

# Sign-of-life crawler

---

This paper describes the approach used by the crawler. Last updated on 23/04/2020.

## Agenda

I -PARKING CATEGORIES.....	2
II -MAIN CRAWLER.....	4
1 ) Python Visit.....	5
2 ) Browser Visit.....	5
3 ) Information Extraction.....	5
4 ) Registrar Identification.....	6
5 ) Redirection Classification.....	7
6 ) Text Classification.....	7
7 ) Dynamic Content Identification.....	7
8 ) Consolidation.....	8
III -TEXT CLASSIFICATION.....	9
1) Parking vocabulary collection.....	9
2) Feature Engineering.....	10
3) Dataset Labelling.....	11
4) Model Training.....	11
IV- PIPELINE PERFORMANCE.....	12
V- SOCIAL MEDIA ACTIVITY.....	13
VI- MAIL SERVER IDENTIFICATION.....	18
VII- LIMITS AND IMPROVEMENT DIRECTIONS.....	19
1) Complex dynamic websites.....	19
2) Dataset size.....	19
3) Texts in images.....	19

## I -PARKING CATEGORIES

Given a top-level domain, our objective is to identify how much of it is used in a meaningful way, at domain name granularity. In particular, we want to classify a domain name in one of the following categories, from lower to higher granularity (level 1 to 4).

category_lv1	category_lv2	category_lv3	category_lv4	Comment
content	high content	high content	High content	= Meaningful website
	low content	Parked Notice Registrar	Parked Notice Registrar	= Registrar default pages
		Blocked	Blocked	= Note on censorship or suspension of website
		Upcoming	Under construction	
			Starter	
		Abandoned	Expired	= Note on expiration of domain
		Not used	Index of	= Specific page: Cf. Figure 1 = with/without unformatted content
			Blank Page	Empty page
			For sale	
			Reserved	
			Parked Notice Individual Content	
			Parked Notice Individual Content	
no content	errors	DNS Error	No address found	= Domain not linked to a server
		Connection Error	Refused Connection	
			Timeout	
		Invalid Response	No Status Code	
		HTTP Error	HTTP_401	= Client or Server errors identified via HTTP status code
			HTTP_403	
			HTTP_404	
			HTTP_408	
			HTTP_500	
			HTTP_502	
			HTTP_504	
			HTTP_other	

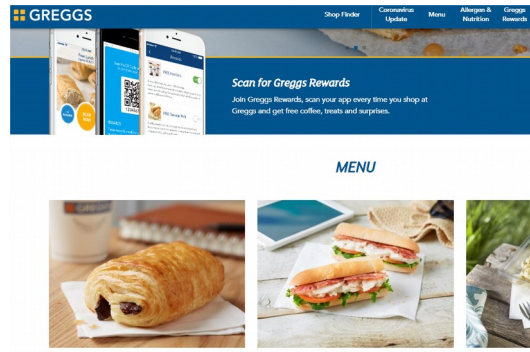


Figure 1 - example of "High content" page

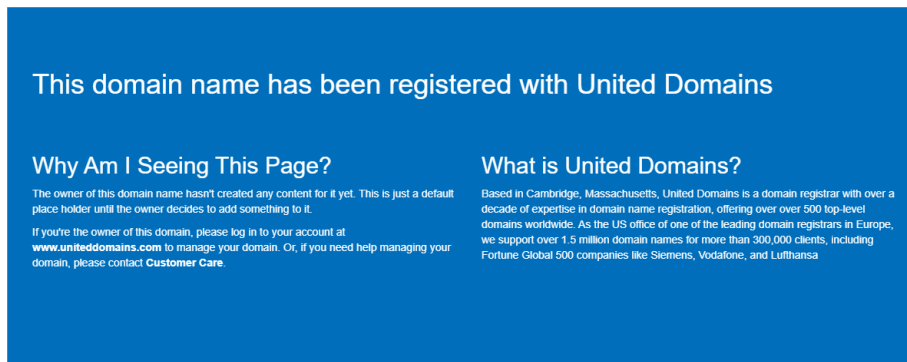


Figure 2 - example of "Parked Notice Registrar" page



Figure 3 - example of " Under construction" page

Index of /			
Name	Last Modified	Size	Type
Parent Directory/		-	Directory
<a href="#">0linux/</a>	2010-Jun-17 11:18:51	-	Directory
<a href="#">0x109/</a>	2013-Sep-19 17:10:28	-	Directory
<a href="#">2mandvd/</a>	2013-Jul-30 16:28:18	-	Directory
<a href="#">3lrvsteam/</a>	2009-Aug-17 15:53:58	-	Directory
<a href="#">3v1deb/</a>	2018-Oct-05 02:32:28	-	Directory

Figure 4 - example of "Index Of" page



This site can't be reached

discordapp.com's server IP address could not be found.

