UNIVERSITY OF MILANO - BICOCCA
DEPARTMENT OF COMPUTER SCIENCE, SYSTEMS AND COMMUNICATIONS
DOCTORAL SCHOOL IN COMPUTER SCIENCE - XXIX CYCLE

PH.D. THESIS

Profiling Linked Data

Author: Ing. Blerina SPAHIU

Advisors: Prof. Andrea MAURINO
Dr. Matteo PALMONARI
Tutor: Prof. Flavio De PAOLI

I dedicate this thesis to my life-coaches; mom and dad.
I could not have asked for better parents or role-models.
Thank you for making my life an amazing journey!

Abstract

Recently, the increasing diffusion of Linked Data (LD) as a standard way to publish and structure data on the Web has received a growing attention from researchers and data publishers. LD adoption is reflected in different domains such as government, media, life science, etc., building a powerful Web available to anyone.

Despite the high number of datasets published as LD, their usage is still not exploited as they lack comprehensive metadeta. Data consumers need to obtain information about datasets content in a fast and summarized form to decide if they are useful for their use case at hand or not. Data profiling techniques offer an efficient solution to this problem as they are used to generate metadata and statistics that describe the content of the dataset. Existing profiling techniques do no cover a wide range of use cases. Many challenges due to the heterogeneity nature of Linked Data are still to overcome.

This thesis presents the doctoral research which tackles the problems related to Profiling Linked Data. Even though the term of data profiling is the umbrella term for diverse descriptive information that describes a dataset, in this thesis we cover three aspects of profiling; topic-based, schema-based and linkage-based. The profile provided in this thesis is fundamental for the decision-making process and is the basic requirement towards the dataset understanding.

In this thesis we present an approach to automatically classify datasets in one of the topical categories used in the LD cloud. Moreover, we investigate the problem of multi-topic profiling. For the schema-based profiling we propose a schema-based summarization approach, that provides an overview about the relations in the data. Our summaries are concise and informative enough to summarize the whole dataset. Moreover, they reveal quality issues and can help users in the query formulation tasks. Many datasets in the LD cloud contain similar information for the same entity. In order to fully exploit its potential LD should made this information explicit. Linkage profiling provides information about the number of equivalent entities between datasets and reveals possible errors.

The techniques of profiling developed during this work are automatic and can be applied to different datasets independently of the domain.

Acknowledgements

I would like to thank everyone who contributed in some way in the journey of PhD pursuit. First and foremost, I would like to thank my supervisors Prof. Andrea Maurino and Dr. Matteo Palmonari for their support and confidence in me. I am grateful for all of the support and contribution along this journey. Especially, I gratefully acknowledge to Prof. Andrea Maurino for the funding received to support my thesis and making this experience productive and stimulating. I am also thankful to my advisor, Prof. Flavio De Paoli for his guidance and interest in my work.

I would like to thank Prof. Christian Bizer, Prof. Heiko Paulheim and Robert Meusel for their collaboration and guidance during the PhD period abroad at the University of Mannheim. Also, I would like to thank, Melisachew Wudage Chekol and Cheng Xie, for being so friendly and making my stay in Mannheim more enjoyable.

I owe a debt of gratitude to Dr. Anisa Rula for her untiring support and guidance throughout my PhD studies. I am thankful for finding a new friend in her and for the excellent example she has provided as a successful woman, wife and mother.

Every achievement in this thesis was accomplished with the help and the support of colleagues and collaborators. Many thanks to Dr. Riccardo Porrini, Dr. Mamoun Abu Helou and Marco Cremaschi.

A special thanks to my friends, Rukiela, Thimilda and Denada. Thank you for being there every time I needed to speak to someone and for making me laugh throughout my darkest days. Thank you to my nephew Matis for making these days sweet with his cute smile and first words. Many thanks to Mira and Kidi for being part of this journey and all the memories we made throughout these years.

Most importantly, thank you to my mother, father, brother and sister-in-law. Thank you for teaching me the real values of life. Thank you for your unconditional support. I have never known a day apart from your love, and that is a priceless gift to me. I hope my life and my choices have honored you.

To the love of my life, Gjergji, thank you for being a friend and a partner. Your love has given me the strength to become better than I was yesterday. Thank you for walking by my side towards our dreams.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Data has become a main pillar in every research field and has turned into a significant resource for every business. With the advancement of the technology, e.g., network connection and storage capabilities, it has become very easy to enable fast moving and sharing data despite geographical distance. This enables users to access huge amount of data more easily. Making available and sharing large datasets allows data consumers to realize transparency, repeatability and interoperability [20]. Despite the fact that organizations have access to more data than ever before, their ability to extract meaningful knowledge is not balancing its accessibility. Most of the data publishers in the World Wide Web (WWW) are more interested in publishing data rather than caring about the format of the data they make available. Most of the data on the Web are published in CSV, XML, or marked up as HTML tables. These formats hide a lot of semantics and structures about the data. Because these formats are not expressive enough, the actual use and consumption of data needs further refinement on many fronts.

Recently, Tim-Berners Lee [21] introduced the Linked Data (LD) paradigm. It refers to a set of best practices for publishing and connecting structured data on the Web. The adoption of such best practices assures that the structure and the semantics of the data are made explicit which is also the main goal of the Semantic Web. The datasets to be published as Linked Data need to adopt a set of principles in a way that it would be simple for them to be searched and queried [26]. These datasets should be published adopting W3C standards in RDF[1] format and made available via SPARQL[2] endpoint queries. RDF provides

[1] https://www.w3.org/RDF/
[2] http://www.w3.org/TR/rdf-sparql-query/

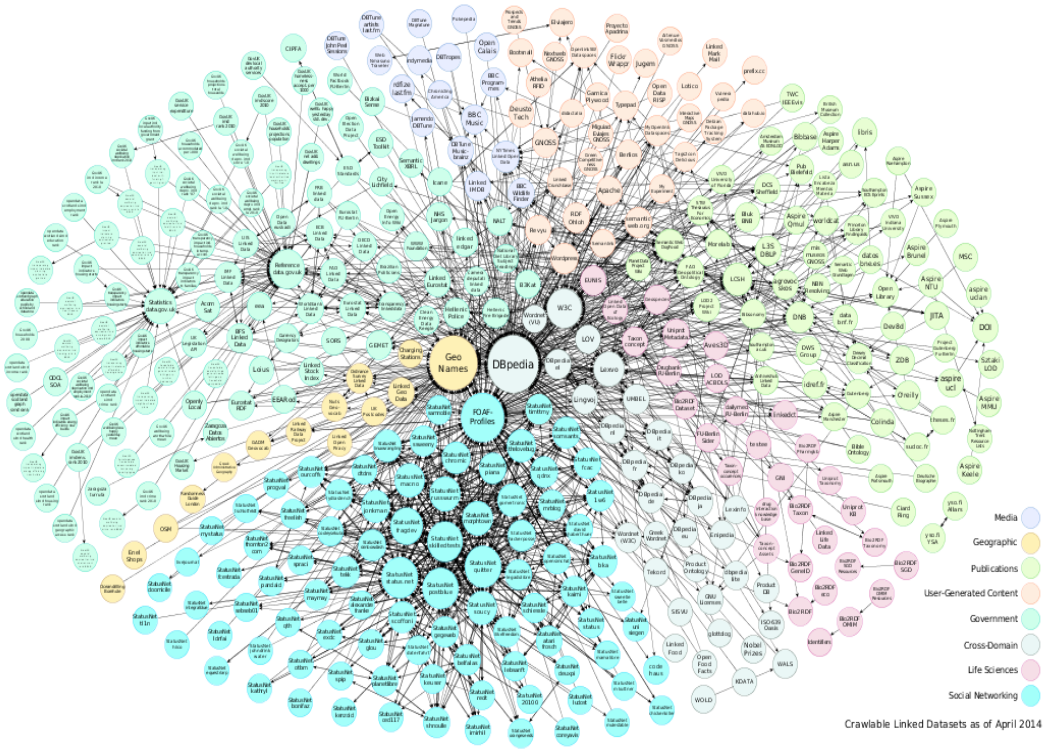


FIGURE 1.1: Linked Data (LD) Cloud

a generic, graph-based data model which comprises; *entities* (such as Italy, Milan), *literals* (such as "milan", "1986"), *classes* (such as country, city, string and date) and relations denoted as *properties* among them (such as totalPopulation) [96]. On the other hand, Linked Data make use of vocabularies and ontologies to describe the semantics of their data by defining classes and properties.

The adoption of Linked Data over the last few years has raised from 12 datasets in 2007, to more than 1000 datasets as of April 2014 [124], a number that is constantly increasing. These datasets[3] cover different domains which is also shown by the different colors in the LD cloud described in Fig. 1.1. In the LD cloud there are about 900,129 documents describing 8,038,396 resources [124]. Although publishing such vast amount of data adopting the principles of Linked Data has many advantages, its consumption is still limited [93].

Even though the Linked Data is considered a gold mine, its consumption is limited as understanding a large and unfamiliar RDF dataset is still a key challenge.

[3]http://linkeddatacatalog.dws.informatik.uni-mannheim.de/state/

The data understanding and data exploration process is supported by descriptive information or metadata that describe the dataset. As a result of a lack of comprehensive descriptive information the exploitation of such data is limited.

Data profiling are techniques that support data consumption, understanding and exploration with statistics and useful metadata about the content of the datasets [98]. Metadata are data about the data. They provide important information for any dataset regardless of the domain they belong to, e.g., geographic, music, publications, movie, etc. For many use cases it is important to be aware of the descriptive information and simple statistics about the data such as dataset size, relations in the data, the topics of the dataset, etc. Although data profiling comprises a broad range of descriptive information, throughout this thesis we will focus on three aspects of profiling; topic-based profiling, schema-based profiling and linkage-based profiling. The descriptive information given by these three aspects of profiling is fundamental for dataset understanding and decision-making. **Topic-based profiling** gives an overview about the subject or the theme that a dataset or parts of a dataset belong to. As a prerequisite for being able to understand and consume a dataset, it is necessary to know its topics. The **schema-based profiling** gives an ontology-driven summarization of the dataset focusing on the relation between classes used in the dataset. As datasets published as Linked Data are large, data consumers will go under time-consuming activity if they decide to check manually each dataset to find in which one, for example the classes are described with richer set of properties, considered as the number of properties describing a class. Still, schema-based profiling will help users not only for the decision-making process but also in other use cases as in 5. Finally, **linkage-based profiling** provides information about equivalent entities. Entities can be explicitly stated as equivalent when they refer to the same real world object. Linkage-based profiling is able to provide the list of equivalent entities among two or more datasets and the total number of links connecting these entities. Different datasets in the LD cloud contain similar or different information for the same entity. Thus, before deciding which dataset to use, data consumers need to know the overlap between datasets, so they are able to collect all the available information for the given entity.

