ESSnet Trusted Smart Statistics – Web Intelligence Network

Grant Agreement Number: 101035829 — 2020-PL-SmartStat

Joint report for Work Package 2 (Online Based Enterprise Characteristics) and Work Package 3, Use Case 5 (Business register quality enhancement)

Report: URL finding methodology

Draft (ver. 5.0), 2022-01-31

Prepared by:

1. **Heidi Kühnemann (HSL, Germany, Heidi.kuehnemann@statistik.hessen.de)**
2. Arnout van Delden (CBS, Netherlands)
3. Donato Summa (Istat, Italy)
4. Johannes Gussenbauer (STAT, Austria)
5. Alexandra Ils (Destatis, Germany)
6. Katja Löytynoja (Statistics Finland)

Web Intelligence Network

Funded by the European Union

This document was funded by the European Union.

The content of this deliverable represents the views of the author only and is his/her sole responsibility. The European Commission does not accept any responsibility for use that may be made of the information it contains.

Web Intelligence Network

Funded by the European Union

Table of contents

1 Introduction

Enterprise websites are a promising new data source for official statistics. Both ESSnet Big Data projects showed how enterprise websites can be used to extract enterprise characteristics like social media links, e-commerce functionalities, online job vacancies, and more (Stateva et al. 2020). Therefore, scraped enterprise websites can supplement the Information and Communication Technologies (ICT) survey, which struggles with low response rates, missing data and other error sources that typically occur in surveys with voluntary participation. Work package (WP) 2 of the Web Intelligence Network (WIN) aims to bring the ICT use cases to production. Additionally, enterprise websites can serve as a data source for the enhancement of the business register, which is the goal of Work Package 2, Use Case (UC) 5 of the WIN. One can use website texts to classify economic activity of enterprises. In that way, possibly wrong economic activity codes in the business register can be automatically identified and corrected. Additionally, one could extract email addresses and telephone numbers of enterprises from their websites, since some countries do not have these contact details in their business registers yet.

Web Intelligence Network
Funded by the European Union

For these applications, the enterprise website has to be known beforehand. Enterprise websites can be identified through several means. Many statistical institutes already have (some) enterprise URLs obtained through register information, data purchases or survey responses[1]. This is not the case in all countries, though. Even if most countries do have URLs for some enterprises, the information is usually missing for a large proportion of enterprises in the statistical business register (SBR). Automated methods for finding enterprise websites on a large scale are therefore necessary.

A relatively simple approach is to use available email addresses of enterprises (explained in Ch. 9.2.). Often, the domain of the email address is also the enterprise domain. However, this approach can only be applied if email addresses are available. This report focuses on a different approach, which sends automated requests to a search engine and uses the results to identify the correct enterprise URLs. Usually, this URL finding approach contains four parts:

1. sending search terms to a search engine,
2. scraping the result URLs,
3. extracting the relevant information from the scraped data and
4. creating a machine learning or rule-based model to link websites to enterprises.

This report aims to provide guidelines for finding enterprise URLs within the European Statistical System (ESS). The report is a collaboration between WP 2 and WP 3 UC 5, who both need enterprise URLs as a prerequisite to derive information from this data source.

The report is structured as follows. At first, we explain how enterprise websites are defined in the ICT survey and which consequences this has for URL finding (Ch. 2). Subsequently, Ch. 3 discusses the creation of training data as important prerequisite for URL finding. Chapter 4-7 describe different steps in URL finding: searching the enterprises in search engines (Ch. 4), scraping found pages (Ch. 5), extracting features from the obtained data (Ch. 6) and classifying which websites belong to the enterprises (Ch. 7). Chapter 8 focuses on how to exclude irrelevant results from the search results. In addition, alternative methods and applications for URL finding (Ch. 9) as well as legal issues (Ch. 10) are discussed. Finally, we draw a conclusion in Chapter 11.

2 Defining enterprises and enterprise websites

The SBR is the data source for enterprises in URL finding. Within the EU, enterprises are defined as follows:

The enterprise is the smallest combination of legal units that is an organizational unit producing goods or services, which benefits from a certain degree of autonomy in decision-making, especially for the allocation of its current resources. An enterprise carries out one or more activities at one or more locations. An enterprise may be a sole legal unit.[2]

The ICT survey also uses this concept of the enterprise. Aside from enterprises, the SBR contains legal units that can be directly linked to tax and administrative registers. Since enterprises in the EU are legally required to list their tax/register numbers on their website, and tax/register numbers in general link to legal units, the legal unit is often the foremost entity of interest for URL finding. To derive websites of the

[1] As an example, some NSIs were able to ask for the enterprise URL in the ICT survey.

[2] https://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31993R0696:EN:HTML

Web Intelligence Network

Funded by the European Union

enterprise, as defined above, one should then define rules about how to aggregate websites from legal units to websites of enterprises. Depending on the use case and country, the unit of interest for URL finding is either legal units or enterprises. The remainder of this report uses “enterprise” as term for the unit of interest in URL finding, but is applicable to legal units as well.

According to the methodological manual of the ICT survey, enterprise websites are websites that enterprises use to present their businesses. The enterprise does not have to be the owner of the website/domain, it can also be a third party website. An example is the website of a group of enterprises to which it belongs or the website of the parent enterprise. Retail enterprises that have a presence on the web through the enterprise that holds the rights (royalties in the case of franchising) for the brand are counted as having a website. Examples for this would be the parent enterprise Bosch and the associated enterprise is a retailer of Bosch garden tools or Levis jeans and its retail shops. The complexity of the web presence can vary from simply an ‘enterprise locator’ on a map (in particular for retailers — e.g. find your closest Bosch tools dealer) and indirect advertisement of the products/services to more sophisticated functionalities. As long as the presence of such enterprises on the web is through other websites for the specific products or services (not through yellow pages or telephone directories) it is considered that the enterprise has a website. The enterprise should have some control over the content. However, it does not include any presence of the enterprise on the web. That would be too broad, as it would include the presence of the enterprise (e.g. its name or its contact information) in directories and online yellow pages.

Enterprise websites can be any type of website, independent of its sophistication or the services provided. However, the ICT survey definition does not consider social media accounts as an enterprise website.

Enterprises on e-marketplaces where they have the possibility to advertise themselves, quote prices for ad hoc services etc. are no enterprise websites either. Note that an enterprise may have e-commerce web sales and still not have a website as the sales are through e-marketplaces.