DNS\_PROBE\_FINISHED\_NXDOMAIN

Reload

Figure 5 - example of "No address found" page

## II -MAIN CRAWLER

To reach this goal, a fully automatic pipeline has been set up with Python. Its general architecture is described in the following picture.

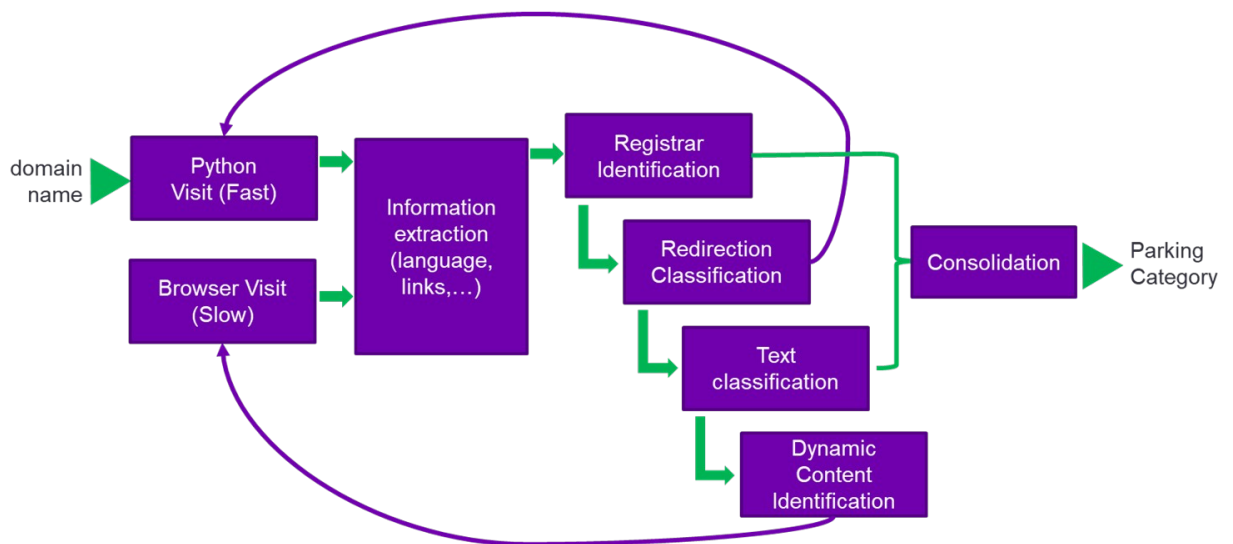


Figure 6 - General architecture of the crawler

In general:

- The domain is visited with Python to collect the HTML page
- Information is extracted from that first visit
- Registrar links are searched, if one is found, the page is classified as "Parked Notice Registrar"
- If a redirection is detected, we repeat from the first step with the redirection target link (the results of the target link classification are used for the initial domain)
- The displayed text is analyzed to identify parking notes (ex: "website under construction")

- If the page hasn't been classified as parked so far, the page is analyzed to identify clues of dynamic content generation.
- If the page needs to be interpreted, the domain is visited with Chrome browser, controlled by Python. When the full page is rendered, we repeat the steps from information extraction to text classification

The following sections describe each step in more details.

## **1 ) Python Visit**

For a given TLD:

- 50,000 domain names are collected from its zonefile into a csv file. Based on samples analysis, 50,000 URLs will yield a confidence interval below 1 % range.
- Each URL is visited using the following format of full URL: http:// + URL (without www.).
  - o Python, like many other general purpose codes and like all browsers (Chrome, Firefox...), has HTTP client libraries. So far, "aiohttp" is used. This library allows communicating on the internet by sending messages following the HTTP or HTTPS protocols. In particular, the standard GET method will yield the content of the main page of a website, which is an HTML page. However, the html code is not interpreted to maintain a high speed of visits.

