0

/html/body.html front not-logged-in one-sidebar sidebar-first page-node/div.jumbotron opm-content-container/div.container/div.row/div.col-sm-3 well-resp/div.block block-block last odd#block-block-7/p/a

https://www.justice.gouv.fr/lettre-direction-affaires-civiles-du-sceau

/html/body.html front not-logged-in one-sidebar sidebar-first page-node/div.jumbotron opm-content-container/div.container/div.row/div.col-sm-3 well-resp/div.block block-block first odd#block-block-6/p/a

https://opm.justice.gouv.fr/content/notice-de-renseignements-sur-laccès-à-la-profession-de-greffier-de-tribunal-de-commerce-et

/html/body.html front not-logged-in one-sidebar sidebar-first page-node/div.jumbotron opm-content-container/div.container/div.row/div.col-sm-3 well-resp/div.block block-block last odd#block-block-7/p/a

https://www.justice.gouv.fr/grands-dossiers/justice-amiable

/html/body.path-frontpage page-node-type-homepage/ [...] /div.fr-col-md-6fr-col-12/article.fr-enlarge-link fr-card fr-card--detail-sm fr-card--horizontal/ div.fr-card__body/div.fr-card__content/h3.fr-card__title fr-title/a.fr-card__link

https://www.justice.gouv.fr/prise-charge-victimes-dinfractions-penales

/html/body.path-frontpage page-node-type-homepage/ [...] /article.fr-enlarge-link fr-card fr-card--detail-sm fr-card--horizontal/div.fr-card__body/ div.fr- card__content/h3.fr-card__titlefr-title/ a.fr-card__link

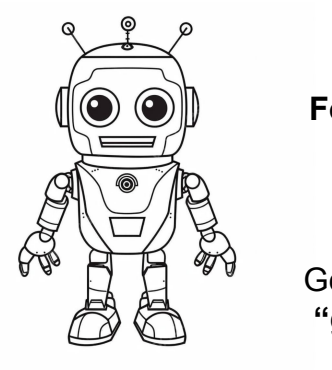
https://www.justice.gouv.fr/plan-daction-justice

/html/body.path-frontpage page-node-type-homepage/ [...] /div.fr-col-md-6 fr-col-12/article.fr-enlarge-link fr-card fr-card--detail-sm fr-card--horizontal/ div.fr-card__body/div.fr-card__content/h3.fr-card__title fr-title/a.fr-card__link

Learning Fruitful Groups: Reinforcement Learning

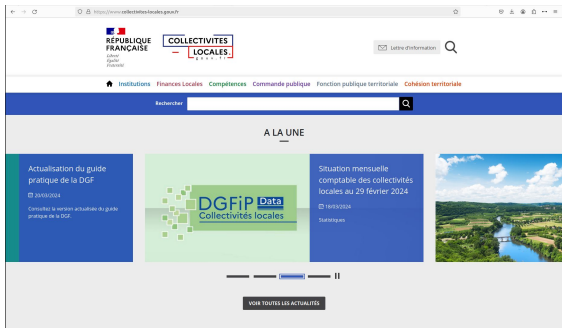
How can we dynamically separate fruitful groups from other ones? Reinforcement Learning (RL).

Leveraging Multi-Armed Bandits strategy [ACBF02], and especially Sleeping Multi-Armed Bandits [KNMS10].



Select a group
Follow a hyperlink from said group

Get a **reward**: a score describing **how "good" was this choice**, to improve future decisions …



How many **targets** can we reach **from this page**?

Reward: Number of reachable, unobserved targets from the HTML page.

Exploration–Exploitation Trade-Off

At each crawling step, the choice of the group is a trade-off between:

- Exploitation of groups so far identified as fruitful
- Exploration of few times selected groups, that might hide fruitful hyperlinks.