According to this definition, an enterprise can have zero, one or several websites. The goal of URL finding is to find at least one website of the enterprise if it has one. It is also valuable information to identify that an enterprise does not have a website. Enterprises with several websites pose a problem though. Sometimes, an enterprise might still have only one main website that contains the information of interest. In the best case, the other websites are not well maintained and rank much lower in search results – so they do not show up among the best search results for a specific enterprise. Consequently, for most purposes, in this case only the main website is of interest to us – and appears as one of the top search results. In other cases, an enterprise has several websites for different purposes. As an example, one website contains general information about the enterprise and the other is an online shop. It would also be possible that an enterprise has different websites for different economic activities, different products and services, etc. In these cases, all of the different websites are of interest.

The URL finders developed in ESSnet Big Data I & II have limited their results to only the main URL for each enterprise. There are several reasons for this. At the one hand, URL finding needs training data of enterprises with known URLs. Usually, existing URL data sets are used that contain at most one URL per enterprise, making it very difficult or impossible to train a machine learning model that predicts several URLs per enterprise. Additionally, URL finding is often restricted to finding domains that the enterprise of interest owns. This reduces the complexity of URL finding and of deriving enterprise characteristics from the website data. If only domains owned by the enterprise are considered, one can assume that each domain is unique to each enterprise and all potential enterprise websites will only appear in the results a few times. All domains appearing much more frequently can then be discarded as blocklist domains (see

Ch. 8.2). Additionally, with a relation of 1:1 for each enterprise and website, all conclusions derived from the website are valid for that respective enterprise. If the enterprise-website relation is many-to-many, we cannot know for sure anymore if the derived characteristic based on a website is valid for all enterprises presented on that website or only for some of these enterprises. As an example, take a website of a specific brand that contains an online shop and subpages for each enterprise (i.e. local store) that sells this brand. Can we assume that all stores presented on the subpages also take part in e-commerce? In many cases, this assumption will not hold. To summarize, the derivation of enterprise characteristics from shared websites gets extremely complicated very fast.

The goal of URL finding in ESSnet Big Data I & II (and WIN WP 2) has been to supplement the results of the ICT survey. Using a different concept of the enterprise website in URL finding than in the ICT survey seems problematic, especially if it is the long-term goal to shorten the ICT questionnaire and substitute some responses with web scraped data. However, so far, the goal was to produce experimental statistics that serve as proximate indicators for the proportion of enterprises having a website.

Searching for only the main website per enterprise has the disadvantage that some information in further steps of the analysis is lost. As an example, an enterprise might have a first domain with general information and job advertisements as well as a second domain with an online shop. For the ICT survey, this enterprise would answer questions about the combined functionalities of all its domains. If the URL finder only considers one of these domains, some aspects of the web functionalities of that enterprise would get lost. This would be similar to not fully or correctly completing the ICT survey. Additionally, restricting the URL search to only domains owned by that enterprise does not reflect reality in many cases, since many enterprises have their website on the domain owned by a different enterprise.

In summary, there are many arguments for trying to predict several websites (and domains) per enterprise: conformity with the ICT survey definition of the enterprise website, a better reflection of enterprise websites in reality and less information loss when analyzing the data from enterprise websites in later steps. It seems worthwhile to explore this possibility, especially since the goal of OBEC (Online based Enterprise Characteristics) in WIN WP 2 is not to produce experimental statistics anymore, but to use URL finding in production.

3 Obtaining training and test data

In order to train and evaluate the URL finding method, statistical offices need to have a data set that contains the URL of a substantial number of enterprises. Training data can be obtained by using and combining already existing URL data. This may have the drawback that often only one URL per enterprise is available and that the data quality might not be very good (e.g. URLs could be missing or outdated). Ideally, one should create such a training data set manually. There are two options for this manual search:

1) Enterprises are manually searched (using a search engine), relevant results are visited and compared with the enterprise data by a person.
2) Enterprises are searched using the URL finder and the data is scraped. The scraped results are then opened in a browser by a person who selects the correct website for that enterprise (again by comparing the data on the website with that in the business register).

Option 2) has the advantage that the resulting training data fits the task perfectly. In option 1) two problems arise: URLs could have been identified as correct that were not found by the URL finder search or the URL finder search found URLs that are correct, but were not identified as correct URLs by a person. Additionally, if a correct enterprise URL was not scraped correctly by the URL finder, this hampers the quality of the training data in option 1) but would not do so in option 2). Consequently, it can be expected that the training data of option 2) has a better quality than that of option 1). On the other hand, option 2) has the disadvantage that the created training data depends heavily on the used search engine and the pages visited by the scraper. Additionally, manually inspecting scraped content can be much more cumbersome than performing a manual search, since scraped data is often not displayed in a browser in the same way as the original website was.

If the manual construction of the training data set is not feasible due to limited time and resources, it is also possible to use URL and enterprise pairs which can be linked deterministically as training data. The deterministic linkage can be done through VAT ID, commercial register number or any other unique ID, which can be found on commercial webpages and is available in the SBR (see Chapter 6 and 7 for more information). Enterprises are under some circumstances required to provide this information on their webpages.[3] The resulting training data can contain all enterprises with their linked URLs. URLs that were listed by the search engine but could not have been linked using the same set of enterprises should also be added to the training data. The idea behind this is that if an enterprise lists their VAT or commercial register number to identify itself on a webpage, it will do so for all webpages the enterprise owns. Thus, the training data is comprised of URL and enterprise pairs, which can and cannot be linked. Note that this procedure yields some disadvantages compared to the manual creation of a training data set. The data set has a high risk of being biased since it is only comprised of enterprises that can be linked through a direct identifier. In addition, this approach should only be considered if the sensitivity of the linkage procedure is very good, e.g. enterprises list a unique ID on their website that unequivocally corresponds to the unit of interest. One upside of this approach is that one can potentially build up a very large training data set.

The training and test set should be a sample from the targeted population. If the targeted population only contains enterprises with 10 or more employees, a random sample should be drawn, possibly stratified by economic activity. If the targeted population contains a high proportion of very small enterprises (<10 employees), bigger enterprises should be (heavily) oversampled. One can expect that a higher proportion of small enterprises do not have a URL. Training data with a majority of small enterprises might contain too few examples of enterprises with URLs.

A sufficiently big training and test set is crucial if one uses machine learning. If the training data is too small, variation within the data is not represented sufficiently. If the test set is too small, the results of the performance scores are unreliable. This can be checked by bootstrapping from the test set and recomputing the score function. If the scores vary too much, one should enlarge the test set. Similarly, one can bootstrap from the training set, retrain the model and recompute the score on a fixed test set. The size of the training set should be increased if this produces vastly different scores. Likewise, one can also simultaneously bootstrap from both training and test set to compute the combined effect.

4 Searching

The choice of the search engine and the search term is crucial for URL finding because if the search query did not return the correct URLs, any subsequent steps will not produce correct results either. Therefore, the percentage of correct URLs returned by the search engine is the upper limit of URL finding performance.