## **2 ) Browser Visit**

In some domains, the only way to have access to the content of the page is to interpret the code. When interpreting a code, each instruction is implemented, whether it is a resource collection (image, video, library, 3<sup>rd</sup> party services), a content generation (text, charts...) or page formatting. To do that, we use Chrome through its webdriver. To adapt instruction from Python to the webdriver, the library "selenium" is used. This approach is equivalent to opening Chrome with your mouse and type the url in the address bar (except with regard to scarcely implemented bot detectors). As output we get the final HTML page with its rendered displayed content.

Even though multiple pages are visited at the same time through multiprocessing, this approach is very slow and must be used only when necessary.

### 3 ) Information Extraction

At this stage, we possess the main HTML page as well as the HTTP communication messages in a raw format. Various information is extracted with the library BeautifulSoup for the following stages:

- A) Redirection elements: Iframes, framesets, meta tag with attribute http-equiv=refresh and window.location object.
- B) Links: Extracted from “a” and “area” tags as well as from redirection objects. Cleaned and converted to absolute urls.
- C) Page Complexity indicators: Size of formatted HTML, quantity of non-trivial HTML tags, quantity of displayed words/letters.
- D) Displayed text: All the visible text is extracted from titles, divs, spans, paragraphs, titles of links, as well as the non-visible ALT attribute of images and “meta name=description” tag ... Formatting objects (ex: bold tag “<b>”) are removed
- E) Language: From the displayed text, we identify the main language of the website preceded by a filtering of acronyms, codes and numbers
  - o A simple natural language processing (NLP) library, called “langdetect” developed and open sourced by Google (Apache License 2.0), allows identifying the language of the displayed text. The recent progress in NLP provides access to more advanced approaches for this task.
- F) Non-text: collection of all attributes and tags as well as inline Javascript

All this information is reused for the downstream stages.

### 4 ) Registrar Identification

If a link to one and only one registrar website is found, the website might be a registrar. However some registrars also provide website building services (ex: Wordpress) and other registrars have a diversified business portfolio (like Microsoft). To filter out the first category, a restriction on the quantity of non-trivial non-repetitive tags is applied. Generated websites tend to have a complex HTML structure. For the second category, a website that displays a lot of other contents must also have a parking note (cf text classification) to be considered in this category. A domain that satisfies these conditions is classified as **Parking Response Registrar**. The registrar that are considered are listed in the Table-2.

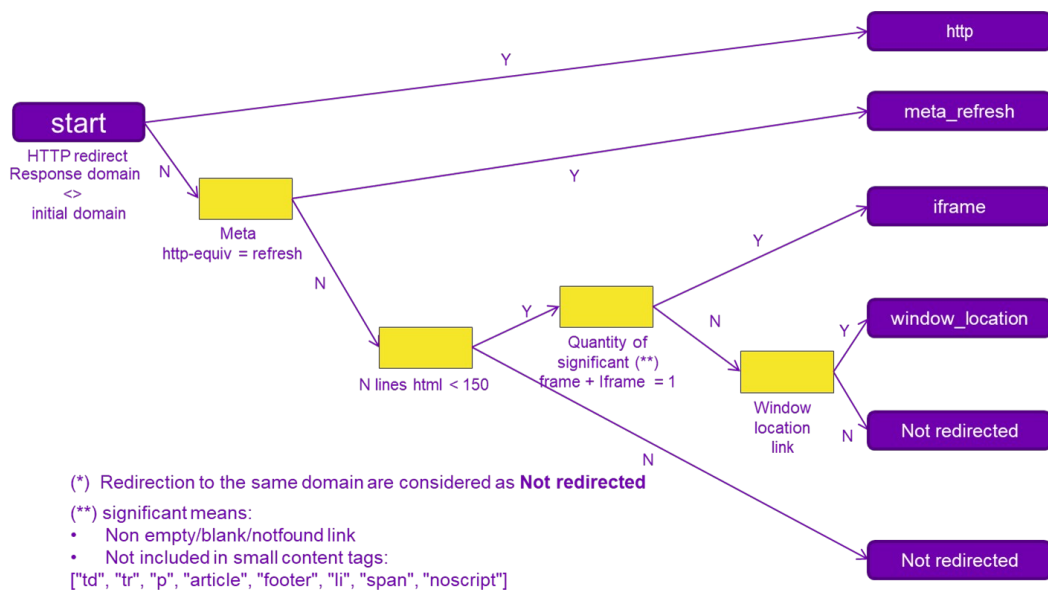
**Table 1 - List of detected registrars/hosting companies**

Registrars
ICANN members
CENTR members
Russia
New Zealand
Estonia
Montenegro

In addition, some registrars have specific parking libraries/links (=Non-text). They usually use the keyword “park” in the naming of their variable. If such keyword is detected in a small page, the domain is considered **Parking Response Registrar**.

