Roomba - Automatic Validation, Correction and Generation of Dataset Metadata

Enhancing Dataset Search and Spam Detection

†‡ Ahmad Assaf, ‡ Aline Senart, and † Raphaël Troncy
†EURECOM, Sophia Antipolis, France
‡ SAP Labs, Sophia Antipolis, France
†firstName.lastName@eurecom.fr, ‡firstName.lastName@sap.com

1. ABSTRACT

Data is being published by both the public and private sectors and covers a diverse set of domains from life sciences to media or government data. An example is the Linked Open Data cloud which is potentially a gold mine for organizations and individuals who are trying to leverage external data sources in order to produce more informed business decisions. Considering the significant variation in size, the languages used and the freshness of the data, one realizes that finding useful datasets without prior knowledge is increasingly complicated. In this paper, we propose Roomba, a scalable automatic approach for extracting, validating, correcting and generating descriptive linked dataset profiles. We target CKAN powered data portals and validate our framework on the Linked Open Data (LOD) cloud. The results demonstrate that the general state of Linked Open Data (LOD) cloud needs more attention as most of the datasets suffer from bad quality metadata lacking some informative metrics needed to facilitate dataset search.

2. INTRODUCTION

The main entry point for discovering and identifying datasets is either through public data portals such as DataHub[1] and Europe's Public Data[2] or private search engines such as Quandl[3] and Engima[4]. Private portals harness manually curated data from various sources and expose them to users either freely or through paid plans. The data available is of higher quality but lesser quantity compared to what is available in public portals. Similarly, in some public data portals, administrators manually review datasets information, validate, correct and attach suitable metadata. This information is mainly in the form of predefined tags such as *media, geography, life sciences* for organization and clustering purposes. However, the diversity of those datasets makes it harder to classify them in a fixed number of predefined tags that can be subjectively assigned without capturing the essence and breadth of the dataset [2]. Furthermore, the increasing number of datasets available makes the metadata review and curation process unsustainable even when outsourced to communities. *Data profiling* is the process of creating descriptive information and collect statistics about that data. It is a cardinal activity when facing an unfamiliar dataset [3]. It helps in assessing the importance of the dataset, in improving users' ability to search and reuse part of the dataset and in detecting irregularities to improve its quality. Data profiling includes typically several tasks: **Metadata profiling**: Provides general information on the dataset (dataset description, release and update dates), legal information (license information, openness), practical information (access points, data dumps), etc. **Statistical profiling**: Provides statistical information about data types and patterns in the dataset, i.e. properties distribution, number of entities and RDF triples, etc. **Topical profiling**: Provides descriptive knowledge on the dataset content and structure. This can be in form of tags and categories used to facilitate search and reuse. In this paper, we propose Roomba, a scalable automatic approach for extracting, validating, correcting and generating descriptive linked dataset profiles. We address the challenges of automatic validation and generation of descriptive datasets profiles. This paper proposes Roomba, an extensible framework consisting of a processing pipeline that combines techniques for data portals identification, datasets crawling and a set of pluggable modules combining several profiling tasks. Roomba validates the provided dataset metadata against an aggregated standard set of information. Metadata fields are automatically corrected when possible, e.g. adding a missing license URL reference. Moreover, a report describing all the issues highlighting those that cannot be automatically fixed is created to be sent by email to the dataset's maintainer. There exist various statistical and topical profiling tools for both relational and Linked Data. The architecture of the Roomba allows to easily add them as additional profiling tasks. However, in this paper, we focus on the task of dataset metadata profiling and present our findings by running Roomba on the LOD cloud[5]. The results demonstrate that the general state of LOD cloud needs more attention as most of the datasets suffer from bad quality metadata lacking some informative metrics needed to facilitate dataset search. The

[1]http://datahub.io
[2]http://publicdata.eu
[3]https://quandl.com/
[4]http://enigma.io/