[3] See Directive 2000/31/EC, Art. 5. (https://eur-lex.europa.eu/legal-content/EN/ALL/?uri=celex%3A32000L0031)

Queries to the search engine can either be done through an API or through scraping. An advantage of an API is the availability of custom settings. As an example, one can define URL patterns in the Google API, which excludes whole domains or websites from the results. An advantage of scraping search engine results is that the results are identical to a "normal" search with that search engine and might contain more information (e.g. Google Maps results). However, scraping a search engine might violate its terms and conditions.

Search engines usually offer a search syntax to fine tune search terms. That way, the search term can be used to exclude some common file types that cannot be rendered to HTML (e.g. docx, pdf, pptx). Be aware that a search term is usually automatically shortened if it exceeds the maximum search term length of the search engine.

4.1 Search terms

The search term usually contains the enterprise name and sometimes additional information, for example the location of the enterprise. Statistics Netherlands (CBS) compared six different search term combinations to see how many of the correct domains were retrieved and how often the correct domains appeared in the search results using the Google API (Delden et al. 2019). The search term "legal/trade name + postal code + 'contact' + '-site'"[4] gave the best results (the correct domain appeared at least once in the search results for 72.2% of the enterprises). However, similar search terms (e.g. street name instead of the postal code) performed nearly just as well. The Italian National Institute of Statistics (Istat) achieved good search results with only the name of the enterprise (Barcaroli et al. 2016), but results further improved when adding the VAT-ID to the search term. The choice of the search term will also depend on language- and country-specific factors. As an example, the ZIP code as part of the search term worked well for CBS because Dutch ZIP codes are easily recognizable (four digits followed by two letters), while German ZIP codes consist of only five digits. These five digits could also be found within any other arbitrary number, which might produce wrong hits. In this case, the municipality name might produce better results. An argument for using the municipality name is also that enterprises might mention it more often on their websites than their ZIP code.

4.2 Search engine comparison

Amongst all search engines, Google dominates the market by a large margin. However, numerous other search engines are available which might be useful for URL finding.

For the choice of search engine, two criteria are most important:

1. Can the search engine identify the correct URL of a searched enterprise?
2. Is it possible to send a large number of requests to the search engine in a short time, preferably with no or low costs?

To judge the quality of search results, we compared Google, Google API, Bing, Yahoo and Duckduckgo, which have been used in previous URL finding projects conducted within the ESS. We compared the performance of these search engines for a sample of Italian and Hessian enterprises. The Italian sample consisted of 99 enterprises with URLs, which have been downloaded from the publicly available website https://www.downloadaziende.it. The name was used as search term in each search engine. Additionally, we used a sample of 100 Hessian retail trade enterprises with manually checked URLs. There, we used a combination of the enterprise name and municipality as search term.

[4] '-site' contained a list of blocklisted domains to be excluded from the search results.

Table 1: Comparison of search results

% domains matched	GOOGLE	GOOGLE API	BING	YAHOO	DUCK
Italian sample	74.8	66.7	64.7	63.6	57.6
Hessian sample	89	87	62	59	NA[5]

Table 1 shows the proportion of successfully retrieved URLs within the samples. According to this small comparison, Google seems to be the best possible choice. Especially scraping the search engine obtained the best results. The Google API still yielded better results than the other search engines. The difference between the Google search and Google API in the Italian sample is noteworthy. Google states that search results between normal search and API differ because the API "doesn't include Google Web Search features such as Oneboxes, real-time results, universal search, social features, or personalized results" and it is "limited to a subset of the total Google Web Search corpus".[6] However, there is no information available on how big this subset of the total Google Web Search corpus is or which and why websites are excluded from the API search.

The big differences in the Google API results for the Italian and Hessian enterprises, with 66.7% versus 87% of enterprises with correctly found URLs, could be due to the small samples. As mentioned earlier, the correct URL was returned for ca. 72% of the enterprises in the more representative Dutch sample. The search performance for Hessian enterprises in a larger sample was much better: when searching for 1466 retail enterprises with at least one manually checked URL, at least one of the correct URLs was among the search results for 91.5% of the enterprises. This result might be in part due to a better URL data quality, because the Hessian URLs were searched manually while the Dutch and Italian URLs originate from already existing, maybe older data sources. Additionally, the Hessian data included more than one URL for some of the enterprises, which makes it more likely that at least one of the websites was found.

To compare the effect of the search term and the Google API configurability, the Hessian sample of 100 enterprises was searched with different search terms (Name vs. Name + Municipality) and different configurations (blocklist exclusion patterns in the API vs. no blocklist exclusion patterns). The differences are relatively small, but slightly more enterprises were found when adding the municipality to the search term and when using exclusion patterns (Table 2).

Table 2: Google API configuration comparison (100 Hessian enterprises)

Google API configuration	% domains matched
Name + municipality, no exclusion patterns	87
Name + municipality, with exclusion patterns	89
Name, no exclusion patterns	83
Name, with exclusion patterns	87

To conclude, even though results vary widely between countries, the best search engine in terms of result quality seems to be Google (normal search or API). However, it is not advisable to scrape Google search results for a large number of enterprises. This does not only violate Google's terms and conditions but a scraper would also get blocked quickly. In general, when scraping search engines, one should read the

[5] Duckduckgo returned implausibly few search results for the Hessian sample. We therefore excluded it.

[6] https://support.google.com/programmable-search/answer/70392?hl=en

terms and conditions carefully to make sure that they are not violated. The APIs of search engines usually are only free for a limited capacity of requests. As an example, Google API currently costs $5 for every 1000 requests and the first 100 requests are free. Bing API costs $3 per 1000 requests. To summarize, the second criteria, the feasibility of sending a large number of requests to a search engine with low costs, can limit the available options of search engines (or APIs) substantially. Considering the better quality of search results, it currently seems advisable to invest in using the Google API.

Table 3: First four search results in Google API for search term "Destatis"

1	`https://www.destatis.de/EN/Home/_node.html`
2	`https://www.destatis.de/`
3	`https://www.destatis.de/EN/Press/2021/04/PE21_208_611.html`
4	`https://www.destatis.de/EN/Themes/Society-Environment/Population/Migration/_node.html`

5 Scraping

After searching enterprises, the search result URLs are subsequently scraped. However, search results return URLs to specific paths on the domain (see Appendix A for the structure of URLs). As an example, the first four Google API results for the search term "Destatis" are shown in Table 3. All of these URLs lead to the same domain `destatis.de`, which can be considered the correct website of Destatis.