## **5 ) Redirection Classification**

Following the redirection tree in the picture below, if there is only one significant meta-refresh AND/OR frameset and frame OR iframe OR window.location(.href), the HTML page contains less than 150 lines and the link leads to another domain. Here “significant” means the link is not empty/blank.html/notfound and the element is not included in basic content tags like P, TR, TL, SPAN... □ the redirection target page is visited and used for classification. The redirection flag is raised.



**Figure 7 - Redirection tree**

## 6 ) Text Classification

In short, a machine learning algorithm classify if the displayed text is a parking note or not. It takes as input the displayed text and a reference vocabulary with parking connotations. This step is described in more detail in the section III.

## 7 ) Dynamic Content Identification

When there is little displayed text, not enough for a relevant text classification, it might mean that some text is generated dynamically. A visit by browser is done in three cases:

- o Displayed text size below a threshold AND high ratio of inline Javascript compared to HTML size (= length of Javascript code / length of all HTML)
- o Displayed text size below a threshold AND detection of dynamic content creation functions like "document.write", ".createElement", ".appendChild"
- o Displayed text size below a threshold AND detection of Javascript required note (usually in "noscript" tag)

If one of the conditions is met, the results compiled so far are all ignored and recomputed with the browser-rendered page.



## 8 ) Consolidation

Finally, a consolidation step is gathering all the outputs to infer the final categories, starting with errors then content page.

### *a) Errors*

- o If the HTTP Client fail to identify the IP address of a URL, the URL is categorized as **No address found**.
- o If the IP address is found but the corresponding server AND/OR the client computer refuse the connection, the URL is categorized as **Refused Connection**.
- o If the IP address is found but the corresponding server is taking too long to answer, the URL is categorized as **Timeout**.
- o If the HTTP communication doesn't provide a status code, the URL is categorized as **No Status Code**
- o If the server provides a response with no-content AND a non-200 status code, the URL is classified based on the status code among:

Status Code	Description
401	Unauthorized
403	Forbidden
404	Not Found
408	Request Timeout
500	Internal Server Error
502	Bad Gateway
504	Gateway Timeout

Other Status codes are gathered in the category **HTTP\_other**.

### *b) Page with content*

In the following order:

- o If a registrar page is identified, the domain is classified as **Parked Notice Registrar**
- o If the text classification step predicts a parking note, the domain is classified according to the specific vocabulary identified in the page:

Cateogory LV4	Example of specific vocabulary
Blocked	Censored, Blocked, suspended...
Under construction	Construction, Maintenance...
Starter	Wordpress, Wix...
Expired	Expired, Deleted
Index of	Index of
For sale	Purchase, for sale, acquire...
Reserved	reserved, owned...
Parked Notice Individual Content	None of the above

- o If the displayed text has less than 5 words and has no JavaScript, no frames and no links, the URL is considered as **Blank Page**.

### III -TEXT CLASSIFICATION

In order to train a parking note classifier, the following steps are implemented:

- o Identify relevant indicators of parking note. Parking is associated to a list vocabulary that needs to be collected.
- o A classifier only takes numeric values as input. To get the most relevant numeric values called “features”, a feature engineering step is designed.
- o A ground truth dataset is manually compiled.
- o With the above data, the model can be trained to learn the relationship between the features and the parking note ground truth.

#### 1) Parking vocabulary collection

There is a limited vocabulary that is associated with parking pages. With the gathered experience, 66 “root” english words have been identified like “park”, “sale”, “register”... . The detailed list is in the file taxonomy.csv, at column root.

A same “root” word can appear in different forms due to :

- o Plurality: website □ websites
- o Tense: park □ parked
- o POS tag: reserve (verb) □ reservation (noun)

To expend the list to all forms, we use the library word\_forms.

Finally, a same word can be found in different languages, for example “domain” is “域名” in Chinese and “домен” in Russian. To account for all possible languages:

- o Each form of each root is translated to 103 languages using Google Translate model.
- o These translations are sometimes joined with prepositions. Prepositions are examples of “stopwords” (= recurrent word of a language with basic meaning). To remove them, we trim leading and trailing stopwords from the tokenized translation (=translation text converted into a list of words).

