Report: Enterprise Web sites

Abstract

The aim of the project Entreprise Websites was creation of the IT tool for scraping data from enterprise websites and using it in order to create useful statistics. The team which worked on this pilot consisted of representatives of NSIs of Netherlands, Poland, Sweden and Slovenia. The initial idea was construction of the process of collecting the URL links of enterprises, creating IT the tool for collecting desired data and testing the methodology of creating the statistics out of collected data. To be more precise the team has focused on web data which are related to job vacancy (JV) advertisements. This project was very risky due to the fact that in comparison with the other teams we needed to establish an IT tool for the collection of the data, and this task is extremely difficult. The second drawback of the project is that people were volunteering their time to work on the project. This means that they had limited time. Our main goal was to create statistics on JV from enterprise websites (excluding employment agencies). This document presents all the work which has been done during 2015 together with proposals for future work.

Description

In reality we would like to collect data from administrative sources and from specialised enterprises and scrape the data from the single enterprises. Then we would like to integrate data from different sources and create statistics (e.g. flash estimates of totals, totals broken down by NACE groups, totals broken down by regions, ratios with information about investments, turnover, etc.).

The original plan was to take the NUTCH software which was installed in the Sandbox, and use the custom-made program from our Italian colleagues to collect the data and focus on statistics. The biggest problem we encountered was getting lists of company URLs. Most of the countries don't have lists or the quality is not sufficient. The initial plan was to try to collect the list of enterprises (for this purpose, the enterprise is defined as being equivalent to the Legal Unit - LUs) then scrape the data (not only about JV, but also about emails, addresses,...) and then create the statistics (only from this source). The idea was the following: Use the names of LUs from the Business Register, type them in the Bing Search Engine (Google would block us), collect first ten hits (URLs), scrape all the content from the each page and score every page with special algorithm. The page (URL) with the highest score would be determined as an official page of company. But we had huge problems with compatibility of different versions of NUTCH and JAVA so we decided to give up on this option. Instead we decided to employ Python software and write the program from scratch. We skipped the first step using Bing and started only with available lists of URLs of companies, which served as a sample. Later we might continue to collect URLs of all companies from business registers. In the case of Netherlands and Slovenia (and Poland) we have samples of URLs of LUs in our business registers. In the case of Sweden they received a couple of thousand of URLs together with ads from employment agencies. We limited the desired data to JV only.

Most of the group will continue with the project after the Sandbox project finishes.

Data characteristics

EXPERIMENT:

For the aim of testing the availability of URLs, performance of the prototype of IT solution and creating statistics, the team has focused on sample data from a Slovenian ICT survey (carried out in 2015).

COMMUNITY SURVEY ON ICT (Slovenia)
Sampling unit: Enterprise
Scope / Target Population:
Economic activity:
Enterprises classified in the following categories of NACE Rev. 2:
- Section C – "Manufacturing";
- Section D, E – "Electricity, gas, steam and air conditioning supply"
"Water supply, sewerage, waste management and remediation activities";
- Section F – "Construction";
Section G – "Wholesale and retail trade; repair of motor vehicles and motorcycles";
- Section H – "Transportation and storage";
- Section I – "Accommodation and food service activities";
- Section J – "Information and communication";
- Section L – "Real estate activities";
- Division 69 - 74 – "Professional, scientific and technical activities";
- Section N – "Administrative and support service activities";
- Group 95.1 – "Repair of computers and communication equipment"
Enterprise size:
Enterprises with 10 or more persons employed.
Geographic scope:
Enterprises located in any part of the territory of the country.

Sample design: Stratified random sample (stratification according to NACE activity and size of units)

Frame size:	6670
Sample size:	1806
Number of responses:	1531
Number of units with URL link:	1335
Estimated nuber of units with URLs (sum of weights):	5375
Percentage of enterprises with websites :	**81%**

From the data reported by respondents, it is estimated that in the population of enterprises with at least 10 employees, around 80% have websites. We can assume that the percentage of enterprises with less than 10 employees which have URLs is lower, but that the percentage of such enterprises that are looking for new employees is also lower.

Web sites of 1335 units that reported URLs were used as an input for testing the prototype of the IT Web scraping system for identifying job advertisements. On the basis of the initial testing of a small sample of units, with the assumption that there are 20 URL links per page and that the program needs 0.1 second for checking each URL, we estimated that on average 13 minutes is needed for processing all links for one enterprise. Therefore, the 1335 units were divided into 10 groups of 130 units and all groups were parallel processed. The time for processing each group ranged from 1 to 8 days (there were units with extremely large numbers of sub URLs). Most of the time was spent for checking the URLs of enterprises in order to detect the set of URLs related to job vacancy advertisements.

After the prototype of the IT tool (Spider) finished processing the data, a sub-sample of 100 units was chosen and manually examined in order to get information about the precision of the tool.

14 enterprises out of 100 manually examined units had advertisements for job vacancies. The application found 9 of these units (65%) as well as 3 units which in reality didn't have job vacancy advertisements.

The following are the reasons, why the application didn't find the other 5 companies:

- Employment URLs don't contain general employment keywords but contain specific profile keywords, so the solution is to use profile keywords in the Spider already.
- Spider didn't find employment URLs because they contained specific employment keywords, which were not in the whitelist at that time and have now been added.
- Spider didn't find employment URL because of JavaScript

- HTTP 403 error, the server doesn't answer to Spider's request. The solution is perhaps a proxy server.
- Unicode error in Spider: It has been repaired.

The following are the reasons why the application found 3 companies by accident:

- Two non-existing job vacancies were found in the media news due to appearance of keywords.
- One non-existing job vacancy was found due to appearance of keywords.
- By accident, a web formula was detected as a job vacancy. The web formula was an invitation to visitors to register in the database of potential candidates for employment.

Additionally, it was found that around 30% of units don't have sub URLs that contain job advertisements. We can assume that those units use other channels (e.g. job portals) for advertising JV.

Activities

1. Task: Testing the possibilities of collecting of URLs of enterprises

Due to the fact that in most of the countries the URL list for enterprises doesn't exist, the team explored the possibilities of scraping the desired links. The requirement for development of this solution was a list of enterprises from the business register. The first solution that was being developed between April and August 2015 was using software Nutch + Lucene + Solr + Java. This solution started with company names, and produced URLs of the companies. That part was mainly prepared by Istat. The solution collected the first 10 hits for the name of the company in the Bing search engine and predicted which one was most likely to be the official URL of the company. The prediction was done with a custom made scoring algorithm.

However, we abandoned that solution due to difficulties with compatibility of different versions of Nutch and Java. Perhaps it was also too advanced to use such software for experimenting while we still didn't know how to detect job vacancies.

2. Task: Investigation of existence of URLs at participating NSIs

The best coverage of existing URLs by a NSI was found in Netherlands where one third of population is covered. In Slovenia the coverage is not so good, but in the ICT survey, all sampled units were obliged to report their URLs (if they have any) .

Statistics Slovenia found that approximately 20% of enterprises don't have their own websites.

The following table contains the current state of the input data (enterprise list), specifying if names and URLs are available and the size of the list (number of enterprises included).

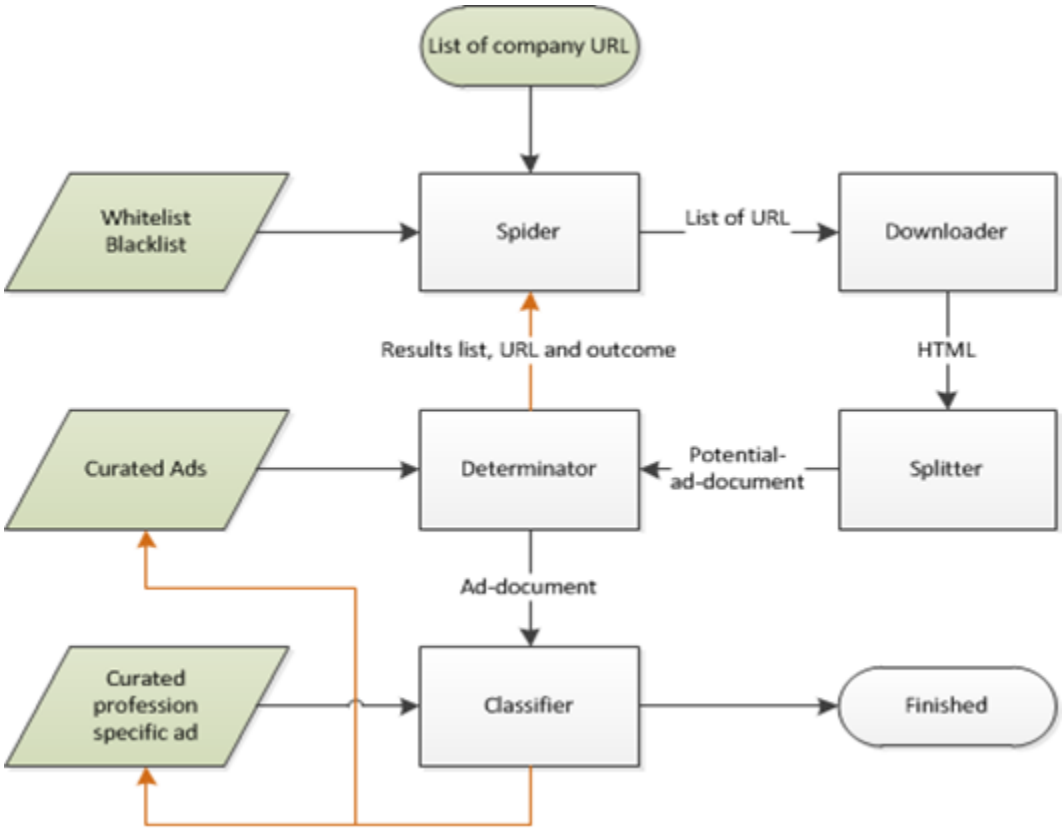
Country	List of names (yes/no)	List of enterprises (yes/no)	Number of enterprises
Italy			
Netherlands	Yes	Yes	482,458 with URL (from a total of 1,509,407 enterprises)
Slovenia	Yes	Yes	15,000
Sweden	Yes	No URLs	13,000
Poland			