Doing so works well... But requires to tell, just given a hyperlink's URL, if its a target or not:

- `https://www.assemblee-nationale.fr/14/pdf/ta/ta0686.pdf` $\longrightarrow$ Easy
- `https://www.education.gouv.fr/media/90068/download` $\longrightarrow$ Manageable
- `https://www.justice.gouv.fr/en/node/9961` $\longrightarrow$ Requires learning...

URL Classifier

Naive way: HTTP HEAD requests over the URL to get Content-Type from the header. But too costly.

Idea: Actively train a URL classifier, taking as input an URL, outputting one of the two following classes: target or HTML.

"on–the–fly" training allows the classifier to adapt to its environment and to eventual drastic changes in it.

It takes advantage of URLs that are automatically labeled when following a hyperlink during the crawl.

Dataset: Selected Websites and Characteristics (1/2)

	Starting URL	Multilingual	Fully Crawled
ab	`https://www.abs.gov.au/`	✗	✗
as	`https://www.assemblee-nationale.fr/`	✗	✗
be	`https://www.bea.gov/`	✗	✓
ce	`https://www.census.gov/`	✗	✓
cl	`https://www.collectivites-locales.gouv.fr/`	✗	✓
cn	`https://www.cnis.fr/`	✗	✓
ed	`https://www.education.gouv.fr/`	✗	✓
il	`https://www.ilo.org/`	✓	✗
in	`https://www.interieur.gouv.fr/`	✗	✓
is	`https://www.insee.fr/`	✓	✓
jp	`https://www.soumu.go.jp/`	✓	✗
ju	`https://www.justice.gouv.fr/`	✗	✓
nc	`https://nces.ed.gov/`	✗	✓
oe	`https://www.oecd.org/`	✓	✓
ok	`https://okfn.org/`	✓	✓
qa	`https://www.psa.gov.qa/ar/Pages/default.aspx`	✓	✓
wh	`https://www.who.int/`	✓	✗
wo	`https://www.worldbank.org/`	✓	✗

Dataset: Selected Websites and Characteristics (2/2)

	#Available (k)	#Target (k)	HTML with Tar. (%)	Target Depth
ab	952.26	263.26	8.86	8.94 (± 2.56)
as	994.36	93.56	4.34	4.47 (± 4.23)
be	31.23	15.84	32.19	5.73 (± 3.21)
ce	988.37	257.68	3.47	4.23 (± 0.48)
cl	5.54	3.70	5.40	2.80 (± 0.82)
cn	12.80	7.49	13.87	4.26 (± 1.59)
ed	102.71	10.47	3.95	11.89 (± 13.22)
il	993.92	26.56	2.53	9.63 (± 9.83)
in	922.46	22.98	1.54	66.94 (± 39.43)
is	285.55	168.88	41.34	5.20 (± 1.81)
jp	998.26	256.34	6.30	21.37 (± 49.34)
ju	56.61	14.85	4.85	86.91 (± 86.30)
nc	309.97	84.94	18.87	3.63 (± 1.66)
oe	222.58	45.04	15.61	6.28 (± 5.65)
ok	423.12	12.95	0.74	2.64 (± 2.89)
qa	4.36	2.45	4.15	3.03 (± 0.61)
wh	351.86	55.59	14.19	4.43 (± 0.62)
wo	223.67	23.10	2.38	4.52 (± 0.69)

Crawlers (1/2)

1. SB-Classifier: Our approach with URL classifier
2. SB-Oracle: Our approach with an oracle (perfect, costless URL classifier)
3. Random: Random crawler
4. BFS: Breadth-First Search
5. DFS: Depth-First Search
6. Focused: A standard focused crawler (focus = targets), without topical features:
   - approx. depth of the hyperlink
   - 2-gram representation of the hyperlink's URL
   - 2-gram representation of the hyperlink's anchor

   $\longrightarrow$ ablated version of our approach: neither tag path structure information, nor RL