The final list of words is provided in taxonomy.csv. Whenever a word from that list is found on a website, it is a strong indicator of parking but not always. The goal of the machine learning is to figure out when it is the case.

## 2) Feature Engineering

We provide different numeric values to the model:

- o The count of each root words: if a page only contains “park”, “parking” and “website”, the model will be provided the following table. The relevance of these features is straightforward.

park	website	other roots		
2	1	0	0	...

- o Count of root words “pair” in the same and successive sentences:

The root words are split into three categories:

1. Core words: They are words that implicitly refers to the website/page/domain like “website”, “content”, “space”, “page”, “account”, ...  
The domain of a given website “abc.com” is one of those keyword: “abc.com” is replaced by “x\_url”
2. Attribute words: They are words that implicitly indicates a lack of content like “parked”, “blocked”, “coming”, “unavailable”, “construction”...
3. The other words that doesn’t fit in the first two categories like “hello\_world”, “index\_of”, “lorem ipsum” ...

The vast majority of parking notes are a combination of one core word and one attribute word in the same sentence or successive sentences like “parked page” or “Welcome to abc.com. Purchase now”. To provide this information to the model, the text is parsed into sentences, then into words. Sentences are looped over. In each sentence we identify the core and attribute words and count the pairs. When a pair

is identified, the size of the smallest sentence is also provided to the model as a way to filter pairs that occurs by chance.

For example, the text “abc.com. Purchase now. We accept cookies. this page is reserved.” will become:

n_p air_ sam e	n_pair _succ essive	min_se nt_size _same	min_sent _size_suc cessive
1	1	3	1

For information, splitting a text into sentences then words is depending on the language. English have sentence separators (=points) as well as word separators (=spaces). But other languages like Chinese, Korean, Japanese or Indian languages have a more complex rationale. To achieve our goal, various NLP libraries are used to adapt to each language.

- o Quantity of sentences and words: Often parking pages are short. In addition, the bigger the website, the more likely parking words and pairs are to be found. The size of the page will give the model a chance to filter out false positives.

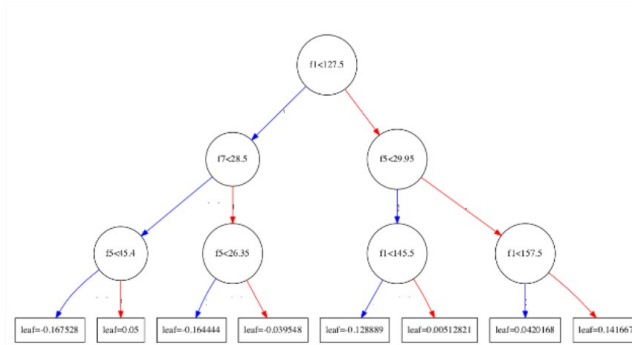
In total, 72 features are provided to the model (=66+ 4 + 2)

### 3) Dataset Labelling

For now, 2000 websites with WHOIS location from all around the world have been opened manually. Among them, 1420 have yielded a page with non-registrar content. Among these, 472 are identified as parked with a note. Additional data would help, especially with more rare features (“censored page”, “domain not linked to a website” ...).

### 4) Model Training

A tree based XGBoost model has been used. It is a model ensemble of decision trees. One decision tree example is represented below. This tree split the dataset into subsets with higher purity with respect to parking class. It tries to form groups of either only non-parked page or only parked page. One split is always done with one feature and one threshold for that feature (for example, domains with more than 100 words vs domains with less than that). The choice of feature and threshold is mathematically optimized to minimize the prediction error. XGBoost resort to multiple decision trees, trained sequentially, where one tree focus on the error of all the former trees together.



**Figure 8 - Example of decision tree where " $f_i$ " is the feature  $i$**

On the dataset, in cross validation, we get the following performance:

Accuracy	92.3 %		Confusion Matrix		PREDICTED	
Precision	88.1 %				not parked	parked
Recall	89.0 %		ACTUAL	not parked	892	57
f1 score	88.5 %			parked	52	420

This trained model is integrated to the crawler pipeline.

## IV- PIPELINE PERFORMANCE

WIP

## V- SOCIAL MEDIA ACTIVITY

### Objective

The objective of this section is to identify how a given domain is leveraging social media like (ex: Facebook, Twitter). It can be used in three ways:

- The domain has its own official social media pages. This is identified through the presence of backlinks inside the HTML page, for example in the form of an icon band like the figure below.