[5]http://datahub.io/dataset?tags=lod

WWW '15 Florence, Italy

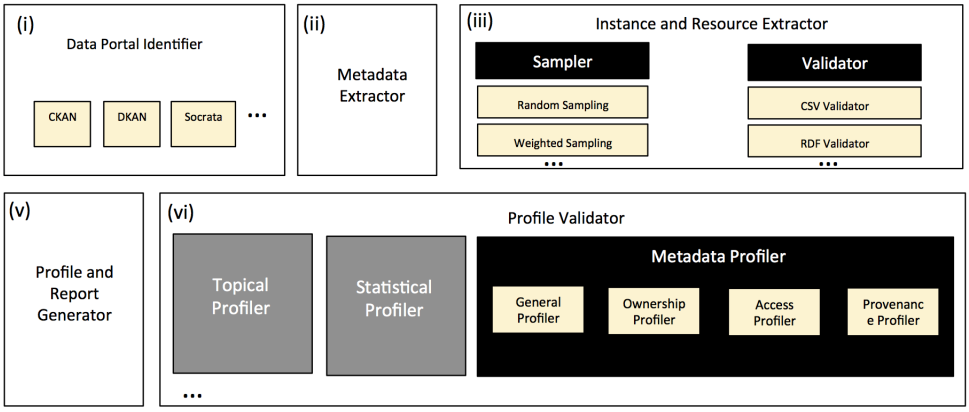

Figure 1: Processing pipeline for validating and generating dataset profiles

noisiest metadata are the access information such as licensing information, resource descriptions as well as resource availability problems.

3. RELATED WORK

The Project Open Data Dashboard[6] tracks and measures how US government websites implement the Open Data principles to understand the progress and current status of their public data listings. A validator analyzes machine readable files e.g. JSON files for automated metrics like the resolved URLs, HTTP status and content-type. However, deep schema information about the metadata is missing like description, license information or tags. Similarly on the LOD cloud, the Data Hub LOD Validator[7] gives an overview of Linked Data sources cataloged on the Data Hub. It offers a step-by-step validator guidance to check a dataset completeness level for inclusion in the LOD cloud. The results are divided into four different compliance levels from basic to reviewed and included in the LOD cloud. Although it is an excellent tool to monitor LOD compliance, it still lacks the ability to give detailed insights about the completeness of the metadata and overview on the state of the whole LOD cloud group and is very specific to the LOD cloud group rules and regulations.
Although the above mentioned tools are able to provide various information about a dataset, there exist no approach that is extensible to combine further information coming from various profiling tools.

4. FRAMEWORK ARCHITECTURE

In this section, we provide an overview of the processing steps for validating and generating dataset profiles. Figure 1 shows the main steps which are the following: (i) Data portal identification; (ii) metadata extraction; (iii) instance and resource extraction; (iv) profile validation (v) profile and report generation.

4.1 System Overview

Roomba is built as a Command Line Interface (CLI) application using Node.js. Instructions on installing and running Roomba are available on its public Github repository[8] and explained in this short screencast[9]. Related functions are encapsulated into modules that can be easily plugged in/out the processing pipeline.

4.2 Data Portal Identification

Data portals can be considered as data access points providing tools to facilitate data publishing, sharing, searching and visualization. CKAN[10] is the world's leading open-source data portal platform powering websites like the DataHub, Europe's Public Data and the U.S Government's open data. Modeled on CKAN, DKAN[11] is a standalone Drupal distribution that is used in various public data portals as well. Socrata[12] helps public sector organizations improve data-driven decision making by providing a set of solutions including an open data portal. In addition to these tradition data portals, there is a set of tools that allow exposing data directly as RESTful APIs like Datatank[13] and Database-to-API[14].

Identifying the software powering data portals is a vital first step to understand the API calls structure. Web scraping is a technique for extracting data from Web pages. We rely on several scraping techniques in the identification process which includes a combination of the following:

- **URL inspection**: Check the existence of certain URL patterns. Various CKAN based portals are hosted on subdomains of the `http://ckan.net`. For example, CKAN Brazil (`http://br.ckan.net`).
- **Meta tags inspection**: The `<meta>` tag provides metadata about the HTML document. They are used to specify page description, keywords, author, etc. Inspecting the `content` attribute can indicate the type of the data portal. We use CSS selectors to check the existence of these meta tags. An example of a query selector is `meta[content*="ckan]` (all meta tags with the attribute content containing the string $CKAN$).

[6]http://labs.data.gov/dashboard/
[7]http://validator.lod-cloud.net/
[8]https://github.com/ahmadassaf/opendata-checker
[9]http://link-to-screencast.com
[10]http://ckan.org
[11]http://drupal.org/project/dkan
[12]http://www.socrata.com
[13]http://thedatatank.com
[14]https://github.com/project-open-data/db-to-api

This selector can identify CKAN portals whereas the `meta[content*="Drupal"]` can identify DKAN portals.