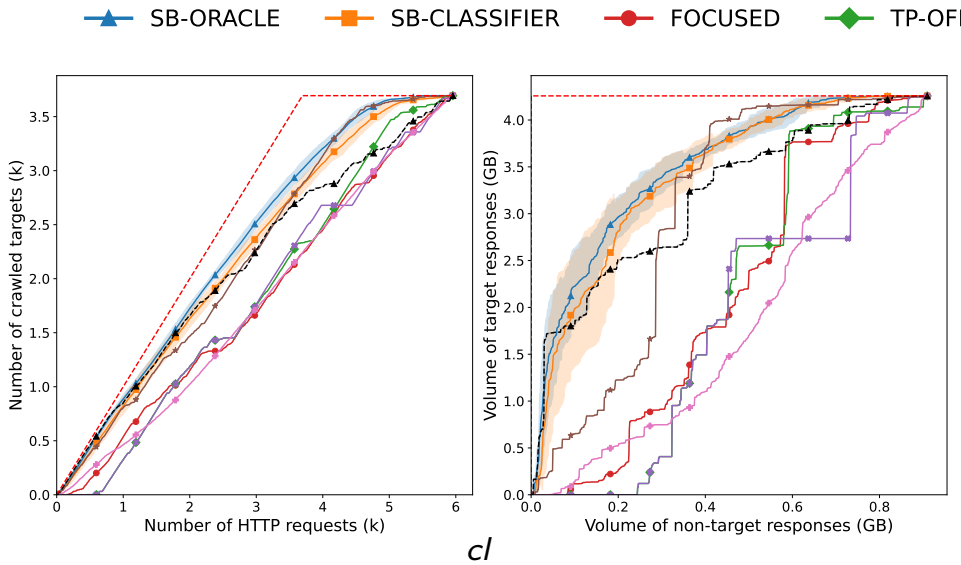
Crawlers (2/2)

7. TP-Off: Offline-Trained, Tag Path-Based Crawler; an adaptation of ACEBot [FS15] to target retrieval
   - Crawls 3k first pages with BFS & learns tag paths groups on this subset $\longrightarrow$ requires to compute benefit of each HTML page: we provide it at no cost with an oracle (unfair advantage)
   - Then matches hyperlinks to existing tag path groups (ordered by likelihood), or given a score of 0 if too distant

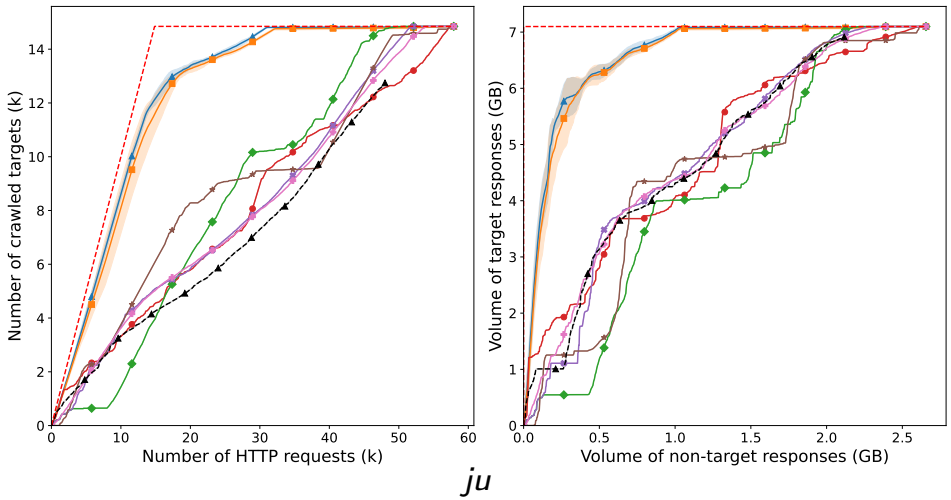
   $\longrightarrow$ ablated version of our approach: learning offline instead of online (closest to our approach)