So far, we have been focusing on western media:

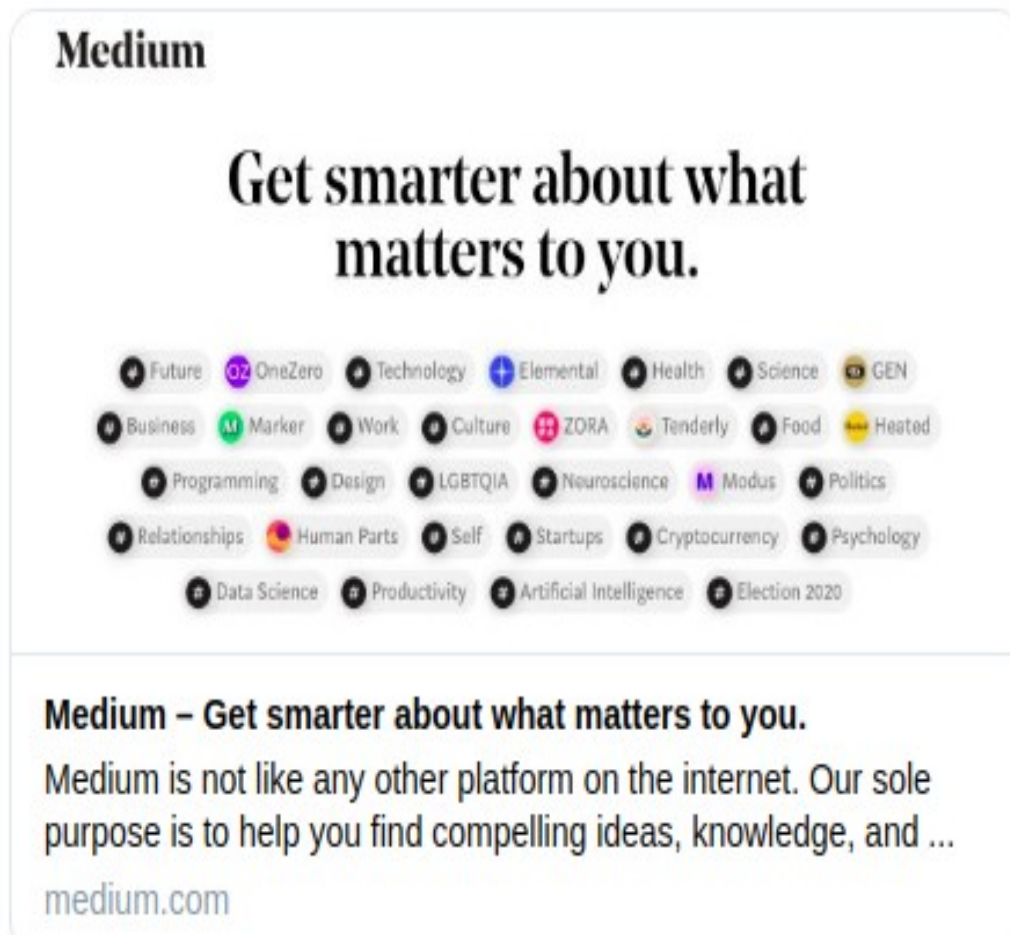
- Twitter
- Facebook
- Instagram
- LinkedIn
- Reddit
- Github

Plenty of others may be implemented in the future, among them other western medias (Youtube, Whatsapp, Messenger, Tumblr, Viber, Snapchat, Pinterest, Medium ...) as well as eastern media (Wechat, QQ, Qzone, Tik Tok, Sina Weibo, Baidu Tieba, Line, Telegram ...) with hundreds of millions of registered users.

- The websites has adapted its content for social media display. Websites can resort to specific HTML tags to optimize display/sharing on Media. Among them:
  - Twitter Cards:

```
<meta data-rh="true" name="twitter:title" content="Medium - Get smarter about what matters to you."/>
<meta data-rh="true" name="twitter:app:name:iphone" content="Medium"/>
<meta data-rh="true" name="twitter:app:id:iphone" content="828256236"/>
<meta data-rh="true" name="twitter:card" content="summary_large_image"/>
<meta data-rh="true" name="twitter:creator" content="@Medium"/>
<meta data-rh="true" name="twitter:description"
  content="Medium is not like any other platform on the internet. Our sole purpose is to
  help you find compelling ideas, knowledge, and perspectives. We don't serve ads-we serve
  you, the curious reader who loves to learn new things. Medium is home to thousands of
  independent voices, and we combine humans and technology to find the best reading for
  you-and filter out the rest."/>
<meta data-rh="true" name="twitter:image:src"
  content="https://cdn-images-1.medium.com/max/1200/1*29XAq2WrtEjUCxRzSgDLXA.png"/>
<meta data-rh="true" name="twitter:site" content="@Medium"/>
```