- **Document Object Model (DOM) inspection**: Similar to the meta tags inspection, we check the existence of certain DOM elements or properties. For example, CKAN powered portals will have DOM elements with class names like `ckan-icon` or `ckan-footer-logo`. A CSS selector like `.ckan-icon` will be able to check if a DOM element with the class name `ckan-icon` exists. The list of elements and properties to inspect is stored in a separate configurable object for each portal. This allows the addition and removal of elements as deemed necessary.

The identification process for each portal can be easily customized by overriding the default function. Moreover, adding or removing steps from the identification process can be easily configured.
After those preliminary checks, we query one of the portal's API endpoints. For example, DataHub is identified as CKAN, so we will query the API endpoint on `http://datahub.io/api/action/package_list`. A successful request will list the names of the site's datasets, whereas a failing request will signal a possible failure of the identification process.

4.3 Metadata Extraction

Data portals expose a set of information about each dataset as metadata. The model used varies across portals. However, a standard model should contain information about the dataset's title, description, maintainer email, update and creation date, etc. We divided the metadata information into the following:

General information: General information about the dataset. e.g. title, description, ID, etc. This general information is manually filled by the dataset owner. In addition to that, tags and group information is required for classification and enhancing dataset discoverability. This information can be entered manually or inferred modules plugged into the topical profiler.

Access information: Information about accessing and using the dataset. This includes the dataset URL, license information i.e. license title and URL and information about the dataset's resources. Each resource has as well a set of attached metadata e.g. resource name, URL, format, size, etc.

Ownership information: Information about the ownership of the dataset. e.g. organization details, maintainer details, author, etc. The existence of this information is important to identify the authority on which the generated report and the newly corrected profile will be sent to.

Provenance information: Temporal and historical information on the dataset and its resources. For example, creation and update dates, version information, version, etc. Most of this information can be automatically filled and tracked.

Although Roomba is generic and accepts any data model to check against, for this demo we have used the the CKAN standard model[15] as do our validation on the LOD cloud hosted on CKAN.

[15]http://goo.gl/8RofC8

After identifying the underlying portal software, we perform iterative queries to the API in order to fetch datasets metadata and persist them in a file-based cache system. Depending on the portal software we can issue specific extraction jobs. For example, in CKAN based portals, we are able to crawl and extract the metadata of a specific dataset, all the datasets in a specific group e.g. LOD Cloud or all the datasets in the portal.

4.4 Instance and Resource Extraction

From the extracted metadata we are able to identify all the resources associated with that dataset. They can have various types like a SPARQL endpoint, API, file, visualization ,etc. However, before extracting the resource instance(s) we perform the following steps:

- **Resource metadata validation and enrichment**: Check the resource attached metadata values. Similar to the dataset metadata, each resource should include information about its mimetype, name, description, format, valid de-referenceable URL, size, type and provenance. The validation process issue an HTTP request to the resource and automatically fills up various missing information when possible, like the mimetype and size by extracting them from the HTTP response header. However, missing fields like name and description that needs manual input are marked as missing and will appear in the generated summary report.

- **Format validation**: Validate specific resource formats against a linter or a validator. For example, node-csv[16] for CSV files and n3[17] to validate N3 and Turtle RDF serializations.

Considering that certain dataset contains large amounts of resources and the limited computation power of some machines on which Roomba might run on, a sampler module is introduced to execute various sample-based strategies discussed in [1] and were found to generate accurate results even with comparably small sample size of 10%.

4.5 Profile Validation

Metadata validation process identifies missing information and the ability to automatically correct them. Each set of metadata (general, access, ownership and provenance) is validated and corrected automatically when possible. Each profiler task has a set of metadata fields to check against. The validation process check if each field is defined and if the value assigned is valid.

There exist a bunch of special validation steps for various fields. For example, for ownership information where the maintainer email has to be defined, the validator checks if the email field is an actual valid email address. A similar process is done to URLs whenever found. However, we also issue an HTTP `HEAD` request in order to check if that URL is reachable or not. For the dataset resources, we use the `content-header` information when the request is successfull in order to extract, compare and correct further metadata values like mimetype and content size.