3. Task: Creating the application for scraping the JV data from URL of enterprises

Assuming the availability of URLs of enterprises, the JV team focused on creating the prototype IT solution for collecting the data about JVs. The basic idea of creating the IT application was:

- Input (URL link of enterprises)
- Detection of sub URLs which are related to JV ads
- Downloading the content of detected URLs
- Splitting the content into separate documents (e.g. part of content which is related to JV in one document, other parts in separated documents)
- Detection of JV ads in the documents related to JV
- Classifying the detected JV ads (occupation, skills,..)
- Output (Id of enterprise, URL link, number of detected JV ads)

Web scraping system for identifying job advertisements



So the team prepared a program with 5 blocks. The following is a detailed description of the webscraping system.

- Spider: The aim of the Spider is to take the website of the company and find all webpages on this website, that relate to employment. Spider checks all the links (and names of tabs) up to depth 3 (or 4) and if the link contains some keywords (like employment, career, jobs, we-are-looking-for, we-are-hiring, etc.), then we save it to a database of possible links that may contain JV. For example the spider finds:
  http://outfit7.com/category/jobs/
  The keyword is "jobs" and the link contains JV ads.
  http://www.istrabenz.si/eng/humanresources/employment
  The keyword is "employment" and the link doesn't contain JV ads.
  The input for the Spider is a list of company URLs and a "whitelist" (list of keywords and phrases related to employment).
  Ideally the Spider would be based on a machine learning algorithm, which would improve based on previous experiences.
- Downloader: The task of Downloader is simply to download the content of the saved URL links. For now, we have problems with pdf files and https sites, but we are trying to resolve these issues.
- Splitter: The aim of Splitter is to split the content of a particular URL into documents. For example, we would like to create the documents so that each contains exactly one JV. We know that there can be several JV on one employment webpage, but for now we assume that there is only one JV per URL.
- Determinator: The aim of Determinator is to detect the JV ads in the documents from Splitter. Again we are preparing the Whitelist ("occupation", "deadline for application", "We are currently looking") and Blacklist ("There are currently no job offers", "student work", ...) of phrases and keywords which could help us to determine presence of JV ads. The more advanced approach is to create clouds of words which are associated with JV ads (see the Swedish presentation).
- Classifier: The aim of Classifier is to classify the detected JV for example by occupation, deadline, address, region, etc.). Here we would use The International Standard Classification of Occupations (ISCO) by ILO (International Labour Organisation). We skipped this task for now due to lack of time and we decided only to work on the Determinator. For now it is enough to detect JV, count them and classify them according to the NACE (activity) code of the company.

The solution with Python Scrapy was more successful and we developed it from scratch. The development was going on between August and November 2015. We thought it is best to develop the solution starting from existing samples of URLs from official business registers. We skipped the first step of obtaining URLs from names of the companies. Later, we can expand the solution to total population by using a Java solution for URL collection or we can even rewrite Java to Python. Python uses Scrapy to check all URLs on the official website of the company up to a certain depth and collects those URLs that are related to employment. Namely those URLs that contain some employment keywords (career, employment, work-with-us, join-us, etc.). The selection of keywords is language-dependent and requires manual semantic work. This is the Spider part. The Spider checks a few hundred links per second, which can be duplicated, and a few dozen unique links per second. For the Detector we first used occupation keywords but that was not accurate enough. Instead, we used keywords that generally describe job vacancy (education, type of work, length of contract, salary, location, responsabilites, expectations, requirements, etc.). In a way, we were creating the definition of job vacancy advertisement. That worked better and enabled us to detect job vacancies. We believe the solution is in this semantic approach. We were also thinking about using the HTML structure of webpages to define HTML structures of job vacancies but on the advice of experts, we decided not to go there because HTML structures are too unpredictable. However, there is still a possibility that HTML structures could help us count the number of job vacancies on a webpage if we parse it as a tree of tags. That would be a very advanced solution.

The formula that counts the number of job vacancies on the webpage is a basic one so the next step is to improve it.

The idea for the next project is to use CYC Knowledge Base, which is a great Artificial Intelligence project. CYC Knowledge Base comprises worldwide knowledge and enables human-like reasoning. We could define the job vacancy advertisement, like we started with keywords, and insert the definition into the knowledge base. Then we could reason if a certain webpage contains the job vacancy advertisement. The link to CYC goes through Artificial Intelligence Lab at the Joseph Stefan Institute in Ljubljana.

http://cycorp.eu/about/

Following steps also include:

- heuristics that would allow the spider to jump to similar domains, because sometimes the companies publish their job vacancies on other sites with similar or different domains
- classification of job vacancies found into ILO classification of occupations.

4. Swedish and Slovenian approach on module "Determinator"

Description of the Determinator

A potential advertisement is fed into a program that predicts whether the text is a job advertisement or not. This is done by machine learning; a model is trained with examples of job advertisements and non-job advertisements. When the model has been trained, it can be used for classifying advertisements.

Input to the module is a text document stripped of all HTML markup and navigational text. The first step is to turn the documents into a *document-term matrix.* This is a matrix where the row corresponds to the document and the columns to the terms, i e words. Each term is weighted by its presence in the document, i.e. if the word appears in the document, its weight is 1; otherwise the weight is 0. An illustration of a Document-Term matrix is shown in Table 1.

Table 1. Example of a Document-Term matrix

	Word 1	Word 2	Word 3	Word 4
Document 1	1	0	0	1
Document 2	1	1	0	0
Document 3	0	0	1	1
Document 4	0	1	1	0

To reduce the computational complexity, the words in the matrix are a subset of all the words available in the corpus. The most common words from both the job advertisements and the non-job advertisements are selected.

Frequent words that are commonly used but contribute little to the content are called *stop-words*. Removing these words first will help us identify the real frequently used meaningful words.

The final model has about 800 columns, i.e. it looks for the occurrence of each of 800 words in the documents. The final model gives only the exact match, i.e. either the document is a job advertisement or it is not (binary classification). There is no partial match.

The training methods (classifiers) applied are *Gaussian Naïve Bayes* and *Decision tree*. The Bayesian method calculates the likelihood that a document belongs to a certain class based on the likelihood that the words, independently of each other, belong to the same class. In contrast, the decision tree is dependent on the composition of terms. It will start by looking for a certain term, if it finds the term it will check if a certain other word is present and repeat until it is confident enough to make a prediction.

Binary classification metrics and benchmarks

The output from a binary classification can be described as in Table 2 (confusion matrix) below. The *Prediction* is what the model predicts, the *Observation* is the actual value, what the model is meant to predict. True positives are in our case job advertisements that are correctly classified as such. If they are incorrectly classified as non-job advertisements, they count as False negatives. False positives are non-job advertisements that are incorrectly classified as job advertisements, and True negatives are non-job advertisements that are correctly classified.

Table 2. Confusion matrix

		Observation	
Prediction		**+**	**-**
	+	True positive, TP	False positive, FP
	-	False negative, FN	True negative, TN

There are a number of measures that can be derived from a confusion matrix to ease the interpretation.

$$Precision = \frac{TP}{TP + FP}$$

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

$$Recall = \frac{TP}{TP + FN}$$

Each measure gives insights into how well a model is performing in a certain aspect. *Precision*, the relative number of true positives among all positives, can be seen as a measure of how certain it is that a predicted class is in fact the correct class. *Recall* is a metric of the relative number of positives that are correctly identified.

There are often trade-offs when optimizing a certain measure. Increasing *Recall* means moving False negatives to True positives, this is done by "lowering the bar" for the classification. But this might also mean a higher risk that True negatives are classified as False positives and thus *Precision* will decrease.