*Code snippet of twitter cards: indicating the key information to display in social media*



*The same website as displayed on twitter whenever the link medium.com is used*

- Open Graph: Similar to Twitter cards, but developed and open-sourced by Facebook, it helps social media understand the content of the page.

```
<meta data-rh="true" property="og:url" content="https://www.nytimes.com"/>
<meta data-rh="true" property="og:type" content="website"/>
<meta data-rh="true" property="og:title" content="Breaking News, World News & Multimedia"/>
<meta data-rh="true" property="og:description"
  content="The New York Times: Find breaking news, multimedia, reviews & opinion on Washington,
  business, sports, movies, travel, books, jobs, education, real estate, cars &
  more at nytimes.com."/>
<meta data-rh="true" property="og:image"
  content="https://static01.nyt.com/newsgraphics/images/icons/defaultPromoCrop.png"/>
```

*Code snippet with key information: url, title, description...*

 **The New York Times**

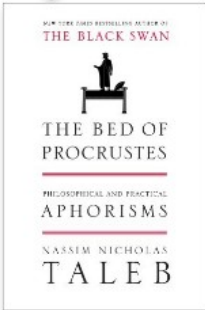
[Breaking News, World News & Multimedia](#)

The New York Times: Find breaking news, multimedia, reviews & opinion on Washington, business, sports, movies, travel, books, jobs, education, real estate, cars & more at [nytimes.com](https://www.nytimes.com). (15 ko) ▾



*Link, as displayed on social media (Facebook)*

- Schema Tags: As a result of a collaboration between major search engine (Google, Bing, Yandex, and Yahoo!), schema tags helps identify key attributes of a product/service, for an enhanced display in search engine results.



**Want to Read** ▾

Rate this book  
★★★★★

## The Bed of Procrustes: Philosophical and Practical Aphorisms

(Incerto #3)

by Nassim Nicholas Taleb (Goodreads Author)

★★★★★ 3.76 · [Rating details](#) · 5,451 ratings · 512 reviews

**The Bed of Procrustes** is a standalone book in Nassim Nicholas Talebs landmark Incerto series, an investigation of opacity, luck, uncertainty, probability, human error, risk, and decision-making in a world we dont understand. The other books in the series are *Fooled by Randomness*, *The Black Swan*, and *Antifragile*.

By the author of the modern classic *The Black Swan*, this ...[more](#)

**GET A COPY**

[Amazon UK](#) [Online Stores ▾](#) [Libraries](#)

Hardcover, 128 pages  
Published November 30th 2010 by Random House

[More Details...](#) [Edit Details](#)

*The page as displayed by the website: formatting and content are a priori complex to identify...*



```

<h1 id="book" class="gr-h1 gr-h1--serif" itemprop="name">
  The Bed of Procrustes: Philosophical and Practical Aphorisms
</h1>
<span itemprop="author" itemscope="" itemtype="http://schema.org/Person">
<div class="authorName__container">
<a class="authorName" itemprop="url" href="https://www.goodreads.com/author/show/21559.Nassim_Nicholas_Taleb"><span
  itemprop="name">Nassim Nicholas Taleb</span></a> <span class="greyText">(Goodreads Author)</span>
</div>
</span>
<a rel="nofollow" itemprop="image" href="/book/photo/9402297-the-bed-of-procrustes">
</a>

```

*... but schema tags, clearly identifies the characteristics of the book with "itemprop"...*

www.goodreads.com > book > show > 9402297-the-be... ▼

## The Bed of Procrustes: Philosophical and ... - Goodreads

The **Bed of Procrustes** is a standalone book in Nassim Nicholas Talebs landmark Incerto series, an investigation of opacity, luck, uncertainty, probability, human error, risk, and decision-making in a world we don't understand. The other books in the series are Fooled by Randomness, The Black Swan, and Antifragile.