[16]https://github.com/wdavidw/node-csv
[17]https://github.com/RubenVerborgh/N3.js

From our experiments, we found out that datasets' license information is noisy. The license names if found are not standardized. For example, Creative Commons CCZero can be also CC0 or CCZero. Moreover,the license URI if found and if de-referenceable can point to different reference knowledge bases e.g. `http://opendefinition.org`. To overcome this issue, we have manually created a mapping file standardizing the set of possible license names and the reference knowledge base[18]. In addition, we have also used the open source and knowledge license information[19] to normalize the license information and add extra metadata like the domain, maintainer and open data conformance.

```
=============================================================

                        Metadata Report

=============================================================

group information is missing. Check organization
     information as they can be mixed sometimes
organization_image_url field exists but there is no value
     defined

=============================================================

                        Tag Statistics

=============================================================

There is a total of: 21 [undefined] vocabulary_id fields
     100.00%

=============================================================

                        License Report

=============================================================

License information has been normalized !

=============================================================

                        Resource Statistics

=============================================================

There is a total of: 10 [missing] url-type fields  100.00%
There is a total of: 9 [missing] created fields  90.00%
There is a total of: 10 [undefined] size fields  100.00%
There is a total of: 10 [undefined] hash fields  100.00%
There is a total of: 7 [undefined] mimetype fields  70.00%
There is a total of: 10 [undefined] cache_url fields
     100.00%
There is a total of: 6 [undefined] name fields  60.00%
There is a total of: 9 [undefined] last_modified fields
     90.00%
There is one [undefined] format field  10.00%

=============================================================

                        Resource Connectivity Issues

=============================================================

There are 2 connectivity issues with the following URLs:
   - http://dbpedia.org/void/Dataset

=============================================================

                        Un-Reachable URLs Types

=============================================================

There are: 1 unreachable URLs of type [file]
```

Listing 1: Excerpt of the DBpedia validation report

4.6 Profile and Report Generation

The validation process highlights the missing information and presents them in a human readable report. The report can be automatically sent to the dataset maintainer email if exists in the metadata.

In addition to the generated report, the enhanced profiles are represented in JSON using the CKAN data model and are publicly available[20].

Data portal administrators need an overall knowledge of the portal datasets and their properties. Roomba has the ability to generate numerous reports of all the datasets by passing formated queries. There are two main set of aggregation tasks that can be run:

- **Aggregating meta-field values**: Passing a string that corresponds to a valid field in the metadata. The field can be flat like `license_title` (aggregates all the license titles used in the portal or in a specific group) or nested like `resource>resource_type` (aggregates all the resources types for all the datasets). Such reports are important to have an overview of the possible values used for each metadata field.
- **Aggregating key:object meta-field values**: Passing two meta-field values separated by a colon `:` e.g. `resources>resource_type:resources>name`. These reports are important as you can aggregate the information needed when also having the set of values associated to it printed.

5. DEMONSTRATION

During the demo we will show how one can find crawl a data portal, generate reports based on manual queries over the datasets metadata, validate a dataset profile and generate a new enriched profile with automatically fixed problems. Moreover, users will be invited to try Roomba providing their own datasets hosted on any CKAN-powered portal and directly check the generated report.

6. CONCLUSION

Roomba is flexible and extensible. It can be plugged into data portals or used as a standalone tool to check for bad quality dataset metadata and identify possible spam. Automatically corrected profiles are of higher quality thus improving dataset search and retrieval. Further work includes introducing workflows that will be able to correct the rest of the metadata either automatically or through intuitive manually-driven interfaces. We also plan to integrate statistical and topical profilers to be able to generate full comprehensive profiles.

7. REFERENCES

[1] B. Fetahu, S. Dietze, B. Pereira Nunes, M. Antonio Casanova, D. Taibi, and W. Nejdl. A scalable approach for efficiently generating structured dataset topic profiles. In *The Semantic Web: Trends and Challenges*, volume 8465 of *Lecture Notes in Computer Science*. Springer International Publishing, 2014.

[2] S. Lalithsena, P. Hitzler, A. Sheth, and P. Jain. Automatic domain identification for linked open data. In *Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2013 IEEE/WIC/ACM International Joint Conferences on*, volume 1, pages 205–212, Nov 2013.

[3] H. Li. Data profiling for semantic web data. In F. Wang, J. Lei, Z. Gong, and X. Luo, editors, *Web Information Systems and Mining*, volume 7529 of *Lecture Notes in Computer Science*, pages 472–479. Springer Berlin Heidelberg, 2012.

[18] https://github.com/ahmadassaf/opendata-checker/blob/master/util/licenseMappings.json

[19] https://github.com/okfn/licenses

[20] https://github.com/ahmadassaf/opendata-checker/tree/master/results