Accuracy, the relative number of correctly classified advertisements, can be problematic due to the composition of the populations. If there are many more false than true examples in the population, one could "optimize" the model by predicting all observations as false and get a large number of True negatives. This will yield a high *Accuracy* but the measure will be useless.

Results and discussion of the model

Both scraped data and manually collected data are combined for training and evaluating the Determinator.

First, job advertisements from Arbetsförmedlingen (the Swedish Employment Services) were scraped. To increase the variation of the data, hundreds of job advertisements were collected manually from enterprise web sites. These advertisements are positive data, representing the job advertisement web pages.

The non-job advertisement data are program scraped from the enterprise URLs from Arbetsförmedlingen. Using these URLs, other enterprise web pages that are related to the job advertisements are scraped. Pages such as the presentation of the products and services of the companies and the employees working experiences are considered related. To increase the text variations of the non-job advertisement examples, Wikipedia pages concerning certain professions and companies were scraped. Also a small portion of manually collected web pages of non-job advertisements were added. There are some English web pages included because the process of scraping did not consider the language. Hence, there are two groups of examples of almost equal size: job advertisement pages and the non-job advertisement pages

The test data and the results are presented in the tables below. The dataset was split into two parts, the training set and the validation set. The training set was used to train the model, and the validation to validate the results (Table 3).

Table 3. Training and validation sets

	Training	Validation
Positive	3284	3224
Negative	4177	4238
Total	**7461**	**7462**

The same training and validation data was used for both classifiers, the results are presented in Table 4.

Table 4. Results for both classifiers

	Gaussian Naïve Bayes	Decision tree
TP	2959	3014
TN	4055	3938
FP	183	300
FN	265	210
Precision	94,2%	90,9%
Recall	91,8%	93,5%
Accuracy	94,0%	93,2%
F1	93,0%	92,2%

In general, the scores of the two methods are high, i.e. all the important scores are above 0.9. The means of recall and precision are both above 0.92 for both methods. This means that the determinator can detect about 92% of the job advertisement pages from the validation examples.

Although the scores are good, the model needs to be improved for practical usage. Partly it is because of the huge variation on the Internet. A system used for web scraping on the Internet needs to handle the variation; therefore, more examples need to be tested with the model. The development of the model does not consider the frequency of the terms and other information known to the web pages is not considered. The goal is to push the recall close to 1, which means all the job advertisements can be correctly identified. The precision score should also be relatively high, at least 90%. In other words only false positives exist; the number of false negatives should be zero.

The execution speed of the model is reasonable for the ordinary amount of pages. We calculate the ordinary amount by 1500 enterprises. For each enterprise, 20 to 50 web pages are expected to be extracted as the input data for this model. The ideal situation is that all pages of job advertisements are covered with some non-advertisement pages included. Therefore the ordinary amount is 30 000 to 75 000 web pages.

Future Work

In the future, the model can be improved by weighting in the frequency of the representing terms or even the document length, i.e., the term frequency (TF) scheme and TF-IDF scheme. IDF stands for inverse document frequency.

N-gram is another alternative to try to improve the model. In the *n*-gram model, it is assumed that the terms are conditionally presented, i.e., the *n*th term is conditioned on the previous *n*-1 terms.

Testing with other training methods, e.g. support vector machine (SVM), could also be considered. SVM is empirically a very successful method for text mining.[1]

There is also other information e.g. the title of the web pages and the attributes of HTML structure, that can be used. If there is more than one job advertisement, the title of these web pages should be identical. The problem is however that this information does not necessarily exist since not every enterprise follows the standard template.

Another validation measure that can be considered in the future is the F1 measure,

$$F1 = 2\frac{Precision * Recall}{Precision + Recall}$$

F1 is the harmonic mean of *Precision* and *Recall* and is satiable if one wants to compare models using only one metric. This measure can be useful for evaluation when we have developed several machine learning models.

Before the training, the data set can be stemmed. This process takes away the syntactical forms of words, e.g. verb forms and noun forms. This will reduce the number of terms in the dataset, and might enable the model to give good predictions with fewer examples.

5. Task: Statistics

The JV team discussed the possible statistics that could be derived from the collected data. The first possibility was to create the existing statistics that are prescribed by EU Regulations (Number of JV ads broken down by NACE activities). The second possibility was finding out the relationship between the Job Vacancies Ads and desired skills for application. The third was finding out the relationship between regional differences.

In reality with the developed prototype of the IT solution we are able only to detect the JV ads with certain precision. So, the proposed statistics were 'Number of enterprises which advertise JV' broken down by NACE activity (and region).

6. Task: Discovering the IT tools for webscraping at large scale

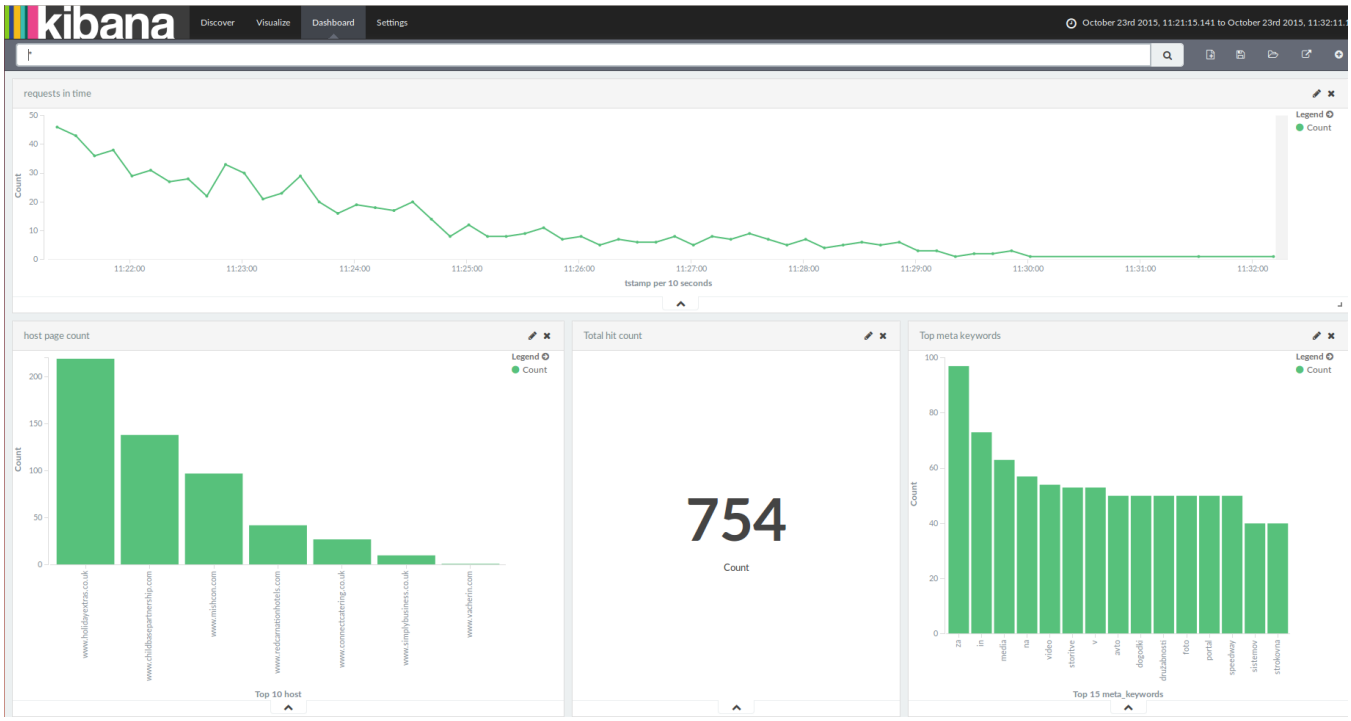
To accomplish the task of enterprise website webscraping on a large scale, there is a need for technology based on parallel processing in a big data infrastructure. During the webscraping process not only the speed and parallelism is important, but also the ability to respect a given website's policy of making multiple requests.