The profiling approaches provided by this thesis in the context of Linked Data is important not only for the dataset understanding and exploration problem but is also very important for different use cases such as:

Landscape view. Most of the datasets published do not provide descriptive information, thus it is difficult to understand their content. Data profiling techniques can help to identify some core knowledge patterns (KP) which reveal a piece of knowledge in a domain of interest [113]. The analysis of the usage of the core classes and properties used in a dataset can help in understanding its content [150]. The summarization of an ontology by extracting the most relevant RDF classes according to a re-ranking strategy is proposed in [149]. The usage of relevant RDF classes is assessed in terms of their centrality to the ontology. The three aspects of profiling considered in this thesis provide a landscape view of the dataset.

Ontology / Dataset integration. Ontologies published on the Web, even for datasets in similar domains can have differences. The basic activity in ontology integration is the creation of mappings between classes and properties among two ontologies [67]. Data profiling techniques can help to understand the overlap between ontologies and help in the process of ontology integration.

To perform a dataset integration process, one should consider ontology mapping, the process of discovering relationships between ontologies. A proposed methodology to derive ontology mapping using examples that are a finite set of negative and positive examples could be found in [130]. Profiling techniques can reveal mappings between classes and properties. For example following the links provided by the linkage-based profiling, can help in the process of class matching. Moreover, these links can serve as input for the process of property matching, as equivalent entities share equivalent properties.

Identification of quality issues. Many data producers produce Linked Data datasets that are of a high quality. However, there are also many datasets, which were extracted from unstructured or semi-structured information being vulnerable for quality issues [78]. Data profiling tools allow the inspection of large datasets for detecting quality issues, by identifying cases that do not follow business rules, outliers detection or residuals [98]. Data profiling helps to identify quality rules and requirements that will support a more thorough data quality assessment in a later step [145]. Schema-based profiling supports the data quality assessment by identifying incongruences in the data such as counterintuitive relations between entities belonging to the same

class, usage of properties that do not conform to the constraint defined by the ontology, etc. Also the linkage-based profiling can help in the process of quality assessment by identifying wrong equivalent links among datasets.

Data visualization for summarization. The main goal of data visualization is to communicate information clearly and efficiently using graphics. Visualizing Linked Data is challenging because the datasets in the LD cloud are large. Thus, the amount of data to be visualized is overwhelming and data consumers do not have enough time to visually inspect datasets. Profiling techniques can support data visualization tools to visualize large multidimensional datasets by displaying only a small and concise summary of the most relevant and important features. This will enhance the comprehension of the user by allowing him to dig into the data by zooming in or out of the provided summary. An example of such use case is Schemr [39], an ontology visualization search engine algorithm based on a combination of text search and ontology matching techniques that allows a user to visualize ontologies by keyword search. Another example is ZoomRDF [147] which allows more resources to display while zooming to a single class, and still maintaining the overview structure of the data. Schema-profiling provided in this thesis help users visualize and navigate summaries for large datasets by dispalying only the most specific relations among classes.

Query optimization. Traditional systems assume that users have good knowledge about the meaning and the content of the dataset when querying the system and are certain that a particular query is the one they wanted to pose [68]. Often users have to execute many explorative queries until they find the information they were looking for because they have no knowledge how the information is represented in the dataset. Data profiling can help users in optimizing query completion or formulation by providing a summary of the content of the dataset [98]. Schema-based profiling constructs summaries that provide information about relations in the data, thus helping users in query formulation tasks. Suppose a user is interested in quering all instances that belong to the class *Actor* who are married with instances of the class *Model*. In this case the user can easily see how this relation is modeled in the summary so she can easily write the query. Moreover, data profiling techniques produce statistics about the size of the dataset, number of entities (instances), classes, properties, etc. This information can be used

to estimate the cost of a query plan and help to determine the most effective way to pose the right query in order to extract efficiently the information we want.

Schema discovery. Schema complexity leads to difficulties to understand and access datasets. Even though schema or ontologies are used to define classes and properties used in a dataset, often can be very hard to understand. Schema understanding has the challenge of understanding the complexity of the structure defined by the ontology as well as the semantics of the labels used to describe classes and relations [54]. Schema summaries provide users a concise overview of the entire schema despite its complexity. By providing a summary of the relations between *types* (such as classes), our schema-based profiling can help users understanding large and very complex schemas.

Entity summarization. The summarization of particular entities can play a central role for semantic search engines and semantic recommender systems [131]. This field of research addresses the problem of ranking features according to their importance for the task of identifying a particular entity. Finding features that best represent the topic/s of a given dataset can not only help the topical classification of the dataset but also understanding the semantics of the information found in the data. RELIN [41] which is an entity summarization framework aims at finding the central topics in a given dataset considering relatedness and informativeness of the description of data elements based on linguistic and information theory concepts. Our topic-based profiling can help with the process of topical classification of LD datasets. The occurrences of the relations in our summaries produced by schema-based profiling can support the summarization of particular entities in the dataset.

Data Analytics. Data analytics is the process of examining data in order to draw conclusions and insights about the particular business the data belong to. Before analyzing such data, data analyst needs profiling results in order to understand what kind of data they are going to analyze so they can appropriately configure analytical tools [98].

Consuming Linked Data is not straightforward, because it is hard to decide which dataset to use as we miss a profile about the content of the data. Thus, to consume Linked Data we have to deal with the challenge of developing techniques

or approaches that provide metadata or descriptive information about the dataset content, so users can exploit them in real use cases. Traditional data profiling tools and techniques can not be applied in Linked Data because of their heterogeneous nature [98] and the adoption of the existing profiling approaches is not straightforward. The topic of profiling Linked Data has not yet received sufficient attention from the Linked Data community and it poses a number of unique challenges.

- Linked Data comes from different autonomous sources and are continuously evolving. The descriptive information or the metadata may depend on the data publishers' will. Often publishers are more interested in publishing their data in RDF format without taking care very much about the metadata. Moreover, data publishers find difficulties in using appropriate terms for the data to be described. Apart from a well-known group of vocabularies, it is hard to find vocabularies for most of the domains that would be a good candidate for the dataset at hand [151]. For the above reasons, profiling techniques must deal with the evolving nature of Linked Data.

- The number of datasets published as LD is constantly growing and includes different topical domains making the LD a heterogeneous environment. Thus, this variety of data implies the development of profiling approaches that will be applicable cross-wise domain.

- In LD cloud there are published billions of triples that poses very high performance and scalability demands. Managing large and rapidly increasing volume of data is a challenge for developing profiling techniques that scale well with the volume of data in the LD cloud.

- In the Linked Data cloud there is a high volume of data. For this reason data consumers need also automatic profiling approaches that allow dataset understanding and exploration.

- Due to the evolving nature of the datasets in the LD cloud there is also a need of developing profiling techniques that use previous profiling results to profile only the changed data, which is referred to as incremental data profiling [98].

- Profiling techniques should deal with the structural, semantic and schema heterogeneity of the LD datasets, thus should be able to deal with different

degrees of heterogeneity in different datasets (especially for the use cases of ontology and data integration).

- Searching through or browsing LD cloud is hard, because the metadata are often not structured and not in a machine-readable format. A machine-readable format is a standard computer language that can be read automatically by a computer. For example if a data consumer wants to select all datasets that belong to the media category, she faces the challenge of having the metadata describing topic not in a machine-readable format. The topic of the datasets in LD cloud was manually assigned and it is not represented in a machine-readable format. Profiling techniques should provide results in a machine readable format so that they can be used in further analysis by computers.

- Lack of unified dimensions, metric and benchmarks makes it difficult to evaluate different methods and architectures for different profiling tasks.

It is very important to identify adequate techniques for profiling Linked Data. Despite the need of having descriptive information and metadata for the exploration process of LD cloud, few efforts are taken for the development of such techniques and tools. Data profiling in the field of Linked Data is relatively new, and because of all the challenges mentioned above, traditional profiling tools can not be applied. When browsing a dataset in the cloud, users are usually fed with its metadata, in case a dataset provide this information, but lack for an insight into its content, thereby having difficulties in determining its relevance.

The profiling results developed during this thesis are fundamental and will support users and data consumers with the decision-making and help them understand and explore Linked Data. The dataset profile can be used not only to detect if the dataset is useful or not, but also to provide useful information for data quality assessment and integration phases. In order to make the profiling process smooth for any user, requiring minimal effort, and be able to apply it for any dataset, we have to consider three fundamental criteria: (i) it should consider different features that describe a dataset, (ii) it should be generated independently of the domain, thus it can be applicable to any dataset regardless of the domain it belongs to, and (iii) it should be generated automatically.

1.1 User Scenario

Ms. Eamla, is an app developer passionate about music who wants to create a new music app *MusicBeats*. She also wants to stay up-to-date with news coming from Twitter account of music artists. She is looking for datasets that contain rich information about the music domain. Ms. Eamla knows that Linked Data cloud is a gold mine because it provides many datasets covering different topical domains. She also knows that datasets published as Linked Data are represented in RDF, a language which is expressive enough. Looking at the Fig. 1.1, she understands that there are more than 1000 datasets covering different topical domains represented by the different colors of the cloud.

She finds that the number of datasets under the *media* domain is 21. But she realizes that there are also datasets under the *cross-domain* or *user-generated content* which might also include media data. Because the LD cloud provides only one topic, the most relevant one, for Ms. Eamla is also important a multi-topic classification of LD datasets. She also found other datasets in the Web, but she does not know their topic/s. In this situation she has two choices; (1) to manually check the content of the dataset and assign it a topic or (2) to give up the topic assignment process and consider all the available datasets for her use case. The first option is very time-consuming, while considering the second option she risks overloading her app with irrelevant data. Thus, these options are not suitable for her goal. To assign the topic of the dataset she needs automatic approaches which given an RDF dataset as input, be able to produce one or a set of topics, that describe the dataset content.