8. TRES [KKP+21]: Topical, RL-Based Focused Crawler; using a Bi-LSTM classifier to assess topic relevance
   - requires keywords and sample pages to define the topic: we provide it with 74 terms about the topic "target retrieval" at no cost
   - requires positive examples of fruitful HTML pages: we provide it with 1k such pages, giving partial access to the solution at no cost
   - can only handle URLs to HTML pages in its frontier: we give access to a perfect URL classifier (oracle) at no cost

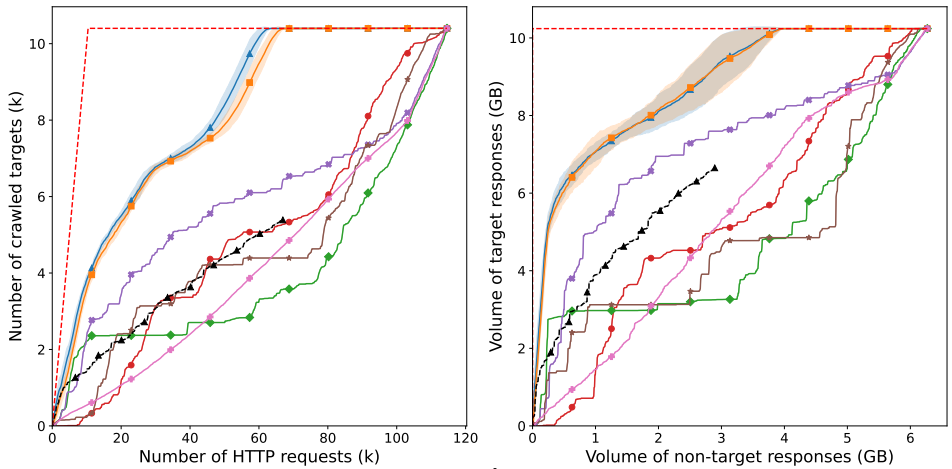
Results (Overview, 1/3)