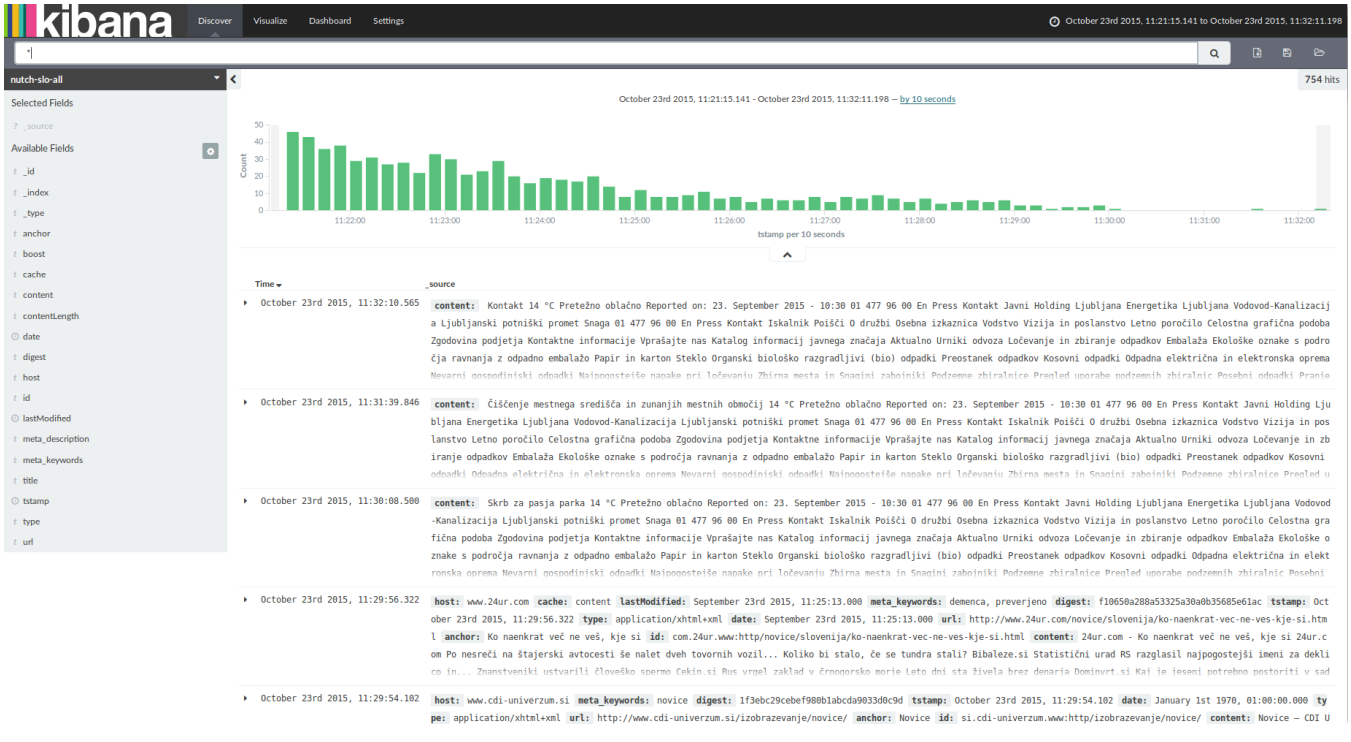
In this context Apache Nutch was selected as a webscraping tool for evaluation. Nutch is an extensible and scalable open source web crawler. It can run on a single machine (multiple threads) but for a production environment it can run in a Hadoop cluster. It also supports many NoSQL storage solutions and indexing architectures for full text search like Apache Solr and Elasticsearch. Nutch can also parse robots.txt which ensures webscraping in good manner by respecting the rules of webscraping of a given website. This lowers the probability of blocking our robot from parsing the same website in repeatable way.

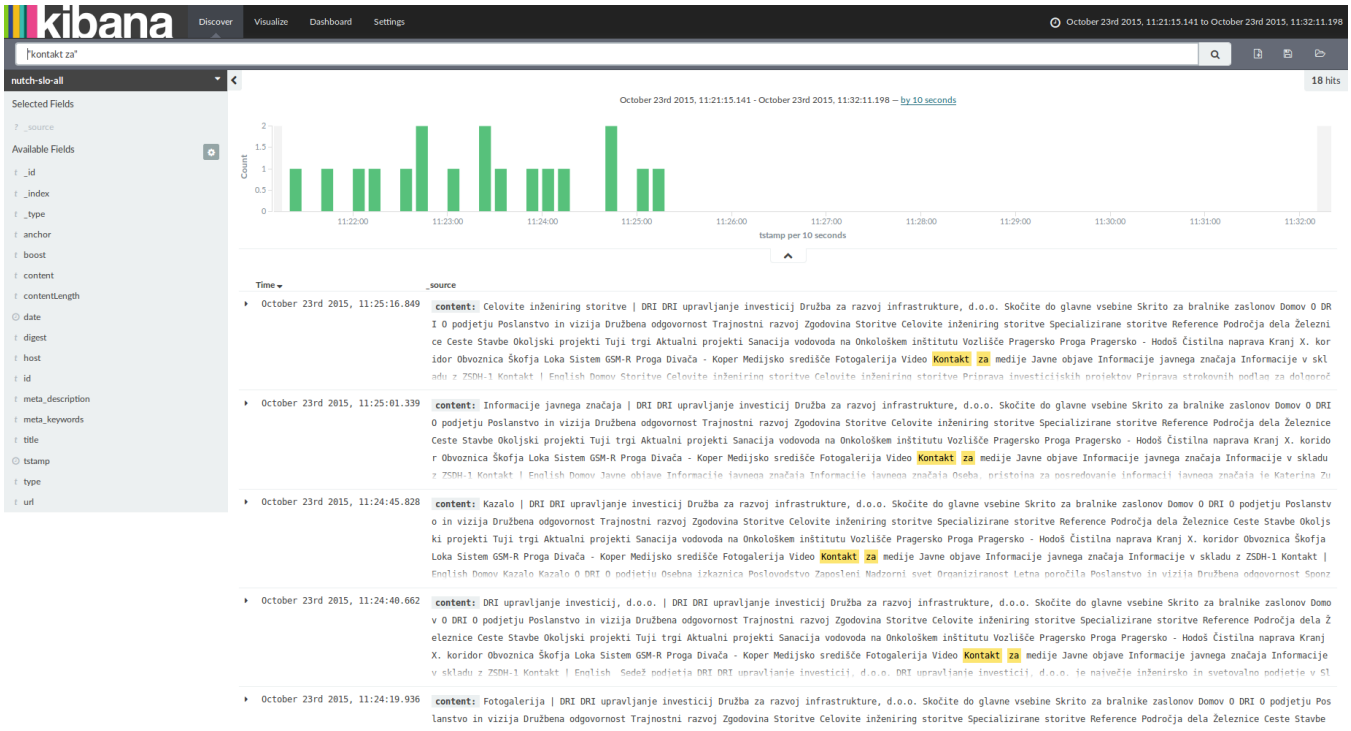
While this solution is good for gathering large amounts of website data, we need a tool for fast exploration of that data. We selected Elasticsearch as a tool for indexing and full text search analysis. It integrates with Apache Nutch and is also capable of distributing through a cluster. It is built on top of a powerful Information Retrieval Library – Apache Lucene and makes its powerful analysis features available. The Elasticsearch platform is extended with a Kibana frontend which provides a feature rich analytics and visualization platform.

Such a stack of tools proved its powerful functionality for getting insight on scraped data and ensured strong potential for further work.

Some screenshots of the Kibana interface with simple dashboard are shown below, using a small dataset of Slovenian websites:


kibana
Discover
Visualize
Dashboard
Settings
October 23rd 2015, 11:21:15.141 to October 23rd 2015, 11:32:11.1
requests in time
Count
tstamp per 10 seconds
Legend
Count
host page count
Top 10 host
www.holidayextras.co.uk
www.childbeepartnership.com
www.mishcon.com
www.redcarnationhotels.com
www.connectcatering.co.uk
www.simplybusiness.co.uk
www.vacherin.com
Total hit count
754
Count
Top meta keywords
Top 15 meta_keywords
za
in
media
na
video
storitve
v
avto
dogodki
druzabnost
foto
portal
speedway
sistemov
strokovna