Once she finds the topic of the dataset, she needs to know more about its structure. For example, she found that two datasets *LinkedBrianz*[4] and *DBpedia*[5] have at least *media* as one of their topics. Even though knowing the topic of these datasets can reduce the searching space, still the app creator has difficulties in knowing the semantics and structure of the two datasets and their coverage in terms of instances. Thus, this decision-making raises many questions such as: (1) How are music artists/ groups/ songs described in these two datasets? (2) How many instances are covered? (3) Is there any incongruence in the data? She can answer these questions by looking at the ontology. But ontologies might be large

[4]http://linkedbrainz.org/
[5]http://dbpedia.org

and underspecified. Thus, by only looking at the ontology, our music app creator is not able to answer the above questions. These answers can be collected with explorative queries, but at the price of a significant server overload for data publishers and high response time for data consumers.

After looking at the relation inside the two datasets, she finds that there are some entities of a given class in *LinkedBrianz* dataset that are described with richer set of properties. Rather, in *DBpedia* instances belonging to the same class are not described with a rich set of properties. For her, it is very important to have complete information describing an instance, thus she needs to find equivalent instances in different datasets in order to enrich her database for the app. As above, she also has two options; (1) to manually check if instances of different datasets are equivalent or not, and (2) to apply techniques for instance matching. If she considers the first option she will go under a very time-consuming process as checking 8 million resources is an unbearable process, while if she considers the second option she has to deal with also some manual work in setting up the actual approaches for instance matching as we will show in this thesis.

Due to these challenges Ms. Eamla might collect wrong datasets for her app, leading to a non successful app, despite the significant time she spent on building it.

1.2 Problem statement

At the beginning of the Linked Data era, the main focus of the community was on publishing datasets following the five star scale (see section 2.6) while nowadays they are more focused on publishing RDF data with the focus of publishing metadata and good quality data, which allows the reliability and usability of datasets in real use cases [4]. Therefore, the metadata creation, that describes the content of the dataset is crucial for data providers. On the other hand, metadata are also important for data consumers or applications that will rely their work on these metadata and help in decision-making problem. Data publishers want to publish the dataset profile, which contain metadata such that the dataset could be found and aggregated by search engines application. Search engine applications by themselves want to discover detailed but not redundant descriptive information about

the datasets they find. This information can be provided using VOID[6] vocabulary. In the actual state of the LD cloud, Fig. 1.1, only 14,69% of the datasets provide VOID metadata [124]. 90,4% of the datasets that provide descriptive information or metadata are limited in providing only the most basic information about the dataset such as *name, license, publisher, creator* or *data dump* but they do not provide any information about the content of the dataset, information such as what is its topic, how are the relationships between types described in the dataset, how many equivalent entities are among datasets, etc.

As we will describe in the following of this thesis, ontologies and vocabularies play a crucial role in understanding the dataset. In order to ease the interpretation of the data one of the best practices for publishing Linked Data indicates the use of existing vocabularies and ontologies. The top five vocabularies used by the datasets of LD cloud are: RDF (98,22%), RDFS[7] (72,58%), FOAF[8] (69,13%), DCTerms[9] (56,01%) and OWL[10] (36,49%) [124]. From these statistics we can deduce that these vocabularies are used by different datasets regardless of their topic. Linked Open Vocabularies[11] (LOV) is an observatory of the semantic vocabularies' ecosystem [140] with the aim to promote the reuse of well documented vocabularies by making available descriptive information about them such as interconnection, versioning history, etc. The analysis of the vocabularies by their creation date indicates a peak in 2011, while from 2011 the number of created vocabularies has decreased because the focus of the data publishers moved from publishing new vocabularies in adopting best practices, thus using existing vocabularies.

Despite the effort done so far by the Semantic Web community [149, 85, 58, 6, 143, 112] still exploring the dataset content by looking only at the ontology is not sufficient. Ontologies are of different size containing many classes and properties such as *DBpedia ontology*[12] or *OBO* (Ontology for Biomedical Investigations)[13] or very small, containing just a few classes or properties such as *bibo* (The Bibliographic

[6] https://www.w3.org/TR/void/
[7] https://www.w3.org/TR/rdf-schema/
[8] http://xmlns.com/foaf/spec/
[9] http://dublincore.org/documents/dcmi-terms/
[10] https://www.w3.org/TR/owl-guide/
[11] http://lov.okfn.org
[12] http://dbpedia.org/ontology/
[13] http://lov.okfn.org/dataset/lov/vocabs/obo

Ontology)[14] and *oad* (Ontology for Archival Description)[15]. For example, the *DBpedia ontology* has 775 classes and 2861 properties thus it is not straightforward to understand the semantics of some entities found in the data. Also, ontologies might be underspecified, for example for some properties we do not have any restriction about their use [134]. The underspecification might occur as a result of the use of general purpose vocabularies such as DCTerms or intentionally left by the data modeling experts to not restrict the use of such property. The ontology of the *L3S DBLP Linked Data*[16] does not have a property that connects the coauthors directly, a property that occurs very often in this domain. In spite of the fact that there are connections inside this dataset to represent this relationship, it is not trivial to discover them because this information is encoded in the data [5].

Moreover, to understand better a dataset a user faces the problem of implicit information in the ontology. OWL makes this information explicit by defining inference. Some datasets materialize inference results in a database so that SPARQL queries can be executed efficiently, thus supporting users in the exploration process. For example, the *DBpedia* dataset makes use of materialized inference thus allowing efficient extraction of specific data subsets. In contrast to *DBpedia*, *LinkedBrainz*, does not make use of materialized inference thus it is more difficult to extract specific subsets of the data. Most of the dataset in the LD cloud do not make use of materialized inference, thus the exploration is not efficient, resulting in a time-consuming activity.

In order to allow applications to understand as much data as possible, relations between classes and properties among different vocabularies should be defined. These relations may define restrictions or equivalences. The top ten properties used to link classes between more vocabularies belong to the RDFS and OWL ontology [124]. In LD the most used property to link to other vocabularies is the `rdfs:range` used by 9,8% of vocabularies [124]. We can observe also in LOV that there are around 1235 links from other vocabularies to RDFS. 1,6% of vocabularies in LD, use `owl:equivalentClass` to relate two equivalent classes belonging to two different vocabularies or ontologies. Dataset owners usually use various and different ontologies to describe their dataset. Therefore, to understand a dataset, data consumers face the problem of knowing the semantics of different ontologies and their mappings.

[14] http://lov.okfn.org/dataset/lov/vocabs/bibo
[15] http://lov.okfn.org/dataset/lov/vocabs/oad
[16] http://dblp.l3s.de/d2r/

The other best practice of Linked Data promotes the idea of improving interoperability and aggregation between datasets in the Web [25]. Data linking is the task of finding entities that refer to the same real-world object by measuring the degree of similarity to their entity descriptors. However, since datasets generally use different identifiers for an entity, their information can not be easily collected. This is an important task nowadays because by following links, a data consumer can easily find and retrieve more data. The interlinking of diverse datasets in the "Web of Data" enables users to navigate from one dataset to the other in the same way users currently navigate from one webpage to another in the "Web of Documents" [70].

While Linked Data accounts for more than a thousand dataset, publishing the number of links between them is several orders of magnitude smaller and by far more difficult to maintain [9]. Fig. 1.1 gives an overview about the linking information in the LD cloud. Interlinking information in the LD cloud is too shallow to realize much of the benefits promised by Linked Data. The central part of the cloud is heavily interlinked with *DBpedia, GeoNames*[17] and *Foaf profiles*[18] being the central datasets for linking entities. Moreover, we could find that the instance describing *Gaillimh* belonging to *GeoNames* is not linked with the instance of *Gaillimh* in *LinkedGeoData*[19] although they refer to the same real-world object. Also, there are cases when entities belonging to different real world objects are linked together. Because of these limitations the LD cloud will suffer from the same kind of problems as the "Web of Documents", consequently the vision of the Semantic Web will fall short [70]. Data interlinking is a well studied problem in the Semantic Web area [55, 79, 28, 62, 51] and two survey papers [116, 122] summarize the effort done in the area of data linking.

However, there are many issues still to be resolved. As Linked Data are huge data, there is a need for developing techniques and tools that can find equivalent entities automatically without any knowledge about the content of the dataset they belong to. Most of the tools developed so far are semi-automated such as Silk [142] or LIMES [103] and need a configuration file to be set up. Although these tools achieve a high performance they require skilled human data publishers going through error prone and time-consuming process for manually creating rules mapping between entities of two datasets. Often datasets make use of different

[17] http://www.geonames.org/
[18] https://datahub.io/dataset/personal-homepages
[19] http://linkedgeodata.org/About

RDF vocabularies and OWL ontologies, while the existing techniques and tools assume data to have only one schema or ontology. As Linked Data are evolving, also links and mappings should evolve, imposing the need of scalable and easy to maintain tools.

To decide if a dataset is useful for our use case at hand or not, we have to access the descriptive information provided by data publishers. To make such a decision we have to face several challenges, described in our user scenario example:

Availability of datasets profile. The process of finding the right dataset for our use case at hand can be supported by the dataset profile along with the dataset. Some datasets may provide accurate and up to date metadata, but most of the datasets in the LD cloud provide metadata that are incomplete, inaccurate or not up to date. Thus, the decision-making is not straightforward because datasets profiles are not always available or of good quality.

Diversity of dataset descriptors. Data publishers use different data formats and modeling languages, e.g., different vocabularies, to represent their data. Moreover, even though datasets use similar vocabularies, they differ in the semantics of the terminology they describe. This is due to the autonomous nature of data publishing activity in Linked Data and the fact that these datasets are created by different people with different objectives.

Extraction of dataset profile. The problem of dataset understanding can be supported by tools and techniques that extract descriptive information about the dataset, in cases when this information is not available or validates the profiles that come along with the data. Users that need to use some data might not be aware of different tools and techniques that exist in this field. Most of the existing tools and techniques solve a specific task such as for example topic extraction, but can not support building a complete profile of the current dataset. Last but not least, the process of dataset profile extraction should be automatic, with less supervision by the user.

The lack of the dataset profiles that provide some descriptive information about the data, is an important challenge and obstacle in dataset consumption.