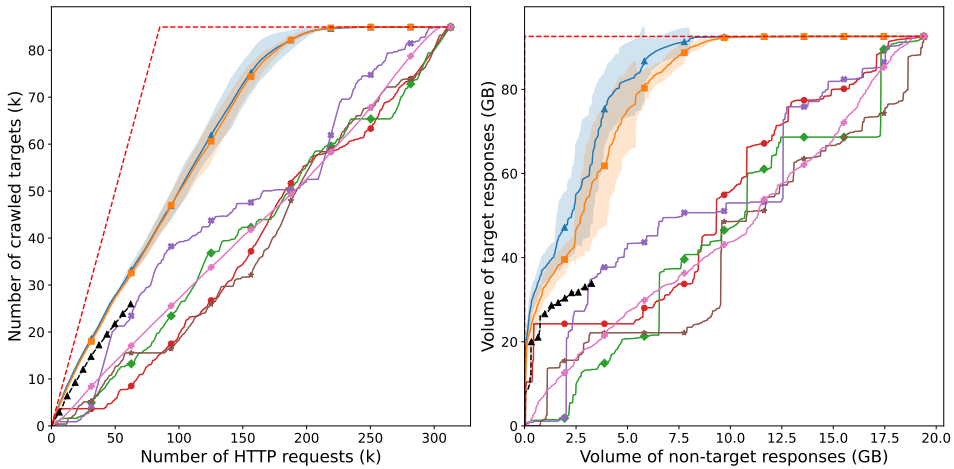
cl



ju

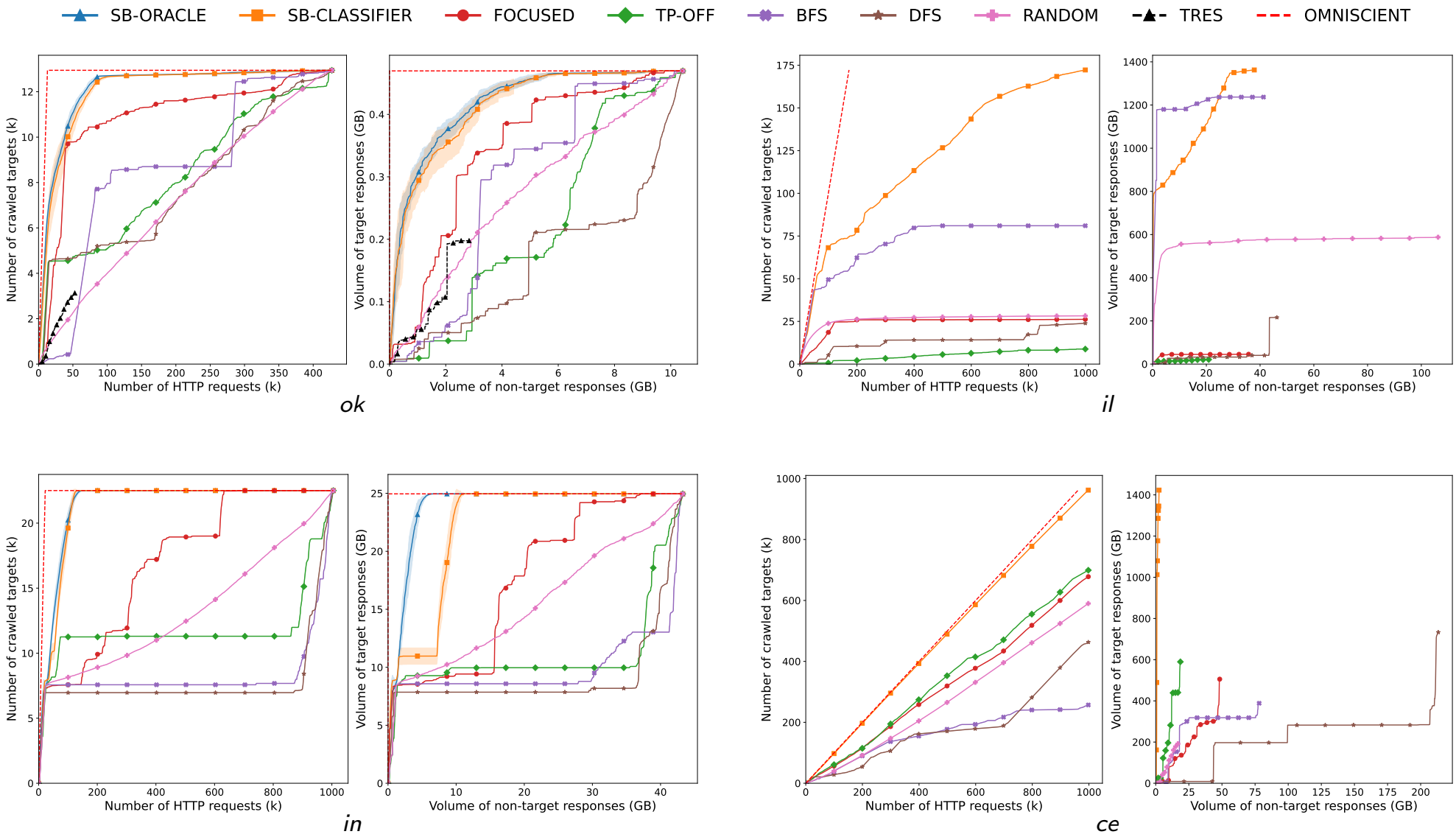


ed

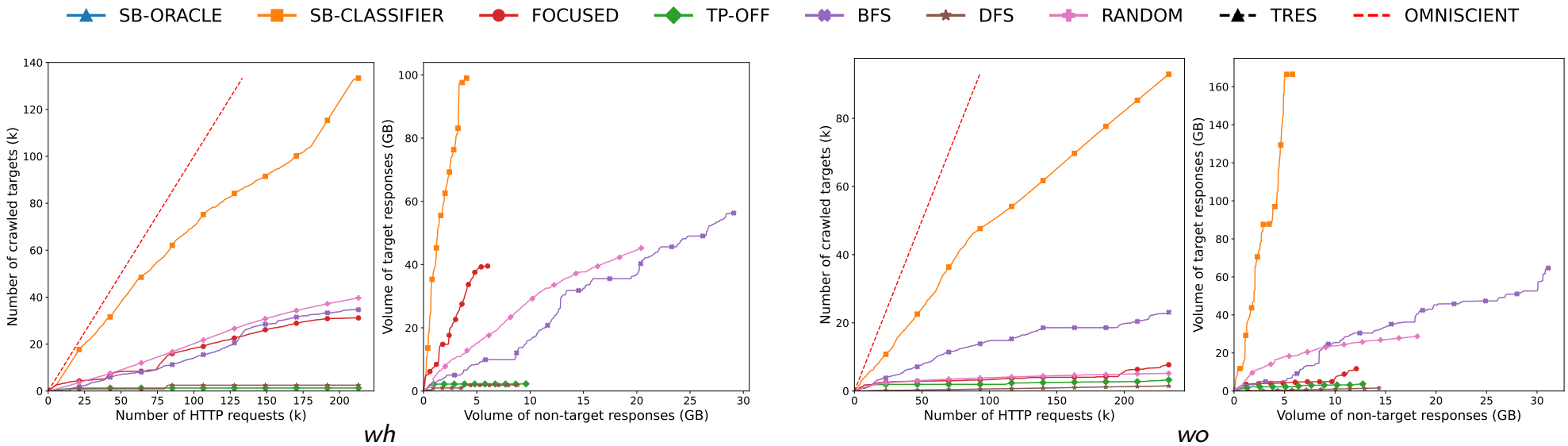


nc

Results (Overview, 2/3)



Results (Overview, 3/3)



Takeaways

- Most applications are based on Web-published data: finding them is not easy. Exhaustive methods are too costly.
- Finding an optimal crawl of a website is intractable $\longrightarrow$ need of a heuristic technique.
- We presented a novel crawling technique based on reinforcement learning to dynamically learn interesting parts of the Website, that are rich in targets.
- We equip our crawler with a URL classifier that is dynamically trained with automatically-generated data, giving the class (HTML or target) of an URL at no extra cost.
- The crawler outperforms standard baselines and three adapted from the literature, sometimes reaching more than 90% of targets within less than 10% of the webpages.

Thank you for your attention!

Any questions?

Code and dataset information

https://github.com/AntoineGauquier/efficient_crawler_for_scalable_web_data_acquisition/

Bibliography I

Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer.

Finite–time Analysis of the Multiarmed Bandit Problem.

Machine Learning, 47(2):235–256, 2002.

Muhammad Faheem and Pierre Senellart.

Adaptive web crawling through structure-based link classification.

In Robert B. Allen, Jane Hunter, and Marcia Lei Zeng, editors, *Digital Libraries: Providing Quality Information - 17th International Conference on Asia-Pacific Digital Libraries, ICADL 2015, Seoul, Korea, December 9-12, 2015, Proceedings*, volume 9469 of *Lecture Notes in Computer Science*, pages 39–51. Springer, 2015.

Andreas Kontogiannis, Dimitrios Kelesis, Vasilis Pollatos, Georgios Paliouras, and George Giannakopoulos.

Tree-based focused web crawling with reinforcement learning.

CoRR, abs/2112.07620, 2021.

Bibliography II

Robert Kleinberg, Alexandru Niculescu-Mizil, and Yogeshwer Sharma.

Regret Bounds for Sleeping Experts and Bandits.

Machine Learning, 80(2):245–272, 2010.

Robert Meusel, Peter Mika, and Roi Blanco.

Focused crawling for structured data.