kibana
Discover
Visualize
Dashboard
Settings
October 23rd 2015, 11:21:15.141 to October 23rd 2015, 11:32:11.198
nutch-slo-all
Selected Fields
_source
Available Fields
_id
_index
_type
anchor
boost
cache
content
contentLength
date
digest
host
id
lastModified
meta_description
meta_keywords
title
tstamp
type
url
754 hits
October 23rd 2015, 11:21:15.141 - October 23rd 2015, 11:32:11.198 — by 10 seconds
Count
tstamp per 10 seconds
Time
_source
October 23rd 2015, 11:32:10.565
content: Kontakt 14 °C Pretežno oblačno Reported on: 23. September 2015 - 10:30 01 477 96 00 En Press Kontakt Javni Holding Ljubljana Energetika Ljubljana Vodovod-Kanalizacij a Ljubljanski potniški promet Snaga 01 477 96 00 En Press Kontakt Iskalnik Poišči O družbi Osebna izkaznica Vodstvo Vizija in poslanstvo Letno poročilo Celostna grafična podoba Zgodovina podjetja Kontaktne informacije Vprašajte nas Katalog informacij javnega značaja Aktualno Urniki odvoza Ločevanje in zbiranje odpadkov Embalaža Ekološke oznake s podro čja ravnanja z odpadno embalažo Papir in karton Steklo Organski biološko razgradljivi (bio) odpadki Preostanek odpadkov Kosovni odpadki Odpadna električna in elektronska oprema Nevarni gospodinjski odpadki Najpogostejše napake pri ločevanju Zbirna mesta in Snagini zaboji
October 23rd 2015, 11:31:39.846
content: Čiščenje mestnega središča in zunanjih mestnih območij 14 °C Pretežno oblačno Reported on: 23. September 2015 - 10:30 01 477 96 00 En Press Kontakt Javni Holding Lju bljana Energetika Ljubljana Vodovod-Kanalizacija Ljubljanski potniški promet Snaga 01 477 96 00 En Press Kontakt Iskalnik Poišči O družbi Osebna izkaznica Vodstvo Vizija in pos lanstvo Letno poročilo Celostna grafična podoba Zgodovina podjetja Kontaktne informacije Vprašajte nas Katalog informacij javnega značaja Aktualno Urniki odvoza Ločevanje in zb iranje odpadkov Embalaža Ekološke oznake s področja ravnanja z odpadno embalažo Papir in karton Steklo Organski biološko razgradljivi (bio) odpadki Preostanek odpadkov Kosovni odpadki Odpadna električna in elektronska oprema Nevarni gospodinjski odpadki Najpogostejše napake pri ločevanju Zbirna mesta in Snagini zaboji
October 23rd 2015, 11:30:08.500
content: Skrb za pasja parka 14 °C Pretežno oblačno Reported on: 23. September 2015 - 10:30 01 477 96 00 En Press Kontakt Javni Holding Ljubljana Energetika Ljubljana Vodovod -Kanalizacija Ljubljanski potniški promet Snaga 01 477 96 00 En Press Kontakt Iskalnik Poišči O družbi Osebna izkaznica Vodstvo Vizija in poslanstvo Letno poročilo Celostna gra fična podoba Zgodovina podjetja Kontaktne informacije Vprašajte nas Katalog informacij javnega značaja Aktualno Urniki odvoza Ločevanje in zbiranje odpadkov Embalaža Ekološke o znake s področja ravnanja z odpadno embalažo Papir in karton Steklo Organski biološko razgradljivi (bio) odpadki Preostanek odpadkov Kosovni odpadki Odpadna električna in elekt ronska oprema Nevarni gospodinjski odpadki Najpogostejše napake pri ločevanju Zbirna mesta in Snagini zaboji Podzemne zbiralnice Pregled uporabe podzemnih zbiralnic Posebni
October 23rd 2015, 11:29:56.322
host: www.24ur.com cache: content lastModified: September 23rd 2015, 11:25:13.000 meta_keywords: demenca, preverjeno digest: f10650a288a53325a30a0b35685e61ac tstamp: Oct ober 23rd 2015, 11:29:56.322 type: application/xhtml+xml date: September 23rd 2015, 11:25:13.000 url: http://www.24ur.com/novice/slovenija/ko-naenkrat-vec-ne-ves-kje-si.htm l anchor: Ko naenkrat več ne veš, kje si id: com.24ur.www:http/novice/slovenija/ko-naenkrat-vec-ne-ves-kje-si.html content: 24ur.com - Ko naenkrat več ne veš, kje si 24ur.c om Po nesreči na štajerski avtocesti še nalet dveh tovornih vozil... Koliko bi stalo, če se tundra stali? Bibaleze.si Statistični urad RS razglasil najpogostejši imeni za dekli co in... Znanstveniki ustvarili človeško spermo Cekin.si Rus vrgel zaklad v črnogorsko morje Leto dni sta živela brez denarja Dominvrt.si Kaj je jeseni potrebno postoriti v sad
October 23rd 2015, 11:29:54.102
host: www.cdi-univerzum.si meta_keywords: novice digest: 1f3ebc29cebef980b1abcda9033d0c9d tstamp: October 23rd 2015, 11:29:54.102 date: January 1st 1970, 01:00:00.000 ty pe: application/xhtml+xml url: http://www.cdi-univerzum.si/izobrazevanje/novice/ anchor: Novice id: si.cdi-univerzum.www:http/izobrazevanje/novice/ content: Novice — CDI U



The challenges that we have come across while developing the prototype of JV application.

1. SPIDER

a) Java Script: The spider inspects the HTML structure of webpages and looks for employment links, that contain specific employment keywords like (career, employment, job vacancy, etc.). This approach works well on static webpages, but dynamic webpages that use Java Script, and are generated at the moment of access, don't offer the spider the possibility to detect employment links.

Example:

http://artdom.si

http://artdom.si/UMETNOST_BIVANJA/Zaposlitev.html

Solution: gather more knowledge about Java Script

b) The speed of Spider: In Slovenia we used parallel processing for the random sample of 1300 companies, so we launched 13 spiders, each of them for 100 companies. Some spiders finished after a day or two and some we running for one week. One spider found 242 links on average.

Solution: Spider uses smarter search like »beam search« or maybe we put it to Sandbox

c) Domain change: We narrow the domains of webpages to the domain of initial company website, otherwise the spider could wander far away. But still it happens that the spider jumps to a new domain like in this case:

https://www.spar.si/sl_SI/splet/spar-slovenija-twitter.html

https://twitter.com/sparslovenija

In this case the spider continues on the »Twitter« domain and starts collecting »Twitter« webpages, which we don't want.

Solution: 1) put Twitter and Facebook on the blacklist of keywords.

2) learn more about domains

d) Protocol change: For example when protocol changes from HTTP to HTTPS like in this case:

http://www.gkn.com/careersacrossgkn/Pages/latest-group-vacancies.aspx

https://careers.gkn.com/

e) Pandora's box. Many strange and long URLs inevitably appear on webpages, so we block links that contain the characters '%' and '~'. For now, we allow the characters '?', '&' and '#' in the link. This is a trade-off question. The more links we allow to get into our net, the more garbage we get, and the more we restrict our net, more employment links can be left outside.

For example, this is an employment link: http://atech.si/index.php?page=105&lang=1, but if we allow question mark '?' in the link, we open the Pandora box for all kinds of document and music links, that will be described in the Downloader section.

Solution: to be discussed.

f) Occupation keywords. Employment link might not contain general employment keywords but it will contain specific occupation words like (analyst, consultant, secretary ..)

Example: http://www.nlb.si/svetovalec-za-podjetja-podravsko-pomurska-regija-27-10-2015

Solution: to include occupations from classification of occupations into whitelist of keywords for URLs.

g) Advertisement keyword. We have allowed the keyword »advertisement« to be used in the whitelist, but it is too general. We get all kinds of links that are not employment links so we excluded it from whitelist.

Solution: We might use keyword »job advertisement«.

h) The separator in the output of Spider is comma ',', but comma might also appear in the URLs, which can confuse output.

Example: http://www.iskra-mehanizmi.si/ism_slo,,kadri,prosta_delovna_mesta.htm

Solution: we might change the separator from comma to semicolon.

2. DOWNLOADER

a) Endings: In Spider we block links that represent images, documents, music files and compressions with endings like .jpg, .ppt, .xls, .doc, .pdf, .mp3, . mp4, .avi, .zip, .rar, and many more …. and protocols ftp://.

We save only »normal« links without endings but when we download them, some of them still turn out to be documents, music files or images.

Examples:

PDF:

http://www.nlb.si/proklik-prirocnik-sdd-ugovori-soglasja

http://www.mg-soft.si/press/zaposlitev.pdf?p1=profile

MUSIC FILE:

http://www.plinarna-maribor.si/bin?bin.svc=obj&bin.id=98F9B701-14DC-6527-192F-937666F04617

http://www.manufaktura.si/mma/oglas_regeneracija_radio/2011121317254244/

IMAGE:

http://www.goinfo.si/mma/Opremljeno_delovno_mesto_/2011052710504432/mid/

Solution: more careful processing of links.

b) There are also »normal« links that in reality offer us to save the document or image or music file:

PDF:

http://www.hyundai.si/files/9861/HY-Mursak15_204x280-Motorevija_TISK.pdf?download

http://www.komunala-radovljica.si/library/includes/file.asp?FileId=168

http://www.pronet-kr.si/GetFile.ashx?id=813

MUSIC:

http://www.hyundai.si/files/9865/Hyundai_Hokej_Mursak_Zvok_17sek_MP4.mp4?download

WORD:

http://www.plinarna-maribor.si/bin?bin.svc=obj&bin.id=2D7F844C-C294-34B6-CECC-A65C2ADCF92A

IMAGE:

http://www.ddmaribor.si/index.php/fotografije/70-lep-literarnoglasbeni-vecer-s-ferijem-lainsckom/detail/1874-lep-literarnoglasbeni-vecer-s-ferijem-lainsckom?tmpl=component&phocadownload=2

CSS:

http://global.careers.ppg.com/CMSPages/GetResource.ashx?stylesheetname=CPJobsLayout

Solution: more careful processing of links.

c) Dowloading errors.

Requests.exceptions.SSLError: [SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed (_ssl.c:590)

Example:

https://www.europapier.com/si/o-podjetju/novice/article/zaposlimo-referenta-v-nabavi-mz-2322/

Solution: catch the error.

d) OCR. Sometimes JV are published in PDF or in a image, so a perfect application would include OCR scanning.