The goal of this thesis is to develop automatic techniques, which enable dataset consumers to profile datasets and improve the descriptive information of Linked Data cloud. The main research question solved during the course of this thesis is:

Given a heterogeneous Linked Data corpus, data profiling can be provided in a domain independent and in an automatic manner through techniques that:

- Could support topical information extraction and classification
- Provide a dataset summary
- Reveal quality issues in the data
- Extract the number of links among equivalent entities between datasets without prior knowledge about the content of the dataset

1.3 Main contributions

To solve the main research question during the course of this thesis we make the following contributions:

Detailed literature review. Linked Data is a relatively new field and profiling Linked Data techniques and tools developed so far are flat and therefore lack the required richness. We conducted a detailed literature review following the proposed review procedure as in [75]. First, we provide a survey of the existing tools used to cover different profiling tasks based on six criteria. This survey provides a broader context of profiling Linked Data which can serve as a starting point for researchers working on similar area. Furthermore, we conducted a detailed review of the most important approaches, tools and techniques to perform topic-based profiling, schema-based profiling and linkage-based profiling of Linked Data. Finally, for each of them we provide a summary of the remarks and open issues (Chapter 3).

Automatic topic classification of LD datasets. As a main contribution of this thesis, we investigated the problem of the automatic topical classification of LD datasets. Up till now, topical categories were manually assigned either by the publishers of the datasets themselves or by the authors of LD cloud. We provide an approach, which takes as input different features that characterize a dataset and assigns a topic. Furthermore, we investigated the problem of assigning more than one topic, as many datasets can have more than one topic (Chapter 4).

Building the gold standard for multi-topic classification of LD datasets. As part of the single-topic classification of LD datasets, we considered as gold standard the information that currently exist in the LD cloud, while for the multi-topic classification we build a gold standard which we make available for further research in this area (Chapter 4).

Evaluation of schema-based summarization of LD datasets. ABSTAT is an ontology-driven Linked Data summarization approach developed internally at our department, to help users understand datasets at hand. It extracts a summary that represents the relations between types in the dataset. Throughout this thesis the experimental part was developed, which provides scientific evaluation that this tool helps users to understand the datasets using query completion tasks. Altogether with the summary, ABSTAT provides statistics that allow users to better explore and understand Linked Data. The summary produced by ABSTAT can help to detect quality issues in the data (Chapter 5).

Automatic similar linkage discovery framework. During this thesis we exploited the actual use of the most known property (`owl:sameAs` to link equivalent entities on the LD cloud and investigate to which extent can we automatically find equivalent entities belonging to different datasets without prior knowledge about their content. For this goal we developed a framework which identifies ambiguities and suggest possible inconsistencies and incompleteness among links that exist in the LD diagram (Chapter 6).

1.4 Thesis outline

The remainder of the thesis is structured in seven chapters which are described in the following:

- **Chapter 2** introduces the core concepts that provide the basic scientific background required for the reader to understand the thesis. It gives an introduction of the Semantic Web standards such as RDF data model, different syntaxes for RDF (RDF serialization formats), ontologies and different languages used to develop them and finally we introduce SPARQL query language used to query RDF. At the end, we provide the description of Linked Data, data published and linked in the Web using RDF.

- **Chapter 3** provides an overview of the state-of-the-art in three areas that are part of this thesis. First we describe the challenges why techniques that profile relational data can not be applied in Linked Data. Afterwards we provide a qualitative comparison of nine tools used to profile Linked Data based on six different criteria: availability, automation, scalability, licensing, usability and maintenance. Finally, we provide a description of the most known approaches used for topic profiling, schema-based profiling and linkage profiling. We provide details on the cases when to apply each approach, what is the input and the output along with the challenges to adapt each of them.

- Topic profiling is described in **Chapter 4**. This chapter is divided in two parts. In the first part we describe the problem of single-topic classification of LD datasets while in the second part we describe the problem of multi-topic profiling. Firstly, we introduce the reader to the importance of topic profiling together with the challenges still presented in this area. Secondly, we describe the data model used for this purpose. Thirdly, we provide details about the approach and different techniques used for the topical classification of LD datasets. Therefore, we present the experiments and results and finally discuss different aspects of the results achieved for the topic profiling.

- In **Chapter 5** we introduce ABSTAT an ontology-driven Linked Data summarization approach, used to mitigate the problem of dataset understanding. We provide a description of the summarization model and the summary extraction pipeline together with the algorithm used in ABSTAT. After, we describe the three user interfaces ABSTAT has at the moment of writing this thesis. Finally, we present different experimental evaluation used to evaluate our summaries. We measure the compactness and the informativeness of the summaries. The compactness is measured by the reduction rate while informativeness is measured by the means of a user study and a quantitative evaluation.

- **Chapter 6** provides details about the problem of finding similar objects among heterogeneous data sources. We first introduce the importance and the challenges still present in the area of linking equivalent entities. Secondly, we describe the proposed approach, giving details for each of the four processes of the linkage pipeline. Thirdly, we provide two metrics used for

string similarity and numeric similarity. Finally, we describe our experiments to measure the performance of the proposed approach together with the discussion of the results.

- In **Chapter 7** we summarize the results and compare them to the motivation presented in this chapter. At the end we outline the expected future work and impact of this research topic.

Chapter 2

Foundations and Technical Background

In the following chapter we give an overview of the foundations and the technical background for the users to understand the work presented in this thesis. In section 2.1 we introduce the reader with the basic concepts of Semantic Web. Section 2.2 gives an overview of RDF data model and its components. We introduce vocabularies and ontologies in section 2.3 and describe the two most know languages for building ontologies; RDFS and OWL. While SPARQL query language is introduced in section 2.4, triplestores in section 2.5 and principles of Linked Data conclude this chapter in section 2.6.

2.1 Web and Semantic Web

2.1.1 World Wide Web

The World Wide Web (WWW) [22] is a global information space consisting of information shared from numerous sources. Since its first implementation, known also as Web 1.0, the Web has evolved rapidly. In the Web 1.0 users could only search and read documents, while in the Web 2.0 users are seen to interact with each other or contribute to the content. The Web 3.0 enables data to be connected from different sources and to be understood by computers so that they can perform increasingly, complex and sophisticated tasks for our benefit. The Web 3.0 is

also referred to as the Semantic Web. Web pages are formatted and annotated with Hypertext Markup Language (HTML). Uniform Resource Identifiers (URI) identify uniquely Web documents which can be accessed through specific protocols such as Hypertext Transfer Protocol (HTTP) [57]. Hyperlinks allow users to navigate and access different documents through Web browsers. Usually, data published on the Web are made available in raw dumps in different formats such as CSV, XML or marked up as (HTML) tables [26]. The data in this format are not in a machine-readable format, thus they can not be processed by computer, making it hard for the user to get the answer if they make any query. Also, documents published on the Web are of non expressive format, i.e (HTML), thus entities described in these documents can not be connected by typed links[1] to related entities in other documents [26]. Since its creation, the Web has evolved from an information space of linked documents to one where both documents and data are linked. These data are published adopting a set of best practices for publishing and connecting structured data on the Web know as Linked Data [26].

The data published in Linked Data should be published adopting W3C standards in Resource Description Framework (RDF) format and made available for SPARQL endpoint queries. The adoption of Linked Data best practices has led in an extension of the Web, where the data are interlinked with each other using typed links. These documents contain more machine-oriented semantic information known as the Semantic Web [23].

2.1.2 Semantic Web

Sematic Web Stack in Fig. 2.1, represents the capsulated architecture of the Semantic Web [24]. The main purpose of the Semantic Web is to convert the "Web of Documents" into the "Web of Data". It shows how technologies that are standardized for Semantic Web are organized to make the Semantic Web possible. It also shows how Semantic Web is an extension (not replacement) of classical hypertext Web. The technologies from the bottom of the stack up to OWL are currently recommended by W3C and accepted to build Semantic Web applications. The bottom layers of the stack are well-known for the hypertext Web and are reported without any change for the Semantic Web. The top layers, contain technologies

[1]The most known use of a link is to retrieve another Web resource, however links can express other types of relationships between resources and have one or more link types specified in their source anchor.

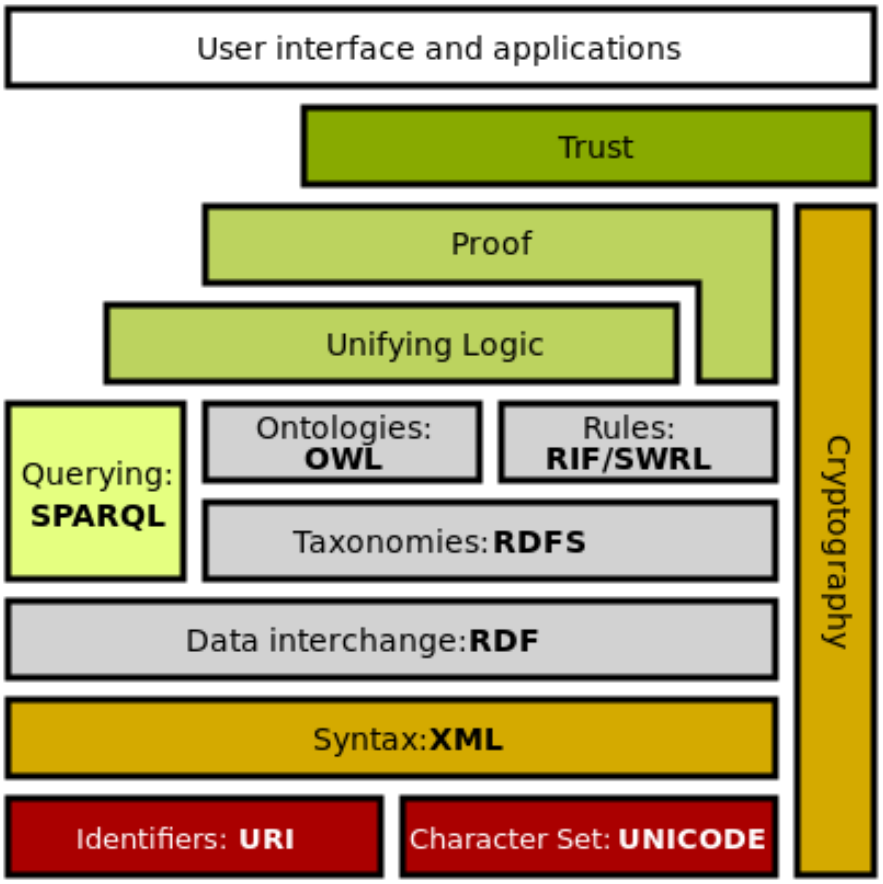


FIGURE 2.1: Semantic Web Stack

which are not standardized yet, while in this thesis the middle layers, RDF, RDFS, OWL and SPARQL will be discussed in the following.