In Jianzhong Li, Xiaoyang Sean Wang, Minos N. Garofalakis, Ian Soboroff, Torsten Suel, and Min Wang, editors, *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, CIKM 2014, Shanghai, China, November 3-7, 2014*, pages 1039–1048. ACM, 2014.

Search Engines (1/3)

Why don't you just Google it?

Examples based on Google services (conducted early 2025):

1. Classical Google Search Engine (and its Programmable Search Engine[1]), by filtering with `site:` and `filetype:`
   - For `site:https://www.justice.gouv.fr/`
     - on `filetype:pdf` $\longrightarrow$ 302 results, while $>$ 9000 exist
     - on `filetype:tsv` $\longrightarrow$ filetype is not even recognized by GS
   - For `https://www.ilo.org/`
     - on `filetype:pdf` $\longrightarrow$ 641 results, while $>$ 41k exist
     - on `filetype:zip` $\longrightarrow$ 0 result, while $>$ 2200 exist

[1] `https://programmablesearchengine.google.com/`

Search Engines (2/3)

2. Google Dataset Search[2]: aims at target search, but...
   - For `site:https://www.justice.gouv.fr/` $\longrightarrow$ 109 tabular data files, out of $>$ 1100
   - For `https://www.ilo.org/` $\longrightarrow$ 93 datasets, out of $>$ 170k
   - For `https://www.census.gov/` $\longrightarrow$ 312 datasets, out of $>$ 800k

3. Google Public Data Explorer: aims at public data exploration, but...
   - build for human readers $\longrightarrow$ only allows manual exploration
   - ranges over a closed set of providers, missing huge ones (US and UN agencies for instance)

[2] `https://datasetsearch.research.google.com/`

Search Engines (3/3)

- Do not fully index any given website, and anyway limit available results to 1000.
- Don't offer any transparency or control over their process.

We devise a method that retrieves targeted data files within a budget, with hopefully fully explainable decisions.

URL Classifier Design

Evaluation metrics

- Intra-website: % of requests needed to reach 90% of targets
- Inter-website: Misclassification Rate (MR)