3. DETECTOR

a) Student work. We would like to ignore student work, but it can happen that JV and student work are published on the same page. How to distinguish between them?

We can put »student work« on the blacklist and ignore the webpage, but in that case we might exclude good JV. For example there was a phrase »student work« in the tab of one of the links on the employment webpage. The detector ignored the webpage because of that because »student work« was on blacklist.

Example:

http://www.intereuropa.si/index.php?page=static&item=18

Solution: to be discussed.

4. CLASSIFIER

Classifier is relatively quickly implementable. When we detect a job vacancy, we just use classification of occupations and see what class of occupation it is.

Technology

In the first solution we have tried to use apache-nutch-1.9 and solr-4.10.4 and lucene-core-3.6.2.jar and lucene-core-3.6.2-javadoc.jar. , which could work with some code adjustment or we could also use Nutch 1.7 and Solr 3.6.2.

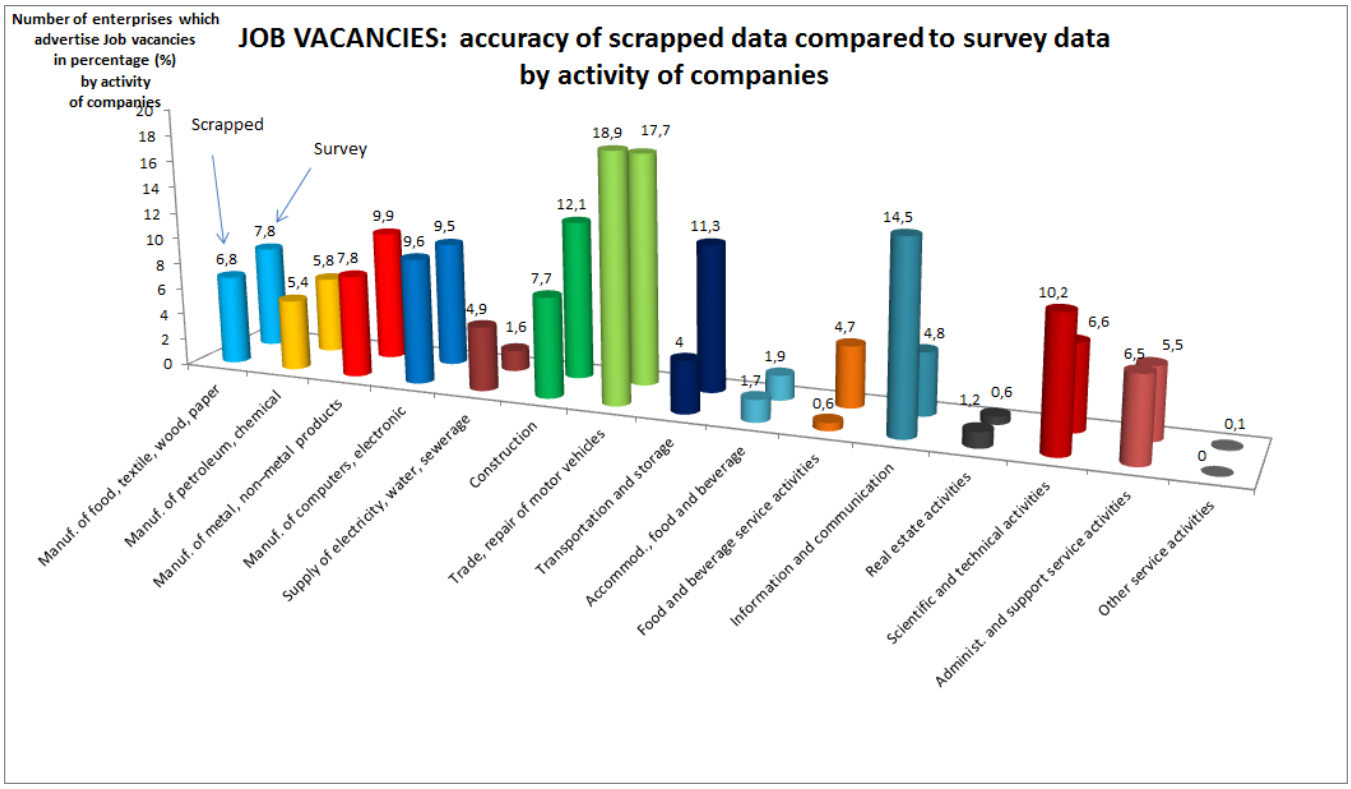
At this point we stopped.

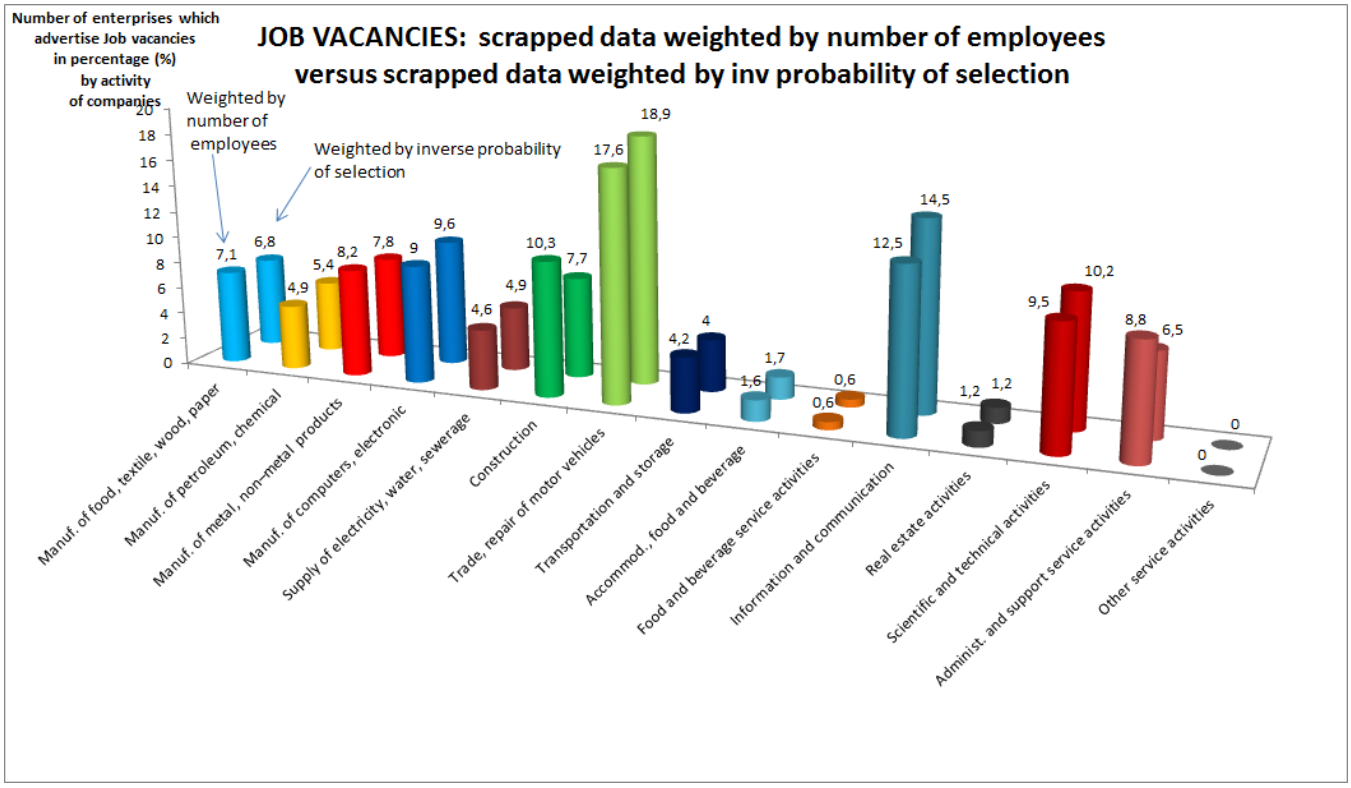
In the second solution we have used software Python 2.7.10 and Scrapy 1.0.3. on the desk computer with