To empower applications to process information on the Web, data should be expressed in a standard format. The Resource Description Framework (RDF), is a framework for representing information in the Web [76]. RDF stands for **R**esource pages, documents, people, everything that is identified by a URI, **D**escription attributes, description or relations, and **F**ramework model, languages or description for resources. This mechanism for describing resources is a major component in the W3C's Semantic Web activity: an evolutionary stage of the World Wide Web in which automated software can store, exchange, and use machine-readable information distributed throughout the Web, in turn enabling users to deal with the information with greater efficiency and certainty [100]. In the following we will describe in more details the representative languages of Semantic Web, part of middle layer, RDF, RDFS, OWL and SPARQL.

2.2 Resource Description Framework: Data Model and Syntax

2.2.1 Data Model

RDF has a simple data model which allows applications to easily process and manipulate it. Having a simple data model and the ability to model disparate, abstract concepts additionally contributed to an expansion of its use in knowledge management applications unrelated to Semantic Web activity.

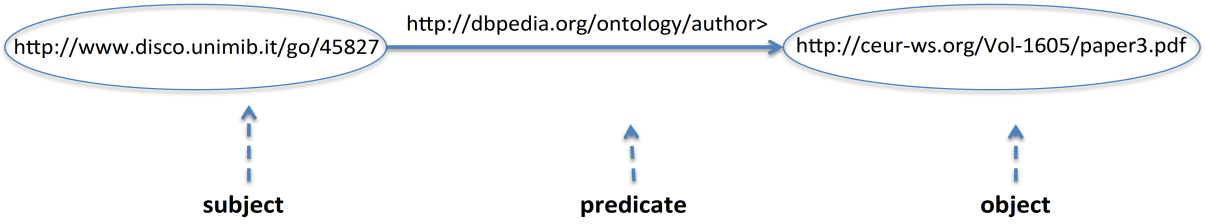


FIGURE 2.2: RDF triple example

The basic construct in RDF is an RDF statement. Every statement is represented by a triple which consists of three terms in the form of `subject-predicate-object`.

The *subject*, identifies the object of the triple that is being described. The *predicate* (property) describes the relationship between subject and object and describes some aspects of the subject, while the *object* is the target or the value of the RDF triple. Every triple represents a single property of the subject. Triples can be visualized as node and arc diagrams as in Fig. 2.2. The subject and the object are denoted as nodes, while the property is denoted as labeled arc. In the Fig. 2.2, the subject belonging to "Blerina Spahiu" <`http://www.disco.unimib.it/go/45827/`> has the property <`http://dbpedia.org/ontology/author/`> whose value is a paper <`http://ceur-ws.org/Vol-1605/paper3.pdf`>. Often, instead of using the whole URI, we use a short format for writing them. These short formats are called prefixes or namespaces. For instance for the example above we can use `disco` instead of `http://www.disco.unimib.it/` or `dbo` instead of `http://dbpedia.org/ontology/`, etc.

In the Semantic Web we refer to things in the world that are described by an RDF term as a **resource**. A **resource** can be any real world entity such as a person, a city, a restaurant, a web page, etc., and abstract concepts such as terms,

classes or property types. These resources are identified uniquely by the URI. A **blank node** is a node in RDF which does not have an identifier and it can not be referenced from outside. In cases when a blank node is found by an RDF parser, the latter generates an internal unique identifier to assign to the blank node. As blank node identifiers are often only unique within the scope of the dataset in which they occur, it is possible that distinct blank nodes in different datasets use the same blank node identifier. Thus, when RDF graphs are merged within implementations, it might be necessary to rename blank nodes in order to avoid collisions [76]. In summary, blank nodes should be avoided unless they are structurally necessary. Literals are used to represent property values such as texts, numbers, and dates and are of two types; *plain* and *typed* literals. The plain literal is a string associated with a language tag [12]. A language tag (e.g. "Athens"@en or "Atene"@it) indicates a language such as English or Italian. The typed literal is a string associated with a datatype URI. A datatype URI is defined by the XML schema and indicates dates, integers and floating-point numbers, e.g. "1986-10-09"^^xsd:date.

- the *subject* is an RDF URI reference or a blank node
- the *predicate*is an RDF URI reference
- the *object* is an RDF URI reference, a literal or a blank node

Definition 2.1. (RDF Triple). Given an infinite set $\mathcal{URI}$ of URIs, an infinite set $\mathcal{BN}$ of blank nodes, an infinite set $\mathcal{PL}$ of plain literals, and an infinite set of $\mathcal{TL}$ of typed literals, a triple (s, p, o) $\in (\mathcal{URI} \cup \mathcal{BN}) \times \mathcal{URI} \times (\mathcal{URI} \cup \mathcal{BN} \cup \mathcal{PL} \cup \mathcal{TL})$ is called an RDF triple where s, p, o represents the subject, predicate and object of the triple.

RDF triples are of two types depending on the type of the object:

- *Object type triple*, where the object of a triple is a URI. Furthermore, object type triples can be distinguished in internal, where the link connecting two resources belong to the same dataset and external, where the link connects two resources that belong to different datasets. An example of the object type triples is the following:

  <`http://www.disco.unimib.it/go/45827/`>

<http://dbpedia.org/ontology/affiliation>

<http://www.disco.unimib.it/>

As the subject and the object are of the same dataset, this is an internal object type triple. An example of external object type triple is the following:
<http://www.disco.unimib.it/go/45827/>

<http://dbpedia.org/ontology/author/>

<http://ceur-ws.org/Vol-1605/paper3.pdf>.

In case there exists a triple where its subject to dataset A and its object belongs to dataset B, for dataset A the link is *outgoing* while for dataset B the link is *incoming*. In the example above for <http://www.disco.unimib.it/> the link is outgoing while for <http://ceur-ws.org/> the link is incoming.

- *Data type triple*, is an RDF triple where the object is of type literal. An example of data type triple is shown in Fig. 2.3:

  <http://www.disco.unimib.it/go/45827/>

  <http://dbpedia.org/ontology/birthDate>

  "1986-10-09"<http://www.w3.org/2001/XMLSchema#date>

Definition 2.2. (RDF Graph.) RDF Graph $\mathcal{G}$ is a finite set of RDF triples $(\mathcal{URI} \cup \mathcal{BN}) \times \mathcal{URI} \times (\mathcal{URI} \cup \mathcal{BN} \cup \mathcal{PL} \cup \mathcal{TL})$.

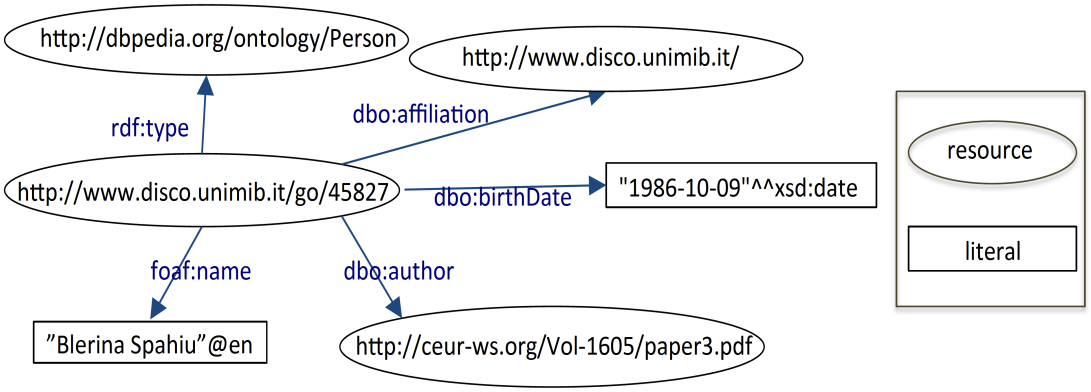


FIGURE 2.3: RDF Graph example

The RDF graph itself represents a resource, which is located at a certain location on the Web and thus has an associated IRI, the graph IRI.

Definition 2.3. (RDF named Graph). An RDF named graph $\mathcal{UG}$ is an RDF graph which is identified by a URI.

There are a number of synonyms being used for RDF graphs, all meaning essentially the same but stressing different aspects of an RDF graph. The RDF document is used when we refer to an RDF graph from a file perspective, knowledge base (KB) when we refer to a collection of facts, vocabulary when we refer to a shared terminology, and ontology when we refer to a shared logical conceptualization [10].

Throughout this thesis we refer to a dataset as a set of RDF triples. In chapter 5 we will give a more formal definition about a dataset.

2.2.2 RDF serialization formats

In the following is presented a family of alternative text-based RDF serializations whose members have the same origin, but balance differently between readability for humans and machines. Since different platforms work better with different data formats, RDF serialization is an important aspect.

N-triples

N-Triples is a line-based, plain text serialisation format for RDF [18]. Each RDF triple is written as a separate line, where each URI is written between angle brackets (< and >) and terminated by a period (.). Typically, files with N-Triples have the .nt extension. In Fig. 2.4, is given an example of the N-Triples format.

RDF/XML

One of the formats for representing RDF triples is RDF/XML. The RDF/XML Syntax Specification defines a normative syntax for serializing RDF graphs as XML documents [17]. Nodes and predicates in RDF have to be represented in XML terms; element names, attribute names, element contents and attribute values [76]. Fig. 2.5 shows an example of the RDF/XML format.

N3

N3 has been developed as a language for expressing data and rules. It extends RDF with features such as variables, universal and existential quantification. N3 is much more compact and readable than XML/RDF notation [18].