When scraping, one has to decide if only the (unstandardized) result URLs, the landing page of the domain or other URLs on the same domain should also be scraped. Since the goal of URL finding is to compare the data of the website with the enterprise data, the pages where the enterprise information is likely to be found should be scraped. Frequently, the enterprise name, contact details, address as well as register numbers and VAT-IDs can be found on the landing, contact or imprint page. Commercial websites within the EU are required to provide this information easily accessible on their website.[7] There might be country-specific regulations that define more specific or even stricter requirements for website owners. Therefore, in general, not only the result URL should be scraped, but also the landing, contact and imprint page (if they exist). Simple regular expressions can be used to identify the contact and imprint pages. When scraping several URLs on the same domain, there should be some idle time between requests. In general, the guidelines for ethical web scraping should be followed (Stateva et al. 2019).

These guidelines also encourage to respect the robots.txt exclusion protocol. It is a standard protocol where website owners can specify which parts of the website web robots are allowed to visit. The instructions for the web scrapers are listed in the file `/robots.txt`. There are a number of existing libraries such as the reppy Python module (Reno 2019) or the robotstxt R-Package (Meissner and Ren 2020) which can be used during scraping to check the file `/robots.txt` before loading the website or any of its subpages. Adhering to the robots.txt is always recommended, however, the robots.txt might restrict the web scrapers from collecting the information needed for further statistical analysis.

The search engine result websites will be very heterogeneous, e.g. in their technical complexity. Many websites have dynamic web content. Therefore, scraping should be done with a javascript rendering functionality (e.g. an automated browser). Unfortunately, rendering javascript increases the bandwidth usage compared to simply sending HTTP GET requests.

[7] See Directive 2000/31/EC, Art. 5. (https://eur-lex.europa.eu/legal-content/EN/ALL/?uri=celex%3A32000L0031)

Scraping enterprise websites on a large scale produces big amounts of downloaded data. Both bandwidth usage and storage capabilities can be an issue. To reduce the bandwidth usage, one could for example choose to not download images. Instead of saving the full HTML code of websites, one can perform some preprocessing steps to the HTML code before saving or only save the extracted information from the website. With more preprocessing steps before saving the contents, less storage will be needed. However, saving the full HTML code of websites might make sense as well if the websites are analyzed for different purposes later on. Scraping a large number of websites is time consuming, computationally expensive and could become a burden on website owners if done too frequently.

When searching for enterprise websites, some identical URLs will appear several times in the results. As an example, this can happen because websites are shared between enterprises, or because the URL finder visits specific links on the found domain (e.g. the landing page or imprint page). It is desirable to scrape the same URL only once (and save the scraped data only once). Otherwise, the same URL will be scraped for several enterprises respectively. If the URL finding takes place over a longer period, this might lead to different scraped content for the same URL for different enterprises and it will require much more storage. Another challenge are redirects and URL synonyms. Often, different URLs will be redirected to the same website. This makes the deduplication of URLs more difficult. When a URL is redirected, the URL after the redirect should be logged and used for all subsequent steps. As an example, when the search engine gives `http://www.statistik.hessen.de` as a result, but it is redirected to `https://statistik.hessen.de/`, the latter URL should be stored for any further use.

During scraping, errors will always occur. Technical errors are identified by a HTTP status code unequal 200. These errors should be recorded. Depending on the HTTP status code, a website could be scraped again later. Sometimes a website will display only a captcha because it identified the scraper as bot. It would be desirable to scrape that website again at a later time. Since there is no standardized format of captchas that can be used to identify them, it is challenging to implement this procedure[8] (and no URL finder has this functionality so far).

6 Data extraction

After the scraping process, features are extracted from the obtained data that will later on serve as input for a classification model. Most importantly, the agreement between the enterprise data from the SBR and the scraped websites has to be quantified.

Depending on the country, different enterprise information will be available for comparison with website data. Enterprise data often consists of the name, address, register/tax IDs and contact information (phone number, email). When comparing the contents of webpages with business register data one should also consider the form in which the information may appear on the webpage. Often, it is unlikely that the enterprise name, address and phone number appear on the website in the exact same way as in the business register. There could be spelling variations (or even typos) in the website or (occasionally) in the business register. Even register numbers and tax IDs can vary in their spelling (e.g. whitespaces in arbitrary places). When preprocessing website and business register data, the goal is to reduce these spelling variations as much as possible.

Preprocessing steps often include: removing duplicate whitespaces, lowercasing words and letters and removing/standardizing language specific characters or special characters in general (e.g. the German “ü”

[8] Be aware that we do not want to solve captchas, only identify their presence.

Web Intelligence Network
Funded by the European Union

to “ue”). One could remove HTML tags, css styles and javascript codes, so that only plain text remains. Especially when searching for short numeric enterprise data (e.g. ZIP codes) website code could generate false positives. URL finders should offer different or easily customizable preprocessing steps so that they can be adapted to country and language specific requirements.

After preprocessing, the available enterprise data is compared with the website text. Usually, exact string matching or regular expressions (to allow for spelling variations) are used. Regular expressions with spelling variations are preferable to exact string matching. A second option would be to first extract only short, relevant text parts of a website and then compute similarity scores with the enterprise data. Since all enterprise websites have different structures, named entity recognition could be used to extract only relevant information (Nadeau and Sekine 2007). However, this has not yet been done in any of the developed URL finders. A third option is to use schema.org microdata, which is used by some websites to provide machine-readable information. Extracting this microdata and then using similarity scores to compare it with enterprise data could be an improvement over using only exact string matching and regular expressions. However, only few websites provide microdata, which makes it doubtful that extracting microdata will improve model performance.

Data from the search snippet or the URL itself can be compared to the enterprise data using string similarity. Furthermore, it is possible to extract the HTML document title (or headers) and compute a similarity to the enterprise name. To compute the string similarity (or string distance) between the enterprise data and words from the website, the search results or the URL, string distance measures like the Jaro-Winkler distance or Levenshtein distance can be used. The Levenshtein distance counts the number of deletions, insertions and substitutions necessary to turn one text block into another. String distance metrics are available in R for example within the stringdist library (van der Loo 2014).

The assumption behind all suggested features is that a website, which contains the enterprise data, is a website of that enterprise. However, this is by far not always the case. Numerous websites only collect and display information about enterprises (sometimes by simply scraping other yellow pages). Sometimes, they contain the legal information about these enterprises more accurately than the enterprise websites themselves. See Ch. 8 on how to deal with this issue.

Besides website-enterprise agreement features, the search position and other search engine data could be used to create features. It can be assumed that websites with a higher search position are more likely to be correct. However, search position as feature should be considered with caution (since a high ranking alone does not necessarily signify relevance). Depending on the country, additional external data sources could be used. As an example, the ICANN (Internet Corporation for Assigned Names and Numbers) provides not only a gTLD (general top level domains, e.g. .com) level lookup tool for domain names but also TLD register lists where e.g. country specific TLD registers can be found. In some cases, for example in Finland, the organization responsible for distributing .fi domain names has both a web based search tool and an API for database access, and are willing to let the National Statistical Institute (NSI) extract the data. Web based lookup tools usually provide no more specific information of the owner of a certain domain than the name of the company, but at least the Finnish API provides also business ID’s. The API can be queried with the name or business ID and returns all registered domains for that enterprise, which contains quite often several hundred domains. Most of these domains are often not in use, but the information is useful for URL finding. If a URL returned by the search engine is among the registered domains of the enterprise, this is a strong indicator that this URL is correct. This can be used as an additional feature for the machine learning algorithm.

After deriving features from websites, the features should be aggregated to the domain or subdomain level. The maximum of each feature seems most suitable, especially when only detecting the presence or absence of characteristics on a website, but depending on the type of feature the mean or minimum could be useful as well.

7 Classification algorithm

After features are prepared, each scraped website has to be labelled (TRUE/FALSE). The training data is used to obtain these labels. It is best to prepare a training set with (*legal unit, URL)* pairs that contains the following options:

1) A scraped website is found, and this website is indeed the dedicated website belonging to the legal unit of interest (TRUE).
2) A scraped website is found, but this website is not the dedicated website belonging to the legal unit of interest (FALSE).

So, if the website is correct according to the training set, it gets the label TRUE, otherwise FALSE.

We can also have the situation that the search result does not obtain any dedicated websites. This result might be obtained after we have removed search results that we consider irrelevant such as collective websites (yellow pages) and HTTP errors. When the training set consists of (*legal unit, URL*) pairs, one can also include legal units without a URL (URL is empty), with labels:

3) No websites are found and the legal unit does not have a website (TRUE);
4) No websites are found, but in reality the legal unit does have a website (FALSE).

There are now at least three possibilities in how to deal with these two groups of legal units – those with and those without a website. Firstly, one could limit the scope to the scraped websites and determine whether they belong to the real units of interest or not, so just using the cases 1 and 2 above. In this case, one does not *explicitly* try to predict whether legal units have a website or not. Secondly, one could make a single classification model that encompasses the cases 1-4 simultaneously, as was done for instance by Van Delden et al. (2019). Thirdly, one could make two separate classification models, one model that predicts whether the legal unit has a URL or not and a second model – for legal units that are predicted to have a URL – that aims to predict whether the URL that is found is the correct one. An advantage of the third approach is that each of the two models can use different features. For instance, features for the model that predicts whether a legal unit has a URL or not could be a) the number of returned search results and b) whether the legal unit is a registered owner of a domain name in certain lists with domain names. Statistics Finland for instance has access to a list with domain names in Finland.

The classification model that one aims to fit can be based on different types of algorithms: rule-based and machine learning. The machine learning can be further divided into classical machine learning and deep learning models. For rule-based models, classification rules are defined beforehand (e.g. if the correct VAT-ID was found, the label is TRUE).

URL finding should always be evaluated on an independent test set. When using machine learning, it is advisable to train and evaluate the different models on several training and test sets, for instance by using cross-validation (Stone 1974). One can then judge the overall, mean, evaluation results, but one can also look into the variation of the evaluation results over the different folds of the cross-validation. Models that show a smaller variation are expected to deliver more stable results.

Evaluation can be applied to the level of the websites (URLs) and to the level of the legal units. The legal unit level is of primary interest, since we want to know: for how many enterprises did we get it right? How one exactly evaluates the outcomes depends on the approach that one takes:

- *'URL-found' - model*: only considering a model for cases 1 and 2;
- *a combined model*: building one model for cases 1-4;
- *two separate models*: one for cases 1,2 and one for cases 3,4.

In case of the *'URL-found' – model* and the situation with *two separate models* the outcomes are binary (TRUE / FALSE) so one can compute the usual evaluation measures: F1, recall and precision. Results of the *combined model* are less straightforward to interpret: although the outcome itself is binary, the meaning of TRUE/ FALSE depends on whether the legal units have a website or not. In fact, we could consider the problem as that we are interested in predicting the combination of legal unit (LU) k and URL, where the true value for URL can be either ∅ (empty) or non-empty with true value g. One can predict whether the URL is empty, URL g or URL h $(h \neq g)$.

One can summarize this with a confusion matrix in Table 4.

Table 4: Confusion matrix in the combined model

True pair	Predicted pair		
	LU k, URL ∅	LU k, URL g	LU k, URL h $(h \neq g)$
LU k, URL ∅	TRUE	FALSE	FALSE
LU k, URL g	FALSE	TRUE	FALSE

Note that in practice a legal unit with a website may have in reality multiple websites g, therefore multiple predicted websites g could be correct.

When the *combined model* is used, the usual F1 score is not unambiguously defined, in fact one could compute an F1 score for (LU k, URL ∅) pairs and for (LU k, URL g) pairs separately. Unfortunately, it is not straightforward to combine them into a single measure, because the true population frequency of legal units without a website may not be accurately known. Maybe an estimate is available from the ICT survey. Alternatively one could compute the accuracy as the proportion of (LU, URL) pairs that are predicted correctly[9].

8 Blocklist

Some URLs should be excluded before or after scraping. URLs with content that cannot be rendered to HTML should be filtered out before scraping when possible. This is done by applying a file blocklist. Additionally, domains that contain the enterprise data but are no enterprise websites should be excluded from the classification (or preferably in earlier steps so that these sites already do not have to be scraped and processed).

8.1 File blocklist

Numerous file types exist on the web that can be rendered to html and even more that cannot. As mentioned in Ch. 4, some common file types with non-html content should be already excluded in the

[9] Not finding all websites of an enterprise is no error, because it is not possible to know for sure how many websites an enterprise has.

search term. When scraping, a larger file blocklist should be applied with less frequent file types (see Appendix B for an example).

8.2 Domain blocklist

A big challenge in URL finding is websites that contain all data of the enterprise of interest, but are not enterprise websites. This occurs frequently on all kinds of websites, for example: websites of municipalities, online telephone books, ecommerce platforms, yellow pages, business directories, social media, etc. We call these websites blocklist domains. The most frequent domains in the search results are usually excluded and added to the blocklist. However, there are two problems with this approach: firstly, some blocklist domains only appear very infrequently in the results, but since there are so many of them, they still pose a problem. Secondly, frequent domains can be enterprise websites, because some enterprises share websites. Therefore, when we assume that enterprises can have more than one website and that they might share websites, whenever a URL appears often in the search results, we cannot assume that it is a blocklist URL. It is necessary to manually determine if this URL is a shared website or a blocklist URL. The blocklist can become quite big, with several hundred domains and possibly even more.

A solution to this problem could be to create a blocklist detection model that filters out these blocklist domains. To that end, already known blocklist domains and enterprise websites could be used as training data for a machine learning model. Some research will be necessary to determine if this additional classification model can improve results and decrease the time needed to manually identify blocklist URLs.

9 URL finding variations

This chapter describes some alternative approaches and applications for URL finding. The first alternative approach omits the scraping of websites and uses only search results. The second alternative approach uses email addresses as data source. Lastly, we describe an idea to use URL finding for searching e-commerce platforms. This demonstrate that software to automatically search and analyze web sources is versatile and potentially useful for other applications in official statistics.

9.1 Using only search results

In the first alternative approach, only search results are used as data source and no additional scraping is done. This approach has been implemented first by CBS (Delden et al. 2019). They sent six different search queries for each enterprise to Google API. The search terms gave in part different results and therefore supplemented each other. Considering that the best performing search term found the correct URL for only 72% of the enterprises, sending several queries reduces the error due to the search engine, which is a main limiting factor for URL finding performance. Instead of scraping the results, only the search result data (URL, title, snippet, pagemap, search position, frequency of a domain in the results) was used as data source to create features. Delden et al. (2019) quantified the agreement between search results and enterprise data with string similarity measures. Compared to scraping the result URLs, this approach is computationally inexpensive and fast. When using search result data, one can benefit from Google´s unparalleled expertise in crawling and extracting relevant information. On the other hand, Google´s algorithm to produce search result data is not transparent and might not always reflect the current content of the website. Search results might not always be reliable or contain the information we are interested in. Furthermore, sending more search requests is costly when using Google API. Despite these drawbacks, the CBS URL finding approach performed with a F1 score of 0.84 very well (Delden et al. 2019).

9.2 Finding URLs with email addresses

As already mentioned, another possible way to identify enterprise URLs is looking at the available enterprise email addresses. Due to the fact that the number of cases (i.e. enterprises with known email address but unknown website URL) is quite high, it is necessary to automate the process as much as possible in order to minimize the number of difficult cases that have to be searched manually.

Let us consider the following email address: `marc.white@mycompany.com`. The first part of the address (the string before the "@" character) is called the local-part and usually contains the name of a person or the name of a specific service or group of people (e.g. info, support, maintenance, unit1). The second part of an email address (the string after the "@" character) is called the domain name and is an identification string that defines a realm of administrative autonomy, authority or control within the Internet.

Normally, for URL retrieval purposes, the interesting part of email addresses are the domain names because, being unique identifiers, they might also be used as website URLs. Excluding the case in which no email addresses for a specific enterprise are available, the problem is that there is no guarantee that the domain name extracted from an "enterprise" email address is also used for the official enterprise website for various possible reasons, for example:

1) the email was obtained by third part company that operates as a general email service provider (e.g. info.mycompany@gmail.com)
2) the email was obtained by third part company that operates as a certified email service provider (e.g. info.mycompany@pec.it)
3) the domain name was misspelled (e.g. info.mycompany@gmaill.com)
4) the company owns several domains and uses different domains for the website and email
5) in case of a small franchise company the domain belongs to the umbrella company
6) the email address is not a company address but a personal one used by an employee of the company (e.g. adam@hotmail.com).

Each of these (non-exhaustive) cases should be treated differently. A good approach would probably be to define a workflow that has to be followed for every email. Possible steps in the workflow could for example be the following:

For the first two cases mentioned above, a possible strategy for being reasonably confident that the extracted domain is also the enterprise website could be as follows.

1) Verify whether the extracted domain for a specific enterprise is unique among the list of all enterprise domains; if it is not unique there are good chances that the domain is not suitable for the URL finding purpose. (Unless in special cases such as franchise companies, it is unlikely that two or more enterprises share the same domain.)

2) Compare the extracted domain with a list of domains belonging to official email provider companies (e.g. gmail, hotmail, tiscali); if a match is found the domain is not suitable for the URL finding purpose. (Unless the enterprise of interest is one of the email service providers.)

For the third case, it would be necessary to make a string comparison between the string representing the enterprise name and the string extracted from the email address in order to assess a possible similarity.

In some cases, especially for very small enterprises, the local-part of email addresses could also be interesting because it could contain the name of the enterprise which is also used as a domain (e.g.

bottegapalladino@gmail.com is the contact email address and the related website, manually verified, is https://www.bottegapalladino.com/).

In any case, a final check of the obtained URL will be worthwhile, for example by scraping the domain and analyzing the content. However, in the vast majority of the cases the expected result should be the correct website.

9.3 E-Commerce Platforms

Using search engine data and scraping the results could also be used for different use cases than finding enterprise URLs. One potential use case concerns e-commerce. Some enterprises might not use their own website for e-commerce but might use e-commerce platforms instead. E-Commerce platforms are excluded from URL finding by definition, which means that estimating e-commerce participation from enterprise websites will be biased. One could therefore perform supplementary searches of enterprises only on e-commerce platforms. This would require a list of all important e-commerce platforms in the respective country. When using Google (or any other) API, one could restrict the search to these platforms by defining them as URL inclusion patterns. All results will then be restricted to these platforms and return subpages of the enterprise on these platforms, if there are any. Similar to URL finding, one should then scrape the results or use the search data to create features and classify if these subpages really belong to the enterprise.

10 Legal issues

URL finding involves sending requests to a search engine with the name and address (or other contact or identifying information) of an enterprise. One has to assume that the search engine saves search queries and is technically able to identify the search queries as business register data from a NSI. For that reason, some NSIs have legal restrictions on which data can be used for searching.

Overall, the German Federal Statistics Office (Destatis) allows URL finding by sending search queries to a search engine with the enterprise names from the SBR paired with additional information e.g. the address, register numbers, or the VAT-ID. However, Destatis suggests several approaches to comply with statistical confidentiality. For example, it is advised to use various search engines for queries. Search engines should – if possible – save as little information as possible, have their legal base in the EU and store the data there. If one enterprise is searched with different combinations of information, a new IP-address should be used. Lastly, it is suggested to anonymize the search query, e.g. by placing a number of false search queries for each real search query.

At Statistics Austria, searching for enterprises through a search engine is not forbidden per se, there are however restrictions on how to use the search engine. In general, Statistics Austria uses the name of the enterprise as well as the full or part of the address. For most types of legal units, this does not breach statistical confidentiality. In the case of one-man businesses, however, the search string cannot include the company name, which might be the name of a person. For these cases, Statistics Austria only uses the address in the search string, which is publicly known anyway.

To conclude, different legal interpretations if and under what conditions search queries with enterprise data (e.g. names and addresses) may be send to a search engine make an ESS wide adaption of URL finding challenging.

Web Intelligence Network

Funded by the European Union

11 Conclusion

The methodology for URL finding has been developed in the ESS for several years already, starting with a first URL finder developed by Istat (Barcaroli et al. 2016) and continued during the ESSnets Big Data I & II, during which time further URL finders were developed by CBS (Delden et al. 2019) and BNSI[10]. Appendix D gives an overview over some URL finders within the ESS. Their performance is already satisfactory. However, different NSIs and ONAs (Other National Authorities) developed – and are still developing – their own URL finders, in part for the following reasons:

1. country- and language-specific particularities required different methodologies that were not available in already existing URL finders,
2. existing URL finders were not written in programming languages that staff members in other NSIs/ONAs knew,
3. existing URL finders were not suitable for the (web scraping) IT infrastructure in other NSIs/ONAs and
4. existing URL finders were too slow or too computationally expensive to be run on a large scale.

Considering the complexity and importance of URL finding, a scalable and highly configurable URL finder that is available for the whole ESS would be desirable. This report summarized the different methodologies that should be implemented within a common software for URL finding. A detailed description of technical requirements goes beyond the scope of this report, but an attempt for a list of requirements was made in Appendix C.

While the methodology in currently existing URL finders has been proven effective, this report identified several areas for further improvement. One of these areas concerns the conceptualization of enterprise websites. Most URL finders so far worked under the assumption that each enterprise can only have one (main) website on a domain that is owned by that enterprise. This assumption does not always hold in reality and is not in line with the definition of the ICT survey. However, we do not know for sure how often enterprises share websites or maintain several websites for different purposes. It is therefore unclear to what extent the assumption of only 1:1 (or 1:0) relations between enterprises and websites causes a bias when deriving information from enterprise websites.

Research should be dedicated to how enterprises present themselves on the web: how often do they own a domain and use that as their main web presence? How often do they have several websites or share websites with other enterprises? If there are several websites and given a specific use case (e.g. e-commerce or NACE code classification), which of them contains the information we are interested in?

Another area for further research is the data extraction. So far, exact string matching is often used to identify enterprise data, but website text contains the information in unpredictable shapes, e.g. spelled differently. String similarity scores could account for these variations, but it is impossible to compute string similarity scores with each segment (e.g. word, sequence of words) on the website. Taking the CBS URL finding approach as an example, one could also use search engine data more extensively. Since search engine data is much more concise, similarity scores can be computed. Furthermore, extracting relevant text segments from the websites with Named Entity Recognition and computing similarity scores

[10] Available on Github: https://github.com/EnterpriseCharacteristicsESSnetBigData/StarterKit/tree/master/URLsFinder

afterwards could be investigated in the future. One should research if machine learning models improve when these different types of features are combined.

Besides using website or search engine data for URL finding, additional country-specific data sources seem promising. Both domain registry data and email addresses contain information on enterprise websites, even though the respective data source is often not sufficient to identify the correct website unambiguously. Both sources could be combined with URL finding, e.g. by comparing parts of the email address or the registered domains with the found domains. The result of the comparison could be added as a feature to the Machine Learning algorithm or could be used to define deterministic decision rules for classification.

Good search engine results are crucial for URL finder performance. In Dutch and Italian samples, the correct website was in 25-40% of the queries not among the first 10 search results (Ch. 4.2). That is problematic, because no matter how good the data extraction and classification algorithm are, the URL finder will not be able to identify the correct website for this proportion of enterprises. There are many possible reasons for an insufficient search result quality, e.g. out-of-date or low quality URL data, a non-optimal search term or a general bad performance of the search engine for that language/region. In general, when using an API, one should study and test different configurations (e.g. language settings, exclusion patterns). Another strategy to improve search engine results is to send several requests per enterprise with different search terms. Improving the quality of URL data will also lead to a better match between search engine results and correct enterprise URLs according to the training data. Manually searching websites of a sample of enterprises and creating training data that way seems the most effective – even though time-consuming – method to do so.

To conclude, parallel to the current efforts to move URL finding towards production within the ESS, further research to improve URL finding performance should be conducted.

12 References

Barcaroli, Giulio; Scannapieco, Monica; Summa, Donato (2016): On the use of internet as a data source for official statistics: a strategy for identifying enterprises on the web. In *Rivista italiana di economia, demografia e statistica* 70 (4), pp. 20–41.

Delden, Arnout van; Windmeijer, Dick; ten Bosch, Olav (2019): Finding enterprise websites. Bilbao (European Establishment Statistics Workshop).

Meissner, Peter; Ren, Kun (2020): robotstxt: A 'robots.txt' Parser and 'Webbot'/'Spider'/'Crawler' Permissions Checker. Available online at https://CRAN.R-project.org/package=robotstxt.

Nadeau, David; Sekine, Satoshi (2007): A survey of named entity recognition and classification. In *Lingvisticae Investigationes* 30 (1), pp. 3–26.

Reno, Lindsey (2019): Reppy: Robots Exclusion Protocol Parser for Python. Available online at https://pypi.org/project/reppy.

Stateva, Galya; ten Bosch, Olav; Maślankowski, Jacek; Summa, Donato; Kowarik, Alexander; Condron, Aidan et al. (2019): ESS web-scraping policy template. ESSnet Big Data II (Deliverable C1).

Stateva, Galya; Kowarik, Alexander; Gussenbauer, Johannes; Summa, Donato; Maślankowski, Jacek; ten Bosch, Olav et al. (2020): Reference Methodological Framework for processing online based enterprise characteristics (OBECs) data for Official Statistics V.2. ESSnet Big Data II (Deliverable C6). Available online at https://ec.europa.eu/eurostat/cros/sites/default/files/WPC_Deliverable_C6_Reference_Methodological_Framework_v2.0.pdf, checked on 8/4/2021.

Stone, M. (1974): Cross-Validatory Choice and Assessment of Statistical Predictions. In *Journal of the Royal Statistical Society: Series B (Methodological)* 36 (2), pp. 111–133. DOI: 10.1111/j.2517-6161.1974.tb00994.x.

van der Loo, Mark (2014): The stringdist package for approximate string matching. In *The R Journal* 6 (1). Available online at https://CRAN.R-project.org/package=stringdist, checked on 11/19/2021.

Appendix

Appendix A URL components

Table 5 shows some common components of URLs with `https://ec.europa.eu/eurostat/cros/WIN_en` as an example. Many URLs also contain the prefix `www`. Technically speaking, this is a subdomain and does not necessarily need to be the same as the domain without `www`. Usually, website owners will have one “correct” version of their URL (e.g. without the `www`) and will redirect to this standard version if the user types in the “wrong” version. As an example, if one navigates to `https://www.ec.europa.eu/eurostat/cros/WIN_en` the browser will redirect to the URL without `www`.

Table 5: URL structure

`https://`	`ec.`	`europa.`	`eu`	`/cros/WIN_en`
protocol	subdomain	domain	top level domain	path

Appendix B Example for a file blocklist

The following file endings indicate that a URL does not contain content that can be rendered to HTML.

```
.mng, .pct, .bmp, .gif, .jpg, .jpeg, .png, .pst, .psp, .tif, .tiff, .drw, .dxf,
.eps, .svg, .mp3, .wma, .ogg, .wav, .ra, .aac, .mid, .aiff, .3gp, .asf, .asx,
.avi, .mp4, .mpg, .qt, .rm, .swf, .wmv, .m4a, .css, .pdf, .doc, .docx, .exe,
.bin, .rss, .zip, .rar, .msu, .flv, .dmg, .xls, .xlsx, .mng?download=true,
.pct?download=true, .bmp?download=true, .gif?download=true, .jpg?download=true,
.jpeg?download=true, .png?download=true, .pst?download=true,
.psp?download=true, .tif?download=true, .tiff?download=true, .ai?download=true,
.drw?download=true, .dxf?download=true, .eps?download=true, .ps?download=true,
.svg?download=true, .mp3?download=true, .wma?download=true, .ogg?download=true,
.wav?download=true, .ra?download=true, .aac?download=true, .mid?download=true,
.au?download=true, .aiff?download=true, .3gp?download=true, .asf?download=true,
.asx?download=true, .avi?download=true, .mov?download=true, .mp4?download=true,
.mpg?download=true, .qt?download=true, .rm?download=true, .swf?download=true,
.wmv?download=true, .m4a?download=true, .css?download=true, .pdf?download=true,
.doc?download=true, .exe?download=true, .bin?download=true, .rss?download=true,
.zip?download=true, .rar?download=true, .msu?download=true, .flv?download=true,
.dmg?download=true
```

Appendix C Technical Requirements

The following non-exhaustive list defines some technical requirements for a common URL finder software within the ESS. In general, this URL finder should be easily configurable, even without knowledge of the programming language it was written in.

- Customizable enterprise data (variables), search terms and blocklist
- Customizable header (user agent string, language and location settings)
- Error handling
- Automation of all/most steps (e.g. by scheduling tasks)
- Rendering javascript
- Easily add support for new search engines
- Configurability of text processing, feature creation and optional inclusion of external data (e.g. domain registry data)
- Scaling, e.g. by scraping in parallel
- Efficient data storage in a database
- Customizability of scraping (e.g. prioritize links with specified patterns)
- Possibility to predict more than one website per enterprise
- Modularity of searching, scraping and data extraction, e.g. send several search queries per enterprise and skip the scraping part

Appendix D URL finder comparison

Searching	BNSI	Istat	HSL	CBS
Search engines	Duck Duck Go, Google, Bing	Bing	Google API, Bing API	Duck Duck Go, Google API
Customizability of search terms	Yes	Yes	Yes	Yes
Other particularities				Several search requests are made for each enterprise and no further scraping is done.
Scraping				
Javascript rendering	No	No	Yes, with PhantomJS or Splash (can be chosen by user). There is also a non-Javascript rendering method available (httr::GET).	Not applicable
Which links on a domain are scraped?	First `n` and last `n` links from the page, where `n` is defined by user.	Only the URL explicitly suggested by the search engine.	Imprint ("Impressum"), about and contact page as well as landing page, up to 20.	Not applicable
Link deduplication	None	None	Yes, within the same found URL. However, since different search results will lead to the same domains or even URLs, there are some duplicates in the scraped pages.	Not applicable
Redirects	The final website is scraped.	The redirection is logged and the final website is scraped.	The redirection is logged and the final website is scraped.	Not applicable

Web Intelligence Network

Funded by the European Union

Data Extraction

Text processing steps	Lower case conversion, whitespace removal, string comparison	Lowercasing, punctuation removal, stemming, lemmatization		Tokenization, Jaro-Winkler similarity to enterprise data
Customizability of text processing steps	No	Processing steps are defined during the configuration of the storage platform, it would be difficult to modify them without the necessary expertise.	A custom function can be created that defines all text processing steps and features. This requires R programming skills. At the moment, two different functions exist from which the user could choose in the configuration file.	Not easily
Customizability of created features	Yes, with input file.	No, name of features and their number is fixed.		Not easily

Classification

Algorithms used	Logistic regression	Deterministic and ML (Neural Networks, Random Forest, Logistic Regression)	ML (Gradient Boosting, SVM, Random Forest, Logistic Regression, XGBoost)	ML (SVM, Random Forest, Gaussian Naive Bayes)
Can more than one website per enterprise be predicted?	No	Yes (in theory)	Yes	No
How is the blocklist maintained?	Manually, no defined procedure	Manually, no defined procedure	Manually, no defined procedure	Manually defined blocklist. Additionally: If a domain was returned more often than the maximum number of search results for one enterprise, it was excluded.
Which evaluation scores were achieved? (Be aware that scores might not be comparable)	Level of the websites: F1 of 0.59 (for URL correct), 0.72 precision	Level of websites: 0.79 accuracy, F1 of 0.81 (URL correct, logistic regression)	Level of enterprises: 82.3% of enterprises were assigned at least one correct website (or rightly no website). Level of the websites: F1 of 0.82 (URL correct, Gradient Boosting)	Level of enterprises: F1 of 0.84 (URL correct; SVM) Level of websites: F1 of 0.77 (URL correct)

Funded by the European Union

Logging and Storage

How are errors logged?	In a file	In a file	In the database	In a file
Which errors are logged?	All errors concerning scraping (Duckduckgo search and scraping) are logged	All errors concerning scraping	All errors in the process of URL finding (e.g. API errors, Scraping errors, errors in data processing steps)	Unsuccessful search requests (API errors and empty search results)
Where is data stored?	In files	In files and NoSQL (Solr)	In Postgresql database	In files
What raw data is stored?	The human readable text plus the textual content of some HTML tags	The human readable text plus the textual content of some HTML tags	All raw data (including API responses and full website HTML code)	All raw data (API responses)

Other information

Programming language	Python	Java	R	Python
Software dependencies (other than imports of packages in the same programming language)	None	Solr, TreeTagger	Apache Kafka & Zookeeper, Postgresql, PhantomJS or Splash	None
Tested on which operating system?	Windows 10	Windows 7, Windows 10, Linux	Linux	Windows 10, Linux

Web Intelligence Network

Funded by the European Union