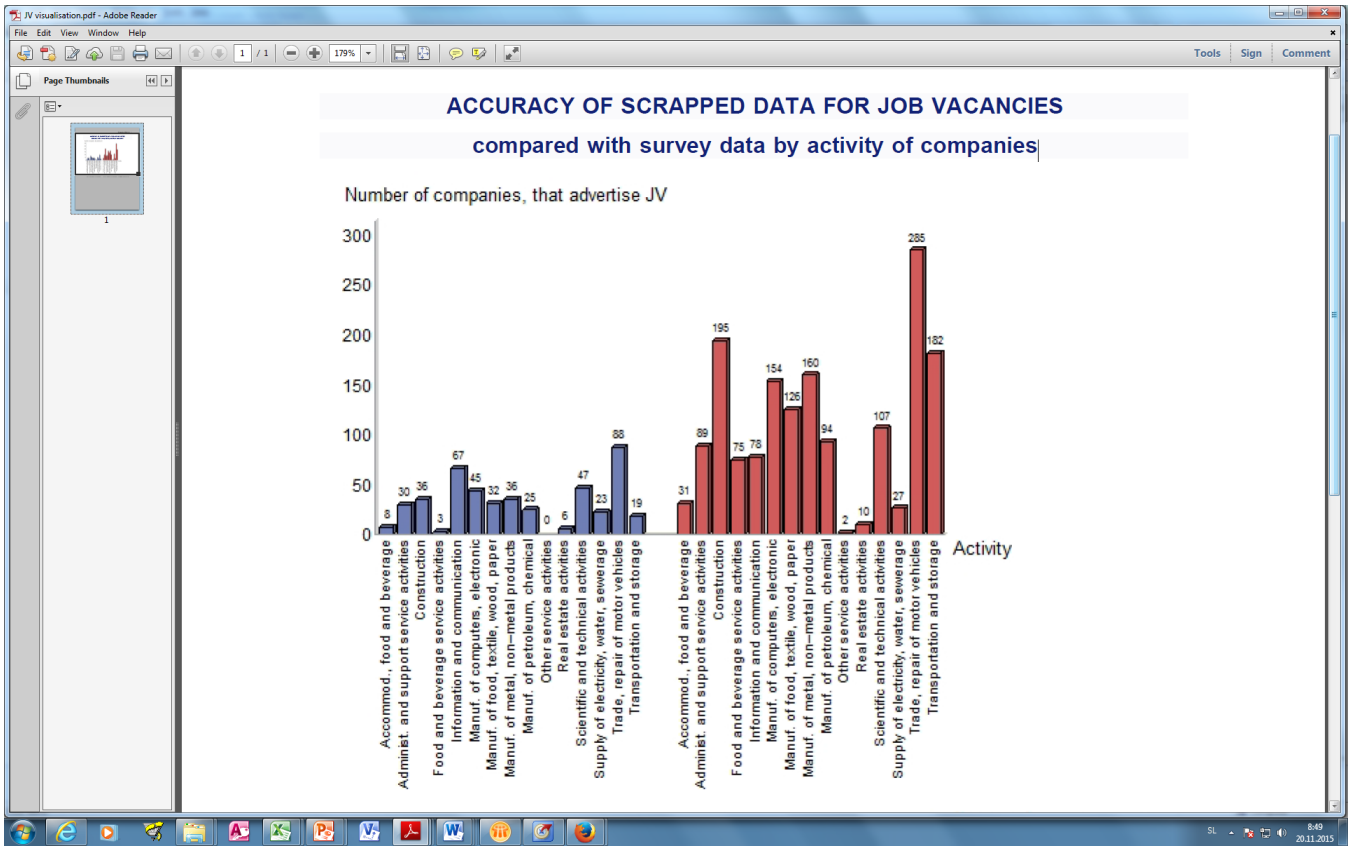
- Intel (R) Core (TM) i5-4590 CPU @3.30 GHz processor
- 8,00 GB RAM
- 64-bit OS

Outputs

QUALITY OF SCRAPPED DATA FOR JOB VACANCIES (SLOVENIAN DATA),
Comparison with survey data on Job Vacancies


Number of enterprises which advertise Job vacancies in percentage (%) by activity of companies
JOB VACANCIES: accuracy of scrapped data compared to survey data by activity of companies
Scrapped
Survey
20
18
16
14
12
10
8
6
4
2
0
6,8
7,8
5,4
5,8
7,8
9,9
9,6
9,5
4,9
1,6
7,7
12,1
18,9
17,7
4
11,3
1,7
1,9
0,6
4,7
14,5
4,8
1,2
0,6
10,2
6,6
6,5
5,5
0
0,1
Manuf. of food, textile, wood, paper
Manuf. of petroleum, chemical
Manuf. of metal, non-metal products
Manuf. of computers, electronic
Supply of electricity, water, sewerage
Construction
Trade, repair of motor vehicles
Transportation and storage
Accommod., food and beverage
Food and beverage service activities
Information and communication
Real estate activities
Scientific and technical activities
Administ. and support service activities
Other service activities


Number of enterprises which advertise Job vacancies in percentage (%) by activity of companies
JOB VACANCIES: scrapped data weighted by number of employees versus scrapped data weighted by inv probability of selection
Weighted by number of employees
Weighted by inverse probability of selection
20
18
16
14
12
10
8
6
4
2
0
7,1
6,8
4,9
5,4
8,2
7,8
9
9,6
4,6
4,9
10,3
7,7
17,6
18,9
4,2
4
1,6
1,7
0,6
0,6
12,5
14,5
1,2
1,2
9,5
10,2
8,8
6,5
0
0
Manuf. of food, textile, wood, paper
Manuf. of petroleum, chemical
Manuf. of metal, non-metal products
Manuf. of computers, electronic
Supply of electricity, water, sewerage
Construction
Trade, repair of motor vehicles
Transportation and storage
Accommod., food and beverage
Food and beverage service activities
Information and communication
Real estate activities
Scientific and technical activities
Administ. and support service activities
Other service activities



Findings

The main problem of the JV pilot is determining the existence of the URLs of enterprises. Additional effort should be put in to create the tool for collecting the URLs or finding the existing databases of URLs. The prototype of the IT tool for detecting the JV data is very promising and could be used as an input for following projects (e.g. ESSNET project starting in 2016). The part of the IT which is not developed yet is Classifier (the team did not have enough time to develop this part) and Splitter (there is still not high enough precision in determining more the one JV in a document which contains more JV ads).

Methodology

In order to collect the data of job vacancies from websites of enterprises there are three methodological issues which were discovered during the project

- Methodology of collecting the URLs of enterprises

  This solution started with names of the companies and produced their URLs. That part was mainly prepared by Istat. The solution collected the first 10 hits for the entered name of the company in the Bing search engine and predicted which one of the hits is most likely to be the official URL of the company. The prediction was done with custom made scoring algorithm.

  The idea of collecting company URLs is to use the list of names of the companies.

  The name of a company is typed in the Bing search engine. The first 10 hits are collected. From these, the most common words (which are related to the company; e.g. name, tel. number, tax number,..) are scraped and then every URL is given the score according to the special algorithm.. The URL with the highest score is recognized as a URL of the company.

- Methodology of detecting the URLs related to JV, detecting the JV ads and classifying JV ads

  The algorithm searches all the sub links of a given URL. If a link contains a key word included on the list of words related to job vacancy (Whitelist) then the URL is stored in the database. There is also a list of key words (e.g. student work) which are considered as signs that the examined sub link is not related to the job vacancy (Blacklist)

Example. Below link contain the keyword "jobs" which is on the Whitelist. The link is stored in the database.

http://outfit7.com/jobs/

The algorithm for detecting the URL which advertises the JV is described in the section Activities (task number 4).

The algorithm for classifying JVs was not developed. Some initially research results could be found at

http://www1.unece.org/stat/platform/download/attachments/116818298/Semantic%20analysis%20of%20job%20advertisements.pdf?version=1&modificationDate=1444024574135&api=v2

- Methodology for creating statistics

The fact that a prototype of the IT solution could only detect whether the enterprise advertise JV or not, determines the methodology.

In reality there will not be a random sample of URL links, therefore it is not possible to use probability sampling techniques to calculate the weights. The recommendation is to use the calibration techniques where the auxiliary variables are number of employees of the unit, turnover of the unit etc. For the purposes of the Slovenian experiment, only the number of employees was chosen as a calibration variable.

Having the information about the number of employees we can calculate the ratio (calibrated weight) of sum of all employees of units in certain domain (strata) and sum of all employees of units which advertise JV in the sample. For example we could calculate the weighted sum of units which advertise JV using number of employees as a weight.