Turtle

```
01. <http://www.disco.unimib.it/go/45827> <http://dbpedia.org/ontology/birthDate>
    "1986-10-09”^^ <http://www.w3.org/2001/XMLSchema#date>
02. <http://www.disco.unimib.it/go/45827>  <http://www.w3.org/1999/02/22-rdf-syntax-
    ns#type> <http://dbpedia.org/ontology/Person>
03. <http://www.disco.unimib.it/go/45827>  <http://dbpedia.org/ontology/affiliation> <http://
    www.disco.unimib.it/>
04. <http://www.disco.unimib.it/go/45827> <http://xmlns.com/foaf/0.1/name> ”Blerina
    Spahiu”@en
05. <http://www.disco.unimib.it/go/45827> <http://dbpedia.org/ontology/author> <http://
    ceur-ws.org/Vol-1605/paper3.pdf>
```

FIGURE 2.4: RDF/N-triples example format

```
01. <?xml version="1.0" encoding="UTF-8"?>
02.
03. <rdf:RDF
04.       xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
05.       xmlns:dc=“http://purl.org/dc/elements/1.1/”
06.
07.       <rdf:Description rdf:about="https://en.wikipedia.org/wiki/Milan">
08.             <dc:title>Milan</dc:title>
09.             <dc:publisher>Wikipedia</dc:publisher>
10.             <region:population>10000000</region:population>
11.             <region:principaltown rdf:resource="https://it.wikipedia.org/wiki/Lombardia"/>
12.       </rdf:Description>
13.
14. </rdf:RDF>
```

FIGURE 2.5: RDF/XML example format

```
01. @prefix dbo: <http://dbpedia.org/resource/> .
02. @prefix foaf: <http://xmlns.com/foaf/0.1/> .
03. @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
04. @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#
05.
06. dbo:George_Clooney rdf:type foaf:Person;
07. foaf:age ”55"^^xsd:int;
08. foaf:birthday "1961-06-05"^^xsd:date;
09. foaf:name "George Clooney"@en .
```

FIGURE 2.6: RDF/N3 example format

The Turtle [17] syntax for RDF is a text-based serialization format. It is a subset of N3 format.

2.3 Vocabularies and ontologies

In the context of Semantic Web, vocabularies are used to define terms (classes and properties) that are used to describe a particular domain. Vocabularies can be very simple, describing only a few classes having few relations among them, or very complex, containing thousands of classes having many relations among them. An ontology consists of a set of axioms which place constraints on sets of individuals and the types of relationships between them [42]. Ontologies are used to classify the terms that can be used in a particular application, characterize possible relationships, and define possible constraints on using those terms [19]. The terms vocabulary or ontology are used with no clear division. The term ontology is used to refer to a formal collection of terms, while vocabulary is used when no strict formalization is required. Ontologies are called pillars of Semantic Web and are crucial in the data integration activities as ambiguities may exist on the terms used in the different datasets, or when a bit of extra knowledge may lead to the discovery of new relationships. All terms in an ontology should have an unambiguous and non-redundant definition. The relationship among these terms are often expressed as a hierarchy. At the root of the hierarchy are expressed the most general purpose classes while all the other classes are connected between each other using the *subtype* property. Property terms describe different attributes for classes. They can also be used to associate different classes together.

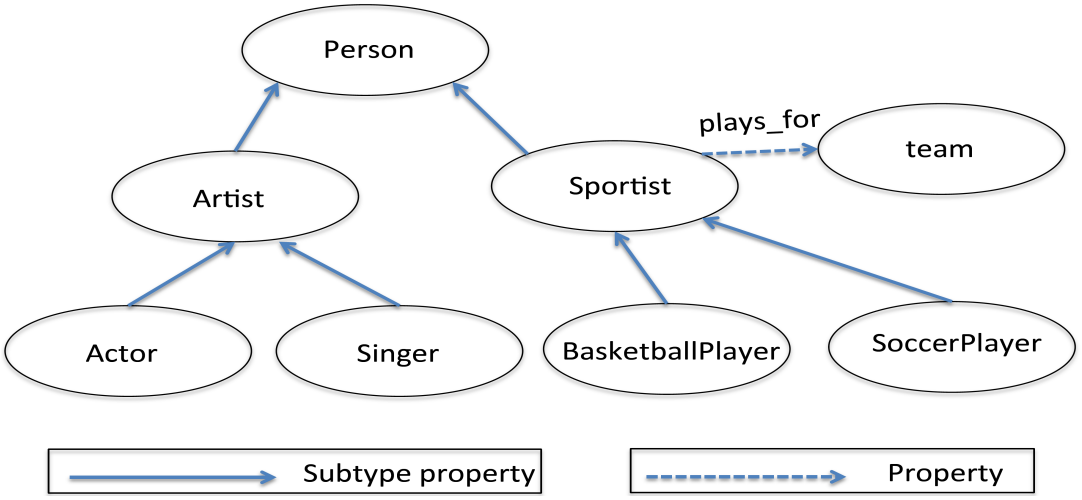


FIGURE 2.7: Ontology example

In Fig. 2.7 is given an ontology example. The class or concept *Person* is the root of the hierarchy with other classes such as *Artist* and *Sportist* being its subtypes. The class *Actor*, and *Singer* are the subtypes of *Artist*, while *Sportist* is the supertype

of *BasketballPlayer* and *SoccerPlayer*. As from the figure, the classes *Sportist* and *Team* are connected with other properties rather than subtype property, which in this example is *plays_for*. Properties determine the constraints that specify the relationship between the subject and the object using a formal language. In the following sections we will introduce two languages for building ontologies: RDFS and OWL.

2.3.1 RDF Schema

RDF Schema (RDFS) provides a data-modelling vocabulary for RDF data [34]. RDF Schema provides description of groups of related resources and the relationships between these resources. It describes properties in terms of the classes of resource to which they apply by using domain and range restrictions. In this thesis we will use `rdf:` as the namespace for <`http://www.w3.org/1999/02/22-rdf-syntax-ns#`> and `rdfs:` the namespace for <`http://www.w3.org/2000/01/rdf-schema#`>. A resource belonging to a class is given by using the predicate `rdf:type`.

The basic classes within RDF Schema are:

- `rdfs:Class`. This is the class of resources that are RDF classes
- `rdfs:Property`. This is the class of RDF properties.
- `rdfs:subClassOf`. This is used to state that a class is a subclass of another, which means that all instances of the first class are instances of the second class. In a triple this predicate means that the subject is a subtype of the object. For example the class *Artist* is subclass of *Person*. All instances of class *Artist* are instances of class *Person* but not vice versa.
- `rdfs:subPropertyOf`. This is used to state that a property is subproperty of another property, which means that all the resources related by the first property are also related by the second property. In a triple this predicate means that the subject is a subproperty of the object. For example, if we have that two instances are related with each other by the property *hasSibling*, and *hasSibling* is a subproperty of *hasRelative*, from this we can deduce that the two instances are also related to each other by the *hasRelative* property.

- `rdfs:domain`. This is used to state that any resource that has a given property is an instance of one class. In a triple where the subject is a property and the object is the class, this is the domain of this property.
- `rdfs:range`. This is used to state that the values of a property are instances of one class. In a triple where the subject is a property and the object is the class, this is the range of this property.

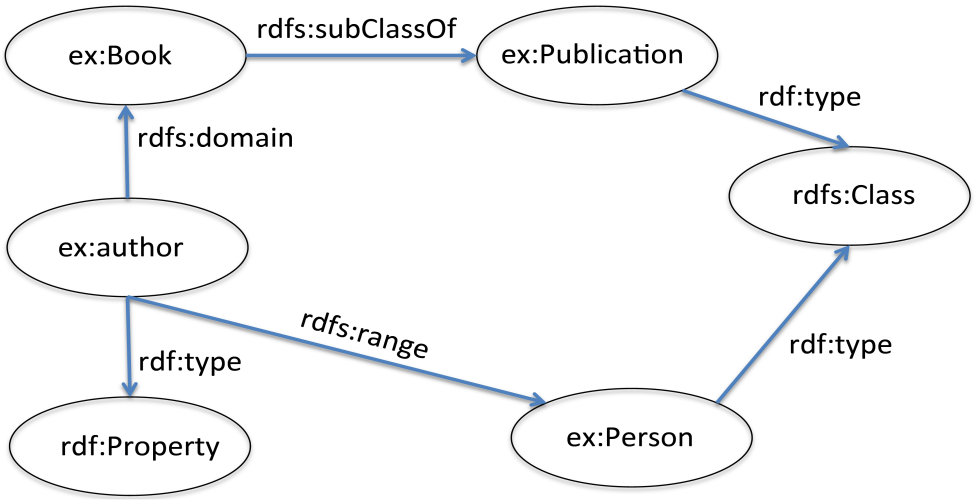


FIGURE 2.8: An RDF Schema example

In Fig. 2.8 is given an example of an RDF Schema. The class *Book* is a subclass of the class *Publication*. The property *author* is an instance of `rdf:Property`, while the two classes *Publication* and *Person* are instances of the class `rdfs:Class`. The property *author* has the domain *Book* and range *Person*.

By using RDFS vocabulary, we can infer additional information. As from the graph, suppose we have a triple describing an instance of the class *Book*, which has the property *author* and as object an instance of the class *Person*. As the class *Book* is a subclass of the class *Publication* then we can infer that the instance of our triple is also of type *Publication*.

2.3.2 Web Ontology Language

The Web Ontology Language (OWL), similarly to RDFS is used to define web ontologies [16]. OWL extends RDFS in enabling to work efficiently with queries and automatic reasoners, and it provides useful annotations for bringing the data

models into the real world. OWL is used to express more complex relationship and to describe more detailed characteristics of properties. While RDFS is used to define the structure of the data, OWL describes the semantic relationships between resources. In contrast to RDFS, OWL adds more vocabulary for describing properties or classes such as relationships between classes (e.g., `owl:disjointWith`, equality (e.g., `owl:sameAs`), richer properties (e.g., `owl:SymmetricProperty`), and class property restriction (e.g., `owl:allValuesFrom`). For instance, one can use `owl:disjointWith` to state that `Dessert` is disjoint from `Seafood` or `Meat`, and it does not assert that also `Seafood` and `Meat` are disjoint. One can use `owl:sameAs` to state that two entities belonging to two different datasets are equivalent. It also makes use of symmetric properties such as `friendOf` or `hasSpose` and inverse properties such as `motherOf` and `hasMother`. Moreover, it also describes property restrictions by value constraints and cardinality constrains. For example, an `owl:allValuesFrom` property constraints all individuals described by the property `hasParent` to have values of class `Humans`. Unlike RDFS, in OWL it is not allowed to say that something can be both an instance and a class. Also in OWL inferences are allowed.