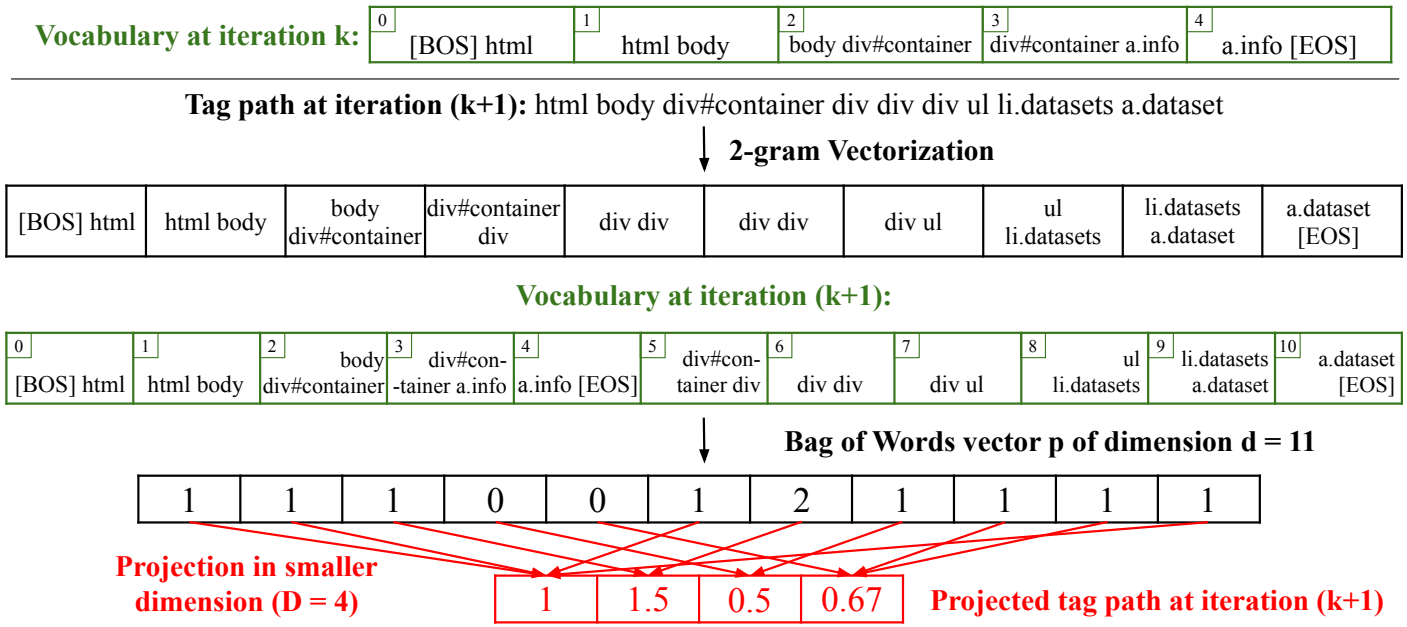
Feature sets

- **URL_ONLY:** URL features only
- **URL_CONT:** URL + anchor text + DOM path + surrounding text

Models LR: Linear Regression SVM: Support Vector Machine NB: Naive Bayes PA: Passive–Aggressive

Variant	*be*	*cl*	*cn*	*ed*	*in*	*is*	*ju*	*nc*	*oe*	*ok*	*qa*	MR
URL_ONLY-LR	82.1	75.1	71.3	53.2	11.7	76.1	36.5	52.6	60.7	15.9	62.3	2.62
URL_ONLY-SVM	82.7	75.7	71.8	63.6	11.3	76.0	37.4	52.2	63.5	16.7	61.5	2.99
URL_ONLY-NB	82.9	75.2	72.1	53.7	11.4	76.3	35.8	52.7	59.7	18.0	63.1	2.92
URL_ONLY-PA	82.3	74.4	71.7	53.3	11.1	75.8	36.7	51.6	60.5	15.9	60.9	**2.56**
URL_CONT-LR	82.2	74.4	71.9	54.3	11.3	76.4	37.8	52.9	64.7	16.8	**60.0**	5.93
URL_CONT-SVM	82.6	75.0	71.8	52.8	11.6	76.4	38.8	53.1	61.1	18.7	**60.1**	6.36
URL_CONT-NB	84.1	74.7	71.9	53.6	11.4	75.7	35.5	52.3	59.9	19.1	60.4	7.15
URL_CONT-PA	82.5	75.1	71.9	53.6	11.6	76.2	38.4	52.1	62.6	16.1	60.6	4.12

Building Groups of Hyperlinks – Merging strategy



Merging strategy: all groups are represented with a centroid. For each tag path:

1. Compute its set of (approx.) k-NN, and an associated similarity score
2. If the top-score $> \theta$ (threshold): merge and update the centroid
3. Otherwise: create a new group

URL Classifier Results (Overview)

Confusion matrix of our URL classifier, on the fully-crawled websites (average over 15 runs)

True \ Predicted	HTML (%)	Target (%)	Neither (%)
HTML	58.04	1.37	0.00
Target	0.75	32.19	0.00
Neither	5.34	2.41	0.00

SD Retrieval Across Sample Targets

Experimental setup

- Random sample of $7 \times 40 = 280$ targets
- 7 diverse websites

	be	*ed*	*is*	*in*	*nc*	*oe*	*wh*
SD Yield (%)	82	35	93	40	83	60	40
Mean # SDs / Target	9.1	2.8	2.9	2.1	2.1	4.9	1.4

- SDs are present in a substantial fraction of targets across all websites
- Even non-statistical websites show notable SD presence
- SDs are concentrated: when present, typically > 2 SDs per target