WEIGHTS (domain (i,j) is defined by activity and size of unit)

$$w_{activity\ \ i,size\ j} = \frac{sum\ of\ number\ of\ employees\ of\ all\ units\ in\ activity\ i\ and\ size\ class\ j}{sum\ of\ number\ of\ employees\ of\ sample\ units\ in\ acivity\ i\ and\ size\ class\ j}$$

STATISTICS

$$Number\ of\ units\ \ which\ advertise\ JV = \sum_{activit\ y\ i\ ,size\ \ j} w_{i,j} \times JV_{i,j}$$

$$JV_{i,j} - number\ of\ units\ which\ advertise\ JV\ in\ activity\ i\ and\ size\ \ class\ size$$

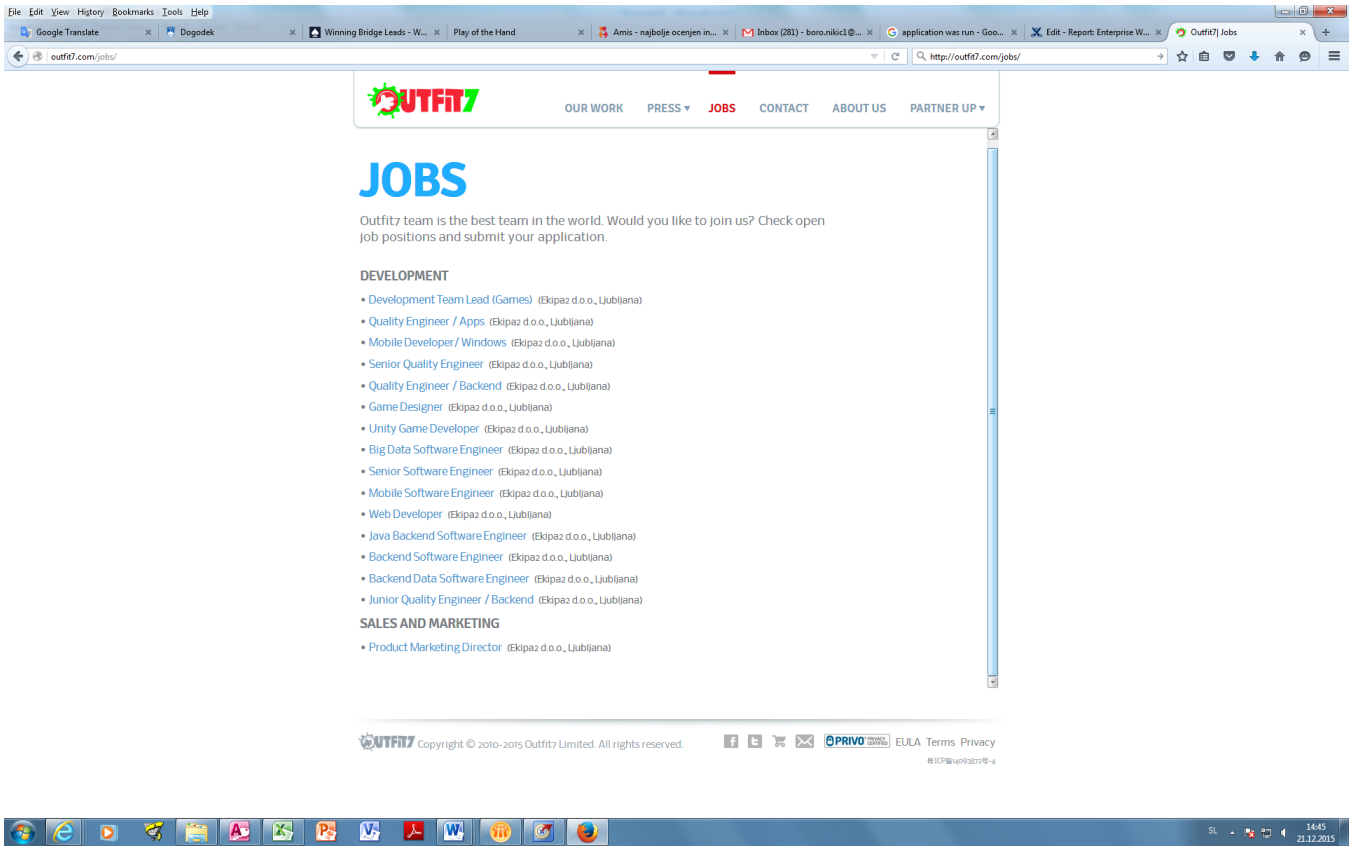
$$(from\ scraped\ data)$$

The ratio could be broken down by activity of units. If we create statistics on level of activity we could also calculate the totals of units advertising the JV by activity and (or) region.

Source

Sources for JV statistics are websites of companies that contain job vacancy advertisements. It needs to be emphasised again that the focus was enterprises which advertise job vacancies for their purposes and not job portals which advertise JV for purposes of other companies. Structures of websites are very diverse which makes the process of web scraping quite difficult. The source is not complete in the sense that around 10% (estimation) of enterprises don't have a website and an additional 30% (out of 90%) don't have sub sites which advertise JV (Slovenian case). There is a need for further investigation about the current situation in different countries regarding the existence of websites and sub websites related to job vacancies. A special dimension is privacy of websites. There are enterprises which do not permit the use of web scraping techniques for collecting data. Some other results of the investigations of sources:

- The main issue is the list of URLs of enterprises which are not (completely) available in the most of the countries
- Some websites of enterprises are in different languages (English, German, Croatian, etc.) and also advertise jobs in other countries. An example of such an enterprise is one of the most successful companies in Slovenia Outfit7, which advertises (only in English) quite a lot jobs in Slovenia and also in other countries.
  http://outfit7.com/jobs/



- Multinational companies like Heineken which advertise JV on their main page (Heineken owns Slovenia brewery Laško http://www.lasko.eu/en/) again for many countries. A speciality of this website is also that you need to confirm your age in order to enter the website (which makes web scraping complicated)
- Some companies advertise JV in pdf (or similar) formats. The current IT solution works only with HTML format.
- Companies change the domains of their websites quite often. There is a need for constant updating of the list of URLs of enterprises. Some important information like date of publishing the JV is often missing.
- Some out of date advertisements may still be visible on website.

Sandbox

The prototype of application is available in the Sandbox together with the testing set of scraped URL links and data. For access to the Sandbox application and data please contact support.stat@un.org. For additional questions about the application please contact Boro Nikic boro.nikic@gov.si or Marko Limbek Marko.limbek@gov.si.

Link to the Wiki page with some instructions.

Application to detect job vacancies

Performance comparison between Sandbox and local computer

To test performance capabilities, 100 Slovenian websites were manually chosen and divided in groups of 25 URLs. Then the prototype of the Python application was run in order to compare how many sub-sites were detected and time needed for processing. The table below table shows that the "Sandbox" is approximately 5 to 9 times faster than local PC desktop.

	Web sites	**Number of detected URLS which are related to job vacancies**	**Time**
Sandbox	First 25 sites	155	1 hour 5 min
Desktop	First 25 sites	163	9 hour 24 min
Sandbox	Second 25 sites	282	2 hour 1 min
Desktop	Second 25 sites	261	12 hour 45 min
Sandbox	Third 25 sites	102	1 hour 19 min

Desktop	Third 25 sites	104	5 hour 20 min
Sandbox	Fourth 25 sites	7967	3 hour 10 min
Desktop	Fourth 25 sites	7737	21 hour 48 min

In order to test parallel processing performance, an additional test was made. 10 websites of enterprises were chosen and divided into 10 groups with 1 URL. The prototype Python application was then run, firstly on the whole group of 10 URLs, and in the second phase the application was run on 10 groups with 1 URL. In both cases approximately the same amount of time was needed (9 hours). The results of experiment show that the Python language already incorporates parallel processing of URLs of enterprises irrespective of the distribution of them in one or more sets.

Conclusions

The team working on the enterprise websites with the focus on JV advertisements discovered the possibility of collecting the URLs of enterprises, determining the methodology of detecting the job vacancy, determining the web scraping system for identifying job advertisements, creating the prototype of the IT solution for identifying job advertisements, and the methodology for creating possible statistics. The team focused on creating and testing the prototype of the IT tool for detecting the job vacancies and achieved great success. The first statistics created look promising but there is a need for further testing and improvements to the efficiency of the IT tool (especially on the Classifier module) which will allow us to create more detailed statistics about job vacancies. Due to the fact that demand for statistics on job vacancies is very high, there are other projects which will be conducted in near future (ESSNET pilots under the Eurostat umbrella), the deliverables of this Sandbox project will be great inputs for carrying out similar pilots. The results of the project will be presented in many events, for example at the European Conference on Quality in Official Statistics (Q2016) http://q2016.ine.es/.

- ***How far are we from something that can be published?***

In terms of creating useful statistics the main problem for publishing results is the lack of lists of URLs of enterprises in most countries. When you don't have the list of URLs it is very hard to collect the data and produce statistics on the whole population.

The team started to scrape the data and detect the JV a few months ago. In our opinion we need at least half a year to test calculated statistics (number of enterprises which offer JV ads broken down by NACE groups). In reality we will most probably use multiple sources (Job portals and administrative data from agency of Employment, etc.) in order to get reliable statistics.

In terms of the prototype IT tool, and methodology of detecting the JV, the results are quite promising and we could say if we work hard we could publish very interesting results (and at least for two countries).

- ***What case has been made for the future of the Sandbox?***

The team are working on installing the prototype of the IT tool for detecting the JV in the Sandbox environment. So the IT tool will be available in the Sandbox and ready (with small modifications of input parameters) for use by other participants (countries). The main problem is sharing URLs. Due to the policy of NSIs, it is not allowed to share collected microdata (URLs) from surveys. One should have the list of URLs if he(she) wants to use the IT tool for web scraping. However, for testing purposes we don't need too much data. We will put a test (manually chosen) set of URLs in the Sandbox.

Task team members

- Ingegerd Jansson (se)
- Marko Limbek (si)
- Boro Nikic (si)
- Oskar Nyqvist (se)
- Karol Potocki (pl)
- Marco Puts (nl)
- Dan Wu (se)