Consider the following example in Fig. 2.8. If we know that *Pride and Prejudice* is of type *Book*, and *Book* is a subclass of *Publications*, in OWL we can infer that *Pride and Prejudice* is of type *Publications*. OWL is part of the W3C 's Semantic Web technology stack and is recognized by W3C Recommandation since 2008. The current version referred to as OWL2 is also recognized as W3C Recommendation since 2012.

OWL semantics is based in Description Logic. Description Logic (DL) is a family of logic-based knowledge representation formalisms designed to represent and reason about the knowledge of an application domain in a structured way [11]. This language provides a set of constructors to build classes and property descriptors. In this chapter we will not introduce a specific formalism to refer to schemas or ontologies as this is necessary only in chapter 5 where we will define the basic concepts of DL.

2.4 Query Language for RDF

Several query languages have been developed for the RDF data model. In this thesis we will describe the most familiar query language that is SPARQL. A comparison of different query languages for RDF data can be found in [110]. W3C Data Access Working Group developed SPARQL (SPARQL Protocol And RDF Query Language) as a standard query language and protocol in 2008 [43]. SPARQL can be used to express queries across RDF datasets. In SPARQL, queries are expressed as a set of triple patterns with each of the three elements of the triple being a variable. SPARQL queries are similar with SQL queries [48] if we consider RDF as SQL relational database, where triples are seen as placed in a table with three different columns; subject, predicate and object. Using SPARQL, users can write queries that consist of triple patterns. SPARQL provides a set of analytical queries operations such as, JOIN, SORT and AGGREGATE. A SPARQL endpoint is an HTTP server (identified by a given URL) which receives requests from SPARQL clients. SPARQL language specifies four different query variations:

- `SELECT query` This is used to extract raw data from a SPARQL endpoint.
- `CONSTRUCT query` This is used to extract information from the SPARQL endpoint and transform the results into valid RDF.
- `ASK query` This is used to provide a simple True/False result for a query on a SPARQL endpoint
- `DESCRIBE query` This is used to extract an RDF graph from the SPARQL endpoint, the content of which is left to the endpoint to decide based on what the maintainer deems as useful information.

Each of these query forms takes a WHERE block to restrict the query, although, in the case of the DESCRIBE query, the WHERE is optional.

Consider we want to know all books that have Haruki Murakami as author in *DBpedia* dataset. In Fig. 2.9 is given the query we have to execute to obtain the list of books that Haruki Murakami wrote.

The query begins with `PREFIX` statements that define the abbreviation for the namespace. In Fig. 2.9 we used three different namespaces lines from 01-03. In

```
01.  PREFIX dbo: <http://dbpedia.org/ontology/>
02.  PREFIX res: <http://dbpedia.org/resource/>
03.  PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
04.
05.  SELECT DISTINCT ?uri
06.  WHERE {
07.        ?uri rdf:type dbo:Book .
08.        ?uri dbo:author res:Haruki_Murakami .
09.  }
```

FIGURE 2.9: An SPARQL query example

line 04, the query begins with a `SELECT`, which contains a variable starting with a "?". Sometimes in SPARQL, variables can also start with a "$". In this example the variable is given by the name `?uri`. Line 06 opens the `WHERE` clause, and in this case it contains two triple patterns. Line 07 specifies that we need those resources that are of type `Book`, while line 08 states that we are looking for those resources for which the author is Haruki Murakami.

Once we execute this query we will get the results in Fig. 2.10

2.5 Triplestore

In order to execute SPARQL queries, RDF data should be stored in triplestores. Triplestores are databases for RDF triple storage. RDF data are also indexed in triplestores which allows users to execute queries more easily and efficiently. At the time of writing this thesis the most knows triplestores are Virtuoso [53] and Sesame [35].

2.6 Linked (Open) Data

Linking data distributed across the Web requires a standard mechanism for specifying the existence and meaning of connections between items described in the data. This mechanism is provided by RDF that provides a flexible way to describe things in the world. As detailed above, RDF can be used to describe people, restaurants, locations, as well as abstract concepts. These statements of relationships

uri
http://dbpedia.org/resource/A_Wild_Sheep_Chase
http://dbpedia.org/resource/Sputnik_Sweetheart
http://dbpedia.org/resource/After_the_quake
http://dbpedia.org/resource/Blind_Willow,_Sleeping_Woman
http://dbpedia.org/resource/Hard-Boiled_Wonderland_and_the_End_of_the_World
http://dbpedia.org/resource/Kafka_on_the_Shore
http://dbpedia.org/resource/Norwegian_Wood_(novel)
http://dbpedia.org/resource/South_of_the_Border,_West_of_the_Sun
http://dbpedia.org/resource/The_Wind-Up_Bird_Chronicle
http://dbpedia.org/resource/Underground_(Murakami_book)
http://dbpedia.org/resource/Dance_Dance_Dance_(novel)
http://dbpedia.org/resource/Pinball,_1973
http://dbpedia.org/resource/Uten_Enten
http://dbpedia.org/resource/Hear_the_Wind_Sing
http://dbpedia.org/resource/What_I_Talk_About_When_I_Talk_About_Running
http://dbpedia.org/resource/After_Dark_(Murakami_novel)
http://dbpedia.org/resource/Colorless_Tsukuru_Tazaki_and_His_Years_of_Pilgrimage
http://dbpedia.org/resource/1Q84
http://dbpedia.org/resource/The_Elephant_Vanishes

FIGURE 2.10: An SPARQL query result example

between things are, in essence, links connecting things in the world. Data published and linked on the Web using RDF, are more usable and more discoverable. The term of Linked Data is referred to a set of best practices for publishing and interlinking structured data on the Web [26].

- Use URIs as names for things
- Use HTTP URIs so that people can look up those names
- Provide useful information about what a name identifies when it is looked up, using open standards such as RDF, SPARQL, etc
- Refer to other things using their HTTP URI-based names when publishing data on the Web

The first Linked Data principle imposes the use of URI references to identify not only Web documents and their content but also real world object.

On the Web the protocol which allows accessing documents is HTTP. The second Linked Data principle imposes the use of HTTP URIs to identify objects, in order to allow them to be dereferenced over HTTP into a description of that object.

Data should be published in a standardized format in order to allow Web applications to process their content. The third principle of Linked Data imposes the use of RDF format and data should be available for SPARQL queries.

The fourth Linked Data principle imposes the use of hyperlinks to connect any type of thing. In the classic Web, hyperlinks are used to link resources but usually they are untyped links, while in Linked Data, links have types which describe the relation among things. These links are called RDF links. Given such connections, one can easily follow links to discover further interesting information for the entities under consideration. Note that this implies that data should not be isolated anymore but rather connected.

Open Data, are the data that can be freely used, reused and redistributed to everyone. Linked Open Data (LOD) project started publishing data adopting Linked Data principles under an open license. Tim Berners-Lee proposed a five-star rating system[2] which allows users to evaluate their dataset by using this rating system.

- ($\star$) Available on the web (whatever format) but with an open licence, to be Open Data
- ($\star\ \star$) Available as machine-readable structured data (e.g. EXCEL instead of image scan of a table)
- ($\star\ \star\ \star$) data is available as machine-readable but in a non-proprietary format (e.g., CSV rather than EXCEL)
- ($\star\ \star\ \star\ \star$) data is available according to all the above, plus the use of open standards from the W3C (RDF and SPARQL) to identify things, so that people can link to it.
- ($\star\ \star\ \star\ \star\ \star$) data is available according to all the above, plus outgoing links to other peoples data to provide context.

[2]https://www.w3.org/DesignIssues/LinkedData.html

This rating scale promotes the potential reusability of Linked Data.

Even though Fig. 1.1 refers to Linked Data datasets that are published under an open license (LOD), the techniques developed during this work can be applied to all Linked Data datasets, regardless of the fact that there are open or not. For this reason, throughout this thesis we refer to the Linked Open Data (LOD) cloud as Linked Data cloud (LD).

Chapter 3

State of the art

This chapter gives an overview of state-of-the art approaches, techniques and tools in profiling Linked Data. At the beginning of this chapter we introduce the importance of profiling Linked Data, as well as describing the main challenges and a brief summary of profiling relational vs. non-relational data in section 3.1. Next, in section 3.2 we provide a survey of general purpose profiling tools available at the moment of writing this thesis. As three aspects of profiling; topic-based profiling, schema-based profiling and linkage-based profiling are the focus of this thesis, for each of them in the following sections we provide a more detailed overview of the existing approaches and provide a summary of the open issues.

The main focus of this thesis is how to perform profiling in order to help users understand and explore the data. The goal of profiling Linked Data is serendipitous re-use; while looking for the data we need, we can find easily the data that fulfill our request. We could make a quick decision if the data is useful for our use case at hand or not if we could know some characteristics of the data.

Challenges in exploring Linked Data are numerous: use of similar classes, incomplete ontological information, error links at instance level, makes it hard for users to understand the overall content of a dataset, as well as understand and find the particular part of the data that might be of interest. Such challenges can often eclipse the benefits of interacting over Linked Data. Now that a huge amount of data is available in the Linked Data cloud, providing techniques for effective Linked Data searching, exploration, and visualization is becoming crucial [64].

Many others have tackled the problems related to profiling Linked Data as we will see in the following of this chapter.

Before introducing the most relevant approaches for each profiling aspect considered in this thesis, we give a brief overview of differences and challenges between profiling relational vs. non-relational data.

3.1 Profiling Relational vs. Non-Relational Data

Many existing profiling methods used to profile relational data, can not be applied in Linked Data, because these methods neither do scale well, nor can handle this kind of data [98]. On the other hand, the relation data-schema in Linked Data is not as strong as in the relational data. In relational data, schemas pre-exist to data and control methods implemented by database management systems enforce only data that comply to the schema. In contrast, Linked Data schemas are not required to pre-exist to data and the enforcement of the compliance of data to the schema is weaker. In this case data are associated to a schema by means of annotations. Moreover, in relational data any statement that is not known to be true is false, while in Linked Data any statement that is not known can not be asserted as neither true nor false [14]. Even though the schema of Linked Data is defined by vocabularies there is no mechanism to enforce data to be compliant to the schema. Vocabularies such as RDFS are not expressive enough to detect inconsistencies, and most important you can use the information in the schema to deduce new knowledge. In Linked Data, a relation between two instances can hold even if the schema does not model such relation between the classes that the instances belong to. Beside these problems, often Linked Data have weakly specified ontologies. As described in section 2.3.2 in RDFS vocabulary classes disjointness and cardinality restrictions can not be modeled. Due to the above reasons, a schema-based profiling for Linked Data is very important. Another challenge is due to the flexibility of Web linking (anybody can link to anything else) and the dynamic nature of the web environment thus requiring also a provenance profiling. Linked Data does not have a well-defined schema and the ontological heterogeneity represents an obstacle, thus it is also necessary to profile its schema before profiling the data itself. Moreover, Linked Data come in different formats, different encoding schemes, they use similar classes, or different vocabularies for the same class, etc. Thus data consumers have to deal with the challenge of

putting much effort in adapting profiling techniques for relational data to the non-relational data, or otherwise to develop new techniques and algorithms that work for non-relational data. The survey [1] summarizes the profiling techniques and tools in the field of relational data. As describing profiling techniques for relational data is out of the scope of this thesis we invite the reader to refer to the survey in [1] for more details.

3.2 Models and Tools for Data Profiling

In this section, we provide an overview of the existing models and data profiling tools used for different profiling tasks. As mentioned in chapters 1 and 2 ontologies and vocabularies are used to describe the semantics of the data found in a dataset, thus can be considered as profiles. These profiles are stored in data catalogues. DCAT[1] is an RDF vocabulary designed with the aim of helping data publishers increase discoverability and enable applications to consume metadata from multiple catalogs easily. When DCAT is applied, the catalogue records (but not the data files themselves) are prepared for publication as Linked Open Data and thus are stored in a machine-readable format. DCAT is a W3C recommendation since 2013. Up till now the creation of data catalogues is done manually, thus being prone to errors.

With the many efforts done by the Semantic Web community, there exist some tools which automatically extract descriptive information and statistics about the data. Statistics and summaries can help to describe and understand large RDF data. As it will be discussed, most of the existing techniques to profile Linked Data are limited to few statistics thus being able to cover only one profiling task.

Roomba[2] [7] is a framework to automatically validate and generate descriptive dataset profiles. The extracted metadata are grouped into four categories (general, access, ownership or provenance) depending on the information they hold. After metadata extraction some validation and enrichment steps are performed. Metadata validation process identifies missing information and automatically corrects them when it is possible. As an outcome of the validation process, a report is produced which can be automatically sent to the dataset maintainer.

[1] https://www.w3.org/TR/vocab-dcat/
[2] https://github.com/ahmadassaf/opendata-checker

The ExpLOD [74] tool is used to summarize a dataset based on a mechanism that combines text labels and bisimulation contractions. It considers four RDF usages that describe interactions between data and metadata, such as class and predicate instantiating, class and predicate usage on which it creates RDF graphs. It also provides statistics about the number of equivalent entities connected using the `owl:sameAs` predicate to describe the interlinking between datasets. The ExpLOD summaries are extracted using SPARQL queries or algorithms such as partition refinement.

RDFStats[3] [82] generates statistics for datasets behind SPARQL endpoint and RDF documents. It is built on Jena Semantic Framework and can be executed as a stand-alone process, important to optimize SPARQL queries. These statistics include the number of anonymous subjects and different types of histograms; URI-Histogram for URI subject and histograms for each property and the associated range(s). It also uses methods to fetch the total number of instances for a given class, or a set of classes and methods to obtain the URIs of instances.

LODStats [8] is a profiling tool which can be used to obtain 32 different statistical criteria for datasets from Data Hub[4]. These statistics describe the dataset and its schema and include statistics about number of triples, triples with blank nodes, labeled subjects, number of `owl:sameAs` links, class and property usage, class hierarchy depth, cardinalities etc. These statistics are then represented using Vocabulary of Interlinked Datasets (VOID) and Data Cube Vocabulary[5].

LODOP[6] [59] is a framework for executing, optimizing and benchmarking profiling tasks in Linked Data. 56 profiling tasks are implemented as an Apache Pig script and are available online[7]. These tasks include: number of triples, average number of triples per resources/ per object URI, number of properties, average number of property values, inverse properties, etc. This tool performs profiling tasks in a runtime of up to hours and implements three techniques to improve the performance optimizing the logical plans of Apache Pig scripts.

ProLOD [32] is a web based tool which analyzes the object values of RDF triples and generates statistics upon them such as datatype and patterns distribution.

[3] https://sourceforge.net/projects/rdfstats/
[4] https://datahub.io/
[5] http://www.w3.org/TR/vocab-data-cube/
[6] https://github.com/bforchhammer/lodop
[7] https://github.com/bforchhammer/lodop

In ProLOD the type detection is performed using regular expression rules and normalized patterns are used to visualize huge numbers of different patterns. ProLOD also generates statistics on literal values and external links. ProLOD++[8] which is an extension of ProLOD is also a browser based tool which implements several algorithms with the aim to compute different profiling, mining or cleansing tasks. In the profiling task, processes are included to find frequencies and distribution of distinct subjects, predicates and objects, range of the predicates etc. ProLOD++ can also identify predicates combinations that contain only unique values as key candidates to distinctly identify entities. The implementation of mining tasks cover processes such as synonym and inverse predicate discovering, association rules on subjects, predicates and objects, etc. It also performs some cleansing tasks such as auto completions of new facts for a given dataset, ontology alignment in identifying predicates which are synonym or identifying cases where the pattern usage is over specified or underspecified.

Loupe [92] is a framework used to summarize and inspect Linked Datasets. Loupe extracts types, properties and namespaces, along with a rich set of statistics. It offers a triple inspection functionality, which provides information about *triple patterns* that appear in the data set and their frequency. Triple patterns have the form <*subjectType*, *property*, *objectType*>.

Aether[9] [91] is a web application used to generate, view and compare extended VOID statistical descriptions of RDF datasets. Users can use Aether to make sense of the dataset content and can help to identify changes between different versions of a dataset, as well as, detecting outliers and errors in the dataset. Statistics produced by Aether include entities, triples and statistics that explicitly relate triples and entities. It takes as input a SPARQL endpoint and can generate an extended VOID description containing a wide variety of statistical spreads describing the dataset.

The Semantic Sitemap[10] approach [47] generates an XML file that informs search engine crawlers about URLs on a website. The Sitemap protocol extends these documents with information about the location of RDF data and about alternative means to access this data e.g. data dumps and SPARQL endpoints. Using

[8] https://www.hpi.uni-potsdam.de/naumann/sites/prolod++/#/graphstatistics/dailymed
[9] http://demo.seco.tkk.fi/aether/#/
[10] http://sw.deri.org/2007/07/sitemapextension/SemanticSitemap-1.2.zip

TABLE 3.1: Data Profiling Tools.

	Roomba	Loupe	ProLOD++	ExpLOD	LODStats
Availability	code in Github	✓	✓	-	✓
Automation	✓	✓	-	-	semi-automated
Scalability	✓	-	-	-	✓
Licensing	-	-	-	-	
Usability	4	4	2	1	3
Maintenance	2016	2016	2016	2010	2012

Semantic Sitemaps, we can obtain individual resource descriptions from a large dump.

Profiling as the activity of providing insights about the data, is not only about providing statistics about value distribution, null values, etc., but also is referred as the process of finding and extracting information patterns in the data.

In the area of schema summarization, Knowledge Patterns (KP) can be defined as a template to organize meaningful knowledge [60]. The approach in [113] identifies an abstraction named dataset knowledge architecture that highlights how a dataset is organized and which are the core knowledge patterns we can retrieve from that dataset. These KPs summarize the key features of one or more datasets, revealing a piece of knowledge in a certain domain of interest.

Encyclopedic Knowledge Patterns (EKP) [104] are knowledge patterns introduced to extract core knowledge for entities of a certain type from *Wikipedia* page links. EKPs are extracted from the most representative classes describing an entity and containing abstraction of properties. The use of EKPs that supports exploratory search is shown in Aemoo[11] to enrich query results with relevant knowledge coming from different data sources in the Web [105].

In order to understand complex datasets, [29] introduces Statistical Knowledge Pattern (SKP) to summarize key information about an ontology class considering synonymity between two properties of a given class. An SKP is stored as an OWL ontology and contains information about axioms derived or not expressed in a reference ontology but can be promoted applying some statistical measures.

In the following we compare nine tools used for Linked Data Profiling using six different criteria, as in Table 3.1 and 3.2.

[11] http://wit.istc.cnr.it/aemoo/

TABLE 3.2: Data Profiling Tools

	LODOP	RDFStats	Aether	Semantic Sitemap
Availability	code in Github	code in Github	✓	✓
Automation	✓	semi-automated	-	-
Scalability	✓	-	-	✓
Licensing	-	Apache Licensing	MIT License	-
Usability	1	1	3	1
Maintenance	2014	2009	2014	2008

Availability is the degree to which a system or a product is available online as a demo, as a tool, only the code available, as a screencast or not available at all. From the nine tools available at the moment of writing this thesis, only Loupe, LODstats and Aether are available as demo or web application which a user can use online. The code in Github is available for Roomba, LODOP and RDFSstats while ProLOD++ is available as a screecast. For ExpLOD we did not find any available information for the availability while for Semantic Sitemap only the code file is available.

Automation is the degree to which a machine and technology make processes run on their own without manpower. For this criterion we used two metrics; tools that are fully automated and semi-automated. Only three of the profiling tools, Roomba, Loupe and LODOP perform the profiling automatically, while LODStats and RDFStats are semi-automatic. There is no automatization information for ProLOD++, ExpLOD, Aether and Semantic Sitemap.

Scalability is the degree to which a system, model or function can cope and perform under an increased or expanding workload. For this criterion we used two metrics; tools that are scalable and perform well with respect to the size of the dataset and tools that are not scalable (they are not projected to handle datasets of a big size). Roomba, LODStats and LODOP consider scalability issues, dealing with huge amount of data, while for the other tools there is no information about scalability.

Licensing is the degree to which a system provides formal permission from a governmental or other constituted authority to do something, to use, make, or modify the original and the restrictions with which they can be redistributed. Different licensing information come with the application or the tool, such as: All rights reserved, MIT License, Apache Licensing, Apache Version 2