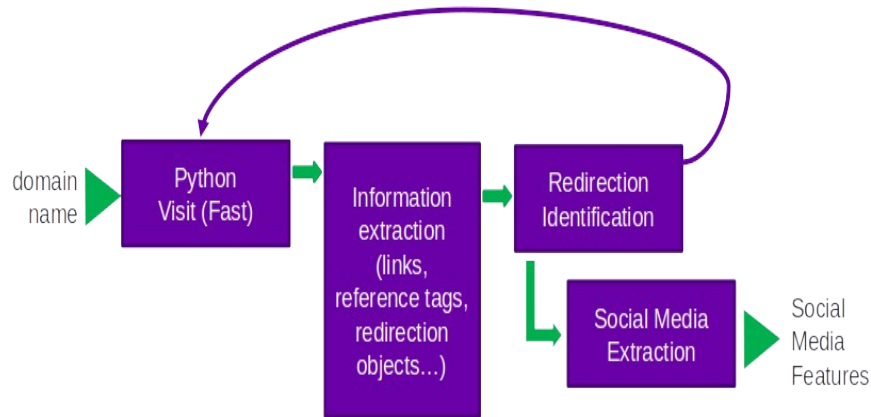
★★★★★ Rating: 3.8 - 5,451 votes

*...the result on Google select and displays the most relevant information for the user (title, rating description)*

- Finally, websites can resort to a sign-on service with Google, Facebook and LinkedIn. With that service, the website can use the platform security of a big tech company as well as requesting for additional information from the social media accounts (eg: List of friends on Facebook...). In return, the social media company gets to know when a given user is connected to the website and adapt its services accordingly (eg: advertisement). **NOT IMPLEMENTED YET.**

## Detailed approach

To detect those features, the following architecture is implemented.



*General architecture of the social media pipeline*

The fast visit and redirection identification are reused directly from the parking crawler. The information extraction is adapted to feed the new social media extraction bloc.

Additional information is extracted:

- All links to social media
- Tag information of links: type of tag (img, a ...) and ancestors line in the HTML tree
- Twitter Cards, Open Graph and Schema tags

The downstream identification of social media is rather straightforward except for one challenge. There might be noisy links to any social media, which doesn't refer to the official account of the domain owner. To make sure we get them, the following logic is applied:

- We filter out libraries (eg. facebook.com/share.php), trivial page identifiers (eg. twitter.com/tr?q=abc") and normalize the remaining links (lowercase, dash and underscore removed..)
- If multiple links are found for a single social media (occurs rarely), a custom logic is implemented:
  - We select the name that matches the domain name

- If it doesn't match, we select the name that is included or includes the domain name
- If it doesn't, the name with the smallest Levenshtein distance to the domain name is selected (= number of letters to add/change/remove to go from the domain to the link name)

## VI- MAIL SERVER IDENTIFICATION

### Objective

If a domain doesn't return a website, it doesn't necessarily mean it is not used. It can also be associated to a mail server. Our objective is to know which one does.

A mail is sent through a protocol called SMTP (Simple Mail Transfer Protocol), different than HTTP(S) used in the previous section. When sending a mail at name@domain.com, this protocol starts with a query to the DNS server of .COM. The DNS returns the MX record of "domain.com" (MX= Mail eXchange). This MX records contains the list of domains of mail exchange servers. These domains can have the same domain as the initial mail (eg, mail0.domain.com) or an external one (eg. mail1.yahoo.com for servers hosted at Yahoo, mail2.live.com for Microsoft). From there, the mail content can be sent to the mail server that will attribute it to the correct final address according.

Two key information can be extracted for a given input domain:

- Whether a domain has a mail service attached: yes if it has an MX record.
- Whether it manages its own mail server: yes if the mail servers have the same domain as the input domain

### Approach

To get this information, a different but simpler pipeline is implemented. Each domain is queried for MX records using dnspython library. The first mail address in the priority list is selected to compare with the input domain.

## **VII- LIMITS AND IMPROVEMENT DIRECTIONS**

### **1) Complex dynamic websites**

Some parked pages generate parking notes using non-explicit libraries (ex: 1234.js) hidden in one or more level of libraries (i.e. the main page calling a library, that calls another library, which writes the parking note). The generated text is not hard-coded even in the browser-rendered page.

### **2) Dataset size**

The current machine learning for text classification could benefit from a more diversified dataset, especially for rare parking notes.

### **3) Texts in images**

Texts that are saved as pixels in an image are currently not considered. For this case, we rely on the text surrounding the image as well as ALT parameter to classify the page. An OCR extension could be developed to extract the text of websites with only one image.