ON THE USE OF INTERNET AS A DATA SOURCE FOR OFFICIAL STATISTICS: A STRATEGY FOR IDENTIFYING ENTERPRISES ON THE WEB[1]

Giulio Barcaroli, Monica Scannapieco, Donato Summa

1. Introduction

Internet as a Data Source is gaining more and more importance in Official Statistics. An increasing number of Statistical Institutes are indeed experimenting the use of new sources of data (also known as Big Data) in order to produce the same or new statistical information in a multisource environment, more efficiently and with higher levels of quality (Citro, 2014).

Many examples of the use of Internet data sources can be reported. For instance:

- Internet queries: the use of Google Trends has been evaluated in order to produce now-casting estimates of unemployment indicators (Fasulo et al., 2015);
- Web prices: Web scraping is already in use in order to collect prices related to goods and services for the construction of Consumer Prices Indexes (Cavallo, 2013);
- Social media: posts in social media, like Twitter or Facebook, can be used in order to support the production of traditional Official Statistics indexes like, e.g., the Consumer Confidence Index (Daas et al., 2012).

The nature of data originating from Internet data sources is such that a number of problems have to be addressed, before making use of them for statistical purposes. The most relevant issue is related to representativeness and selectivity (Buelens et al., 2014), In this respect, different important questions arise in order to assess the usability of a given source:

1. Is it possible to refer collected data to specific units in the population of reference, and, if this is the case, with which level of uncertainty?
2. To what extent the population of reference is covered by a specific Internet data source?

[1] Invited paper to the 53rd SIDES Scientific Meeting – Rome 2016.

3. With respect to the target estimates, is the distribution of related variables in the subpopulation covered by the source significantly different from the one not covered?

In this paper, we will consider a specific case of Internet as a data source, that is the one related to the information contained in enterprises' websites. The representativeness of these data is analyzed with particular regard to points (1) and (2) of the above questions. Also, methodological and technological solutions of the problem concerning the minimization of linkage uncertainty and the maximization of population coverage are described.

The Italian National Institute of Statistics (Istat) is experimenting the use of Internet data to obtain a subset of the estimates currently produced by the sampling survey on survey "ICT usage in Enterprises", yearly carried out by Istat and by the other member states in the EU. Target estimates of this survey include the characteristics of websites used by enterprises to present their business (for instance, if the websites offer e-commerce facilities or job vacancies). The aim of the experiment is to evaluate the possibility to use the sample of surveyed data as a training set in order to fit models that will be applied to the generality of websites (Barcaroli et al., 2015-a) (Barcaroli et al., 2016).

Let us consider the task of correctly referring Internet data to population units (in our case, respectively, data collected from websites need to be referred to the population unit "enterprise"). The availability of a population frame[2] containing all the enterprises included in the target population suggests to use it as the basis for a search of the corresponding websites. Actually, for a subset of enterprises the indication of corresponding websites is already available, as they can be obtained by the indications that sampled enterprises gave in the "ICT usage in Enterprises" survey, and from other sources that can be integrated with such information. However, the size of this subset is insufficient to adequately cover the total amount of enterprises that own a website (estimated in about two thirds of the total).

In this paper, we will detail a strategy to address such an issue, which we will call in short "the URLs retrieval problem". In particular, we will describe a specific process that we designed and implemented able to retrieve, for any given enterprise, the URL of the corresponding website, if any.

The paper is organized as follows. In Section 2, background information is given related to the research and development of Big Data projects within which Istat was involved and where this problem was tackled. Section 3 describes the particular strategy adopted for ensuring the maximization of the coverage of the population of interest, while containing errors related to URLs retrieval. In Section

[2] In Istat the available frame is ASIA ("Archivio Statistico delle Imprese Attive", Statistical Archive of Active Enterprises).

4, the results of the application of the strategy are reported and analyzed. Finally, Section 5 contains conclusions and indications on the future work.

2. Background Information

2.1. Web information extraction pipeline

Obtaining information from the Web is enabled by using Web scraping techniques. Web scraping (Web harvesting or Web data extraction) is basically a process that permits to extract information from websites.

It is possible to distinguish between two different kinds of Web scraping, namely: *specific Web scraping* and *generic Web scraping*.

Specific Web scraping is referred to the case when both structure and content of websites to be scraped are perfectly known, and scraping programs just have to replicate the behavior of a human being visiting the website and collecting the information of interest. A typical area of application is the data collection for price consumer indices; in that case one is typically interested to collect very specific information contained in a website (e.g. a value in a row of a table in a page).

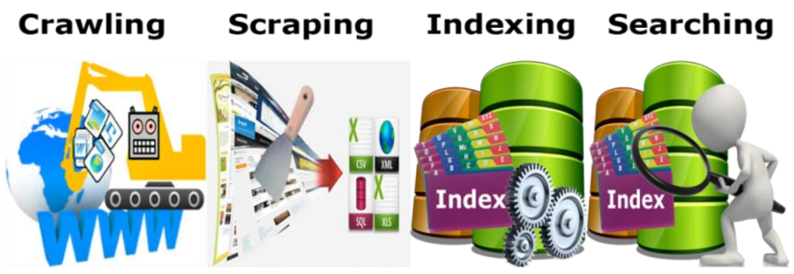
Generic Web scraping, instead, assumes that no a priori knowledge on the content is available, and the whole website is scraped and subsequently processed in order to infer information of interest. In such cases it is up to subsequent phases to retrieve any specific information that could be needed. An example of application of generic Web scraping has been performed for the survey "ICT usage in enterprises".

The Web information extraction process is composed by four main different phases (see Figure 1):

- Crawling: a Web crawler (also called Web spider or ant or robot) is a software program that systematically browses the Web starting from an Internet address (or a set of Internet addresses) and some pre-defined conditions (e.g., how many links navigate, the depth, types of files to ignore, etc.).
- Scraping: a scraper takes Web resources (documents, images, etc.), and engages a process for extracting data from those resources, finalized to data storage for subsequent elaboration purposes.
- Indexing/Searching: searching operations on a huge amount of data can be very slow, so it is necessary (through crawler) to index contents.

Once these phases are completed it is possible to analyze the indexed contents in order to extract relevant information from the collected data and produce the needed outputs.

Figure 1 – *Web information extraction process*



2.2. Official Statistics Projects

In Section 1, we cited the National project concerning the use of Internet as a data source for the "Survey on ICT usage in Enterprises". This case can be classified as a case of generic Web scraping. Indeed, on one side the number of websites that was necessary to scrape for the purpose of this project was consistent (more than 8000), while on the other side it was not possible to rely on a fixed and known in advance structure of the websites (Barcaroli et al., 2015-a) (Barcaroli et al., 2016).

Further projects at National level were carried out within CBS, the Dutch National Statistical Office, as well as by Istat: these project are rather examples of specific Web scraping. In particular, in (Ten Bosh et al., 2014) a first domain of experimentation was related to air tickets: the prices of air tickets were collected daily by Internet robots, developed by Statistics Netherlands supported by two external companies, and the results were stored for several months. The experiment showed that there was a common trend between the ticket prices collected by robots and existing manual collection. Two additional domains of experimentation were Dutch property market and Clothes prices, the first exhibiting more regularity in the sites structure, the latter more challenging with respect to automatic classification due to lack of a standard naming of the items, and variability in the sites organization. Similarly, in Italy a scraping activity was performed to get on consumer electronics prices and airfares (Giannini et al., 2014).

More recently, a Eurostat funded project started (ESSnet Big data, 2016) with two work-packages dedicated to Web scraping, namely one to “Web scraping of job vacancy” and another one to “Web scraping of enterprise characteristics”.

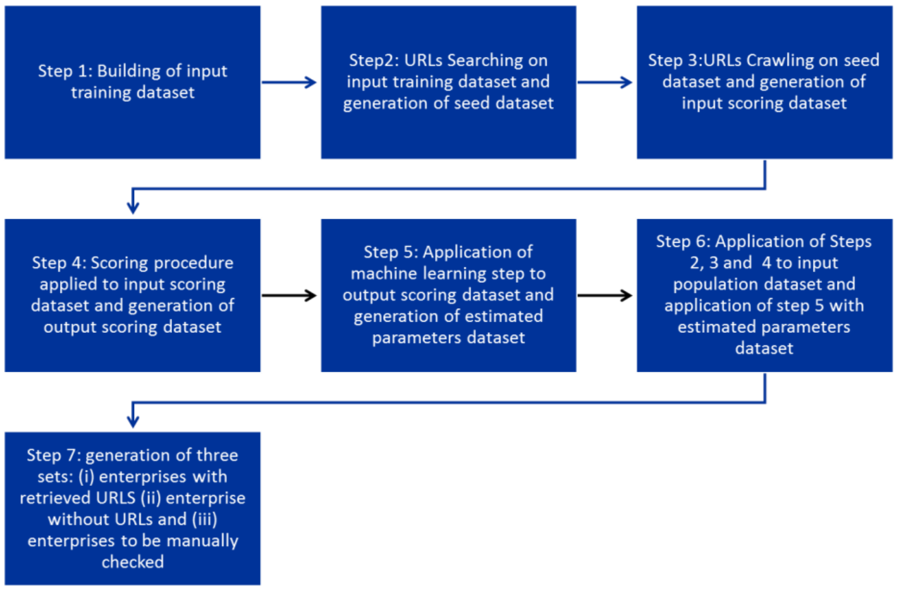
3. The Proposed Strategy

This Section will describe the strategy adopted for solving the URLs retrieval problem. We will start by providing an overview of the approach (Section 3.1), then we will detail the scraping process (Section 3.2, 3.3, 3.4, 3.5), and we later detail the learning process (Section 3.6, 3.7).

3.1. Overview of the Strategy

In order to address the URLs retrieval problem, we developed a step-wise strategy shown in Figure 2.

Figure 2 – *Steps of the overall strategy*


Step 1: Building of input training dataset
Step2: URLs Searching on input training dataset and generation of seed dataset
Step 3:URLs Crawling on seed dataset and generation of input scoring dataset
Step 4: Scoring procedure applied to input scoring dataset and generation of output scoring dataset
Step 5: Application of machine learning step to output scoring dataset and generation of estimated parameters dataset
Step 6: Application of Steps 2, 3 and 4 to input population dataset and application of step 5 with estimated parameters dataset
Step 7: generation of three sets: (i) enterprises with retrieved URLS (ii) enterprise without URLs and (iii) enterprises to be manually checked

In the following sections we will detail each step and we will provide a concrete running example of the strategy as applied to solve the URLs retrieval problem in the case of the survey “ICT Usage in enterprises”.

3.2. Step 1: Building the input training dataset

In this step, the input training dataset is built by considering the units for which the URLs are known as well as the specific features of such units that could be useful to identify them on the Web.

In our case study, we built such a dataset by integrating different sources, namely: third party information (Consodata) and “ICT in enterprises” survey editions of 2013, 2014, 2015. The resulting dataset was composed by 81912 enterprises with URLs.

3.3. Step 2: URLs Searching

In this step, the objective is to retrieve for each unit in the input training dataset, one or more URLs to scrape based on identifying information that is present in such a dataset.

In our case study, we decided to set up an automated procedure that used “enterprises’ denominations” (in Italian “Ragioni Sociali”) for searching the enterprise on the Internet via a search engine. In particular, we used the denomination of the enterprise as a search string, then we queried the search engine and collected the first ten links returned as the result of the search query.

Each of the link was visited in order to estimate the official one. These operations might look very simple if done by humans, but they are indeed very time consuming especially if you have to deal with a huge number of enterprises. Hence, we decided to automate this step by writing a custom Java program that takes as input two files containing the list of the enterprises’ names and the corresponding list of enterprises’ IDs. For each enterprise the program queries a search engine (we are currently using Bing search engine) and retrieves the list of the first ten URLs provided by the search engine.

These URLs are then stored in a txt file, thus having one txt file for each enterprise. At the end, the program reads each produced file and creates the seed dataset, containing the following fields: URL_retrieved, enterprise_Id and position_of_the_URL. Basically we can say that the seed dataset contains the list of potential enterprise official URLs to be scraped.

When we had to choose the search engine to use we tested Google, Istella, Bing and Yahoo in order to determine the most suitable for our objectives. We tried to execute a batch of 10000 automatic queries on each of them, Google and Yahoo blocked us after about 1000 queries so we had to discard them. We focused on Bing and Istella, in both cases we considered about 7000 enterprises with a known site; for each enterprise name we collected the first 10 results from the search engine and compared them with the known site. Using Istella we obtained 50% of matches and not always in the first positions, using Bing we obtained a success rate of 65% so we chose it due to the better obtained results.

3.4. Step 3: URLs Crawling

In this phase the objective is to retrieve for each potential URL of the seed dataset the textual content of the corresponding page. We tested different software solutions for crawling already available off the shelf but they did not satisfy completely our needs, in particular they were not as flexible as we needed (Barcaroli et al., 2015-b).

We decided to develop a custom Java program based on crawler4j (https://github.com/yasserg/crawler4j) and jsoup (https://jsoup.org/). The program takes as input 3 files:

- the seed file produced in the search phase
- a list of URL domains to filter out (directories domains)
- a configuration file

For each row of the seed file, if the URL is not in the list of the domains to filter out, the program tries to acquire the HTML content of the page. From each acquired HTML page the program extracts just the textual content of the HTML fields we are interested in and write a line in a TSV file.

The TSV file just produced is then indexed and loaded in SOLR (http://lucene.apache.org/solr/), an open source enterprise search platform built on top of Apache Lucene.

The resulting document base constitutes the input scoring dataset, to be passed to the following step.

3.5. Step 4: URLs Scoring

In this step, for each document of the input scoring dataset, a score vector is computed and a score is assigned.

For our case study, the resulting dataset, i.e. the output scoring dataset, contains the following fields: *enterpriseId, linkPosition, URL, scoreVector, score.*

In the rest of this section we detail the procedure to compute the score, named URLScorer (see **Errore. L'origine riferimento non è stata trovata.**).

Starting from the input scoring dataset, for each link URLScorer generates a vector of numbers. Every position in the vector contains a number that means “a specific characteristic in the Web page was found or not” (e.g. presence of the telephone number).

For our case study, the elements/characteristics that we considered are the following:

- Simple URL (is the URL in the form www.name.com or not ?)
- VAT (is it present in the page or not ?)
- city (is it present in the page or not ?)
- province code (e.g. NA, is it present in the page or not ?)
- link position (from 0 that means that that was the first link provided by the search engine to 9)
- telephone number (is it present in the page or not ?)
- zip code (is it present in the page or not ?)

For each element/characteristic we computed the confusion matrix obtained by using just that element for classification purposes. Based on that confusion matrix we computed the standard performance measures, i.e. precision, recall and f-measure. Then, we assigned the f-measures of the corresponding confusion matrixes as raw weights to elements; lastly we normalized the raw weights so that the sum of the final (adjusted) weights is 1000 (i.e. normalized weights in **Errore. L'origine riferimento non è stata trovata.**). In order to assign a final score to a link, we summed up the normalized weights of those elements that were actually present in the vector. In a nutshell, we multiplied each number of the vector with a specific coefficient and summed up all the results in order to obtain a score for that link.

To validate the results of URL Scorer, we performed a validation task as follows:

- we selected 20000 enterprises with a known website;
- we compared the URL estimated by the procedure with the highest score with the official URL that we already knew. In particular, comparing only the domain part of the URL (e.g. "rossi.it" if the URL is "www.rossi.it/aboutUs") we found exact matches in 64% of the cases.

The real success percentage is probably higher than the one obtained because sometimes the official Web site that we know is not correct due to several reasons, including:

- it is outdated (the URL is changed);
- it does not exist anymore;
- wrong domain (e.g. “rossi.it” instead of “rossi.com”);
- it is only an information page with contacts (e.g. enterprise contact portals like Italian “paginegialle.it”)

- it is the site that sells the products of the enterprise (e.g. enterprise e-commerce portals, like Italian “mediaworld.it”)
- it is the site of the mother company (franchising enterprises).

In any case, we chose to improve such a result by performing the machine learning step detailed in the following section.

Figure 3 – *Pseudo-code of URLScorer*

```
Input: the textual content of the crawled URLs, a file containing
       enterprises information
Output: Output Scoring Dataset with fields: enterpriseId, linkPosition,
            URL, scoreVector, score

Begin

   normalized weight of telephone
   normalized weight of simple url
   normalized weight of link position
   normalized weight of VAT
   normalized weight of municipality
   normalized weight of province
   normalized weight of ZIP code

   foreach scrapedURL

      load enterprise information

      If (scrapedURL contains telephoneNumber) Then
            Vector[0]=2
      Else
            Vector[0]=1

      If (scrapedURL has simple form) Then
            Vector[1]=1
      Else
            Vector[1]=0

      Vector[2] = (link position - 1)

      If (scrapedURL contains VAT number) Then
            Vector[3]=1
      Else
            Vector[3]=0

      If (scrapedURL contains municipality) Then
            Vector[4]=1
      Else
            Vector[4]=0

      If (scrapedURL contains province) Then
            Vector[5]=1
      Else
            Vector[5]=0

      If (scrapedURL contains ZIP code) Then
            Vector[6]=1
      Else
            Vector[6]=0

      score = 0
      If (Vector[0] = 2) Then
            score = score + normalized weight of telephone
      If (Vector[1] = 1) Then
            score = score + normalized weight of simple url
      If (Vector[2] = 0 OR Vector[2] = 1) Then
            score = score+normalized weight of link position
      If (Vector[3] = 1) Then
            score = score + normalized weight of VAT
      If (Vector[4] = 1) Then
            score = score+ normalized weight of municipality
      If (Vector[5] = 1) Then
            score = score + normalized weight of province
      If (Vector[6] = 1) Then
            score = score + normalized weight of ZIP code

      Write URL line in the output scoring dataset

   end
```

3.6. Using a Machine Learning approach to associate URLs to enterprises

The scoring procedure implemented by URL Scorer produces, for each link associated to a given enterprise, a scoring vector, in which each single position indicates success or failure of the search of a particular information in the website indicated by the link, plus a total score derived from the scoring vector.

A natural choice to select a link as the correct one for the given enterprise could be the one with the maximum score. Unfortunately, this choice would not prevent to choose a non-correct link. In fact, consider that:

- about one third of enterprises do not own a website: for those enterprises, all the links obtained by the search engine are not correct by definition;
- the correct website might not be included in the set of the first 10 links for a number of reasons.

For these reasons, we have to define a more complex decision rule.

Taking into account that a subset of enterprises for which the correct link is already indicated is available, it is possible to adopt a *machine learning* approach under which a model is fitted in this "training" set, and then applied to the set represented by all other enterprises.

In our case study, our <u>input training dataset</u> consisted of 81912, of which 73006 records had at least one page fetched. On the basis of the <u>output scoring dataset</u> we first associated to each enterprise of the 73006 sized set the link with the highest score. As we know if the link is correct or not, a dichotomous variable *correct_Yes_No* says if the URL is the right one or not: this variable plays the role of the Y variable, to be predicted by the model. Together with this information, variables indicating success or failure of the search of *telephone*, *VAT code*, *municipality*, *province* and *zip code* play the role of the X variables (predictors), together with the *link position* and coincidence of the central part of the URL with the name of the enterprise (*simple URL*).

This initial set is split into two equal size subsets, the first acting as the proper training set to fit the model, the second as the test set used to evaluate the performance of the model.

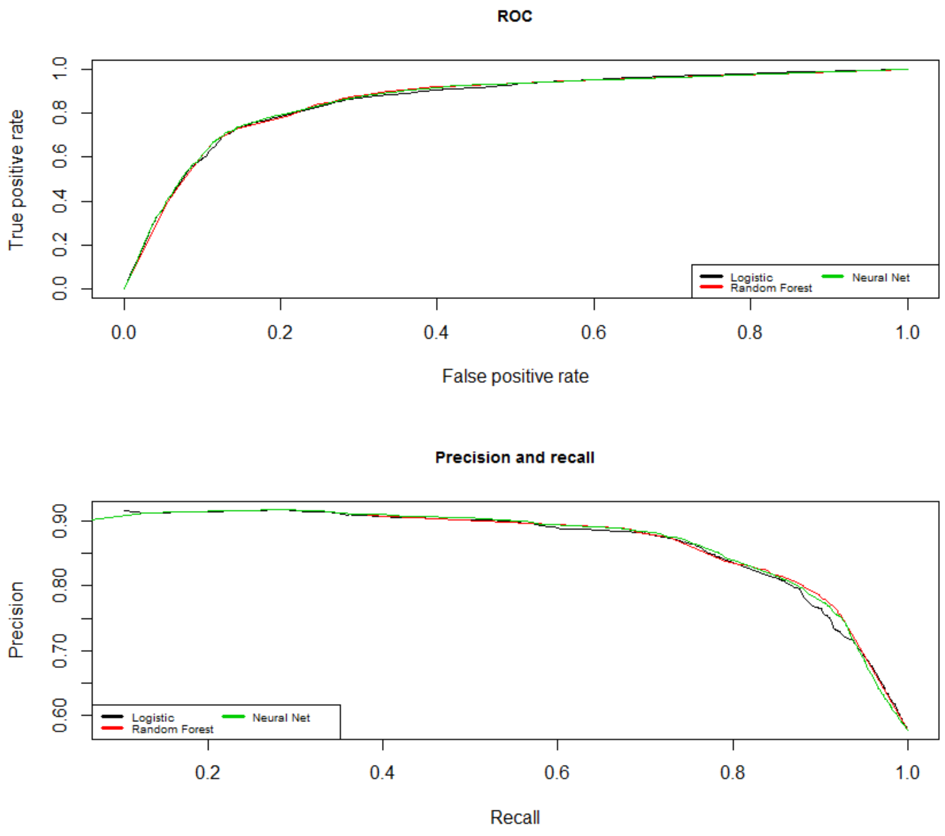
Table 1 – Evaluation of Neural Networks, Random Forest and Logistic Model.

Learner	Accuracy	Sensitivity	Specificity	F-measure
Neural Networks	0.7960	0.8011	0.7890	0.8194
Random Forest	0.7999	0.8278	0.7616	0.8270
Logistic Model	0.7918	0.7857	0.8002	0.8135

Different learners have been fitted and evaluated, namely Neural Networks, Random Forest and Logistic Model. Their performance has been evaluated by considering the classic indicators, that is accuracy, sensitivity, specificity and F-measure (harmonic mean of recall and precision). Their values are reported in Table 1.

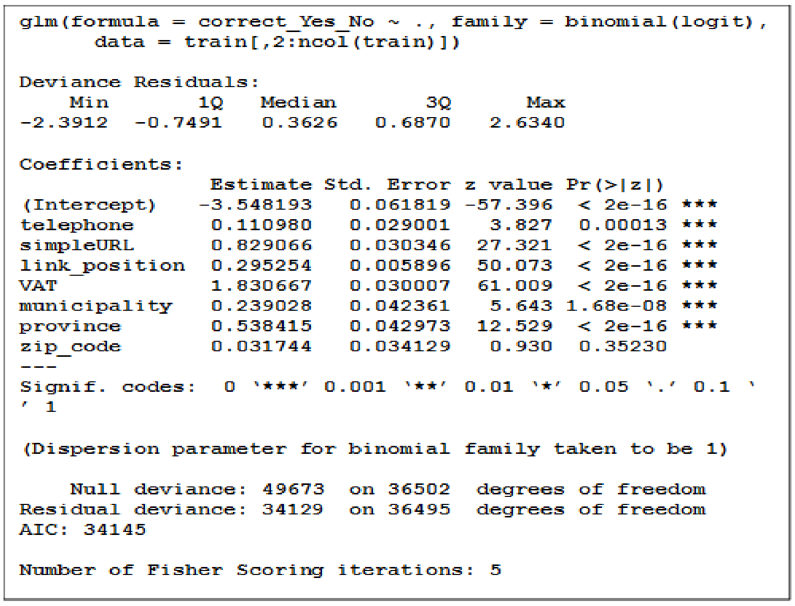
The difference in performance is not significantly different for the three learners, this can be seen also visualizing the ROC and the curves of precision/recall in Figure 4.

Figure 4 – *Performance of the three learners*



Taking into account the statistical properties of the logistic model, this learner has therefore been preferred to the others, also because of the interpretation of the score as a probability. In Figure , the fitting results of the logistic model applied to the training set are shown.

Figure 5 – *Logistic model fitting*



```
glm(formula = correct_Yes_No ~ ., family = binomial(logit),
      data = train[,2:ncol(train)])

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.3912  -0.7491   0.3626   0.6870   2.6340

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)  -3.548193   0.061819 -57.396  < 2e-16 ***
telephone     0.110980   0.029001   3.827  0.00013 ***
simpleURL     0.829066   0.030346  27.321  < 2e-16 ***
link_position 0.295254   0.005896  50.073  < 2e-16 ***
VAT           1.830667   0.030007  61.009  < 2e-16 ***
municipality  0.239028   0.042361   5.643 1.68e-08 ***
province      0.538415   0.042973  12.529  < 2e-16 ***
zip_code      0.031744   0.034129   0.930  0.35230
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 '
' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 49673  on 36502  degrees of freedom
Residual deviance: 34129  on 36495  degrees of freedom
AIC: 34145

Number of Fisher Scoring iterations: 5
```

Once applied to the test set, units have been sorted in ascending order with respect to the score assigned by the logistic model, and have been grouped in 10 balanced classes (Table 2).

Table 2 – *Class of units by their scores*

Group	scores	True positives	False positives	Error rate
1	[0.0312,0.124]	440	3239	0.88
2	(0.124,0.254]	628	3312	0.88
3	(0.254,0.369]	859	2569	0.75
4	(0.369,0.507]	1473	2290	0.61
5	(0.507,0.573]	1895	1555	0.45
6	(0.573,0.725]	3038	830	0.21
7	(0.725,0.862]	2958	561	0.16
8	(0.862,0.921]	3188	428	0.12
9	(0.921,0.936]	1397	141	0.09
10	(0.936,0.943]	5222	480	0.08

By taking all the links in a given class, the error rate depends on the number of false positives in that class. It is clear that the error rate decreases as the score (i.e.

the probability of correct link) increases. If the *acceptation* threshold value is set to of 0.573 as the one to decide if a link is correct or not, the upper five classes are accepted *in toto* and the mean error that can be expected is 0.13, and the total recall is 0.75. In other words, 75% of correct links can be found, together with 13% or erroneous ones.

If the *refusal* threshold value is set to 0.507, the lower five classes are discarded, losing in this 23% of correct links.

It is possible to choose the 5th class (containing about 9.5% of cases), where true and false positives are balanced, as the one to be controlled interactively (for instance, by adopting a *crowdsourcing* platform that is one of our planned future steps).

3.7. Step 6: application of the whole procedure to the input population and Step 7: generation of the results

Once learned the model in Step 5, this step consists of the application of steps 2,3,4 and 5 to the whole input population.

For our case study, i.e. the survey "ICT usage in enterprises", the population of interest of the survey is composed by enterprises with at least 10 employees and operating in different branches of industry and services, the size of such a population is around 200000. By the ICT survey estimates, it is known that about 70% of these enterprises do own a website, used for different purposes.

As described in Section 3.6, starting from our input train set (specifically the part of it with at least one page fetched, of size 73003) a logistic model has been fitted, and threshold values chosen, to be used in order to find additional URLs for remaining enterprise websites.

So, on the complementary set of enterprises for which the URLs are not available, the three steps procedure (searching, crawling and scoring) has been applied. Here, we report the results of the application of the logistic model to the 106019 enterprises for which URLs were not available (i.e. 205759-73006=132753, of these at least one link has been crawled for 106019):

- 26097 (24.6%) URLs have been evaluated as reliable, and associated to the corresponding enterprises;
- 68885 (64.9%) have been considered as erroneous, and excluded;
- 11037 (10.4%) have been addressed to interactive controls.

This latter component is expected to contain about 45% of false positive (see Table 2, the 5th class). So, after the controls about 55% of the 11037 URLs, let us say 6000, should be individuated as correct.

In conclusion, at the end of the process we should obtain a total number of identified URLs equal to about 105000. If we consider a total amount of 130,000 websites pertaining to the enterprises population, we obtain a coverage of near 81%, which can be deemed a satisfactory one.

4. URLs Searching and Crawling Performance

We set up a testing environment with the characteristics shown in Table 3. Let us notice that, being the task massive and resource consuming, this environment is quite under-sized (in terms of RAM and CPU).

Table 3 – *Environment configuration*

CPU	2 cores running at 2.2 GHz
RAM	16 GB
O.S.	Red Hat Enterprise Linux 7
Kernel version	3.10.0-327.22.2.el7.x86_64

Table 3 shows the performance for our running case study.

The execution time of searching and crawling programs takes several hours: this means that explicitly programmatic control to manage failures have been designed and developed in order to manage this long-running feature and get at a result. In terms of dimension of the generated files, being it several Giga bytes it was necessary to adopt a dedicated storage platform (Apache SOLR), as anticipated in Section 3.4. The usage of this platform permitted an indexed access to the generated document base.

5. Conclusions and Future Work

In this paper, we showed an overall strategy for solving the URLs retrieval problem, i.e. the problem of finding a URL corresponding to a statistical unit we are interested to. We detailed the strategy for the case of retrieving URLs of enterprises, with respect to a specific survey run at National and European level, i.e. the survey “ICT usage in enterprises”. The strategy involved several steps with a mix of techniques, ranging from scraping and crawling techniques to machine learning ones. Our results show the feasibility of addressing this problem with a partially automated solution that gets good results both in terms of quality and efficiency.

We have planned as future work to adopt a crowdsourcing platform to manage the interactive task related to enterprises we were not able to manage automatically. Indeed, in our case study around ten thousands enterprises required such a manual effort, which is quite a huge number. We are currently evaluating Crowdsearcher (http://crowdsearcher.search-computing.it/) as a platform for this task.

Table 4 – *Performances*

	TrainSet	TestSet
# of enterprises	81912	132753
UrlSearcher execution time	14h 3min	22h 17min
# urls in seed file	814577	1321323
UrlCrawler execution time	8h 39min	13h 4min
# urls filtered out	470039	846052
# urls after filter	344538	475271
# urls reached	241202	305488
% of reached urls	70.01	64.27
# of enterprises found	76976	117998
# of enterprises with 0 pages fetched	3970	11979
# of enterprises with at least 1 page fetched	73006	106019
Output CSV file size	8.6 GB	10.1 GB

References

BARCAROLI G., NURRA A., SALAMONE S., SCANNAPIECO M., SCARNÒ M., SUMMA D. 2015-a. Internet as Data Source in the Istat Survey on ICT in Enterprises. *Austrian Journal of Statistics*, Vol. 44, pp. 31-43.

BARCAROLI G., SCANNAPIECO M., SUMMA D., SCARNÒ M. 2015-b. Using Internet as a Data Source for Official Statistics: a Comparative Analysis of Web Scraping Technologies. Available at: http://www.websm.org/uploadi/editor/doc/1437484121Barcaroli-etal_WebScraping_Final_unblinded.pdf

BARCAROLI G., BIANCHI G., BRUNI R., NURRA A., SALAMONE S., SCARNÒ M. 2016. *Machine learning and statistical inference: the case of Istat survey on ICT*. 48th Scientific Meeting of the Italian Statistical Society. Proceedings ISBN: 9788861970618

BERESEWICZ M. E. 2015. On representativeness of internet data sources for real estate market in Poland. *Austrian Journal of Statistics*, Vol. 44, N.2, pp.45–57.

BUELENS B., DAAS P. J. H., BURGER J., PUTS M., VAN DEN BRAKEL J. 2014. Selectivity of Big Data. Available at: https://www.cbs.nl/nl-nl/achtergrond/2014/14/selectivityof-big-data.

CAVALLO A. 2013. Online and official price indexes: Measuring Argentina's inflation. *Journal of Monetary Economics*, Vol.60, N.2, pp.152–165.

CITRO C. F. 2014. From multiple modes for surveys to multiple data sources for estimates. *Survey Methodology*, Vol.40, N.2, pp.137–161.

DAAS P., ROOS M., VAN DE VEN M., NERONI J. 2012. Twitter as a potential data source for statistics. *Discussion paper (201221)*. Central Bureau of Statistics. Den Haag/Heerlen, 2012

ESSNET BIG DATA 2016. Eurostat funded Project – Essnet on Big Data, https://webgate.ec.europa.eu/fpfis/mwikis/essnetbigdata/index.php/ESSnet_Big_Data

FASULO A., D'ALÒ M., FALORSI S. 2015. Provisional estimates of monthly unemployment rate using Google Trend. *Proceedings of ITACOSM*. Rome, June 2015.

GIANNINI R., LO CONTE R., MOSCA S., POLIDORO F., ROSSETTI F. 2014. Web scraping techniques to collect data on consumer electronics and airfares for Italian. *HICP compilation*, Q2014 European Conference on quality in official statistics Vienna, 2-5 June 2014.

TEN BOSH O., WINDMEIJER D. 2014. On the Use of Internet Robots for Official Statistics. In MSIS-2014. URL:http://www.unece.org/fileadmin/DAM/stats/documents/ece/ces/ge.50/2014/Topic_3_NL.pdf.

SUMMARY

On the Use of Internet as a Data Source for Official Statistics: a Strategy for Identifying Enterprises on the Web

In this paper, we propose an overall strategy for solving the URLs retrieval problem, i.e. the problem of finding a URL corresponding to a statistical unit we are interested to. We detail the strategy for the case of retrieving URLs of enterprises, with respect to a specific survey run at National and European level, i.e. the survey "ICT usage in enterprises". The strategy involves several steps with a mix of techniques, ranging from scraping and crawling techniques to machine learning ones. Our results show the feasibility of addressing this problem with a partially automated solution that gets good results both in terms of quality and efficiency.

Giulio BARCAROLI, Istat, barcarol@istat.it
Monica SCANNAPIECO, Istat, scannapi@istat.it
Donato SUMMA, Istat, donato.summa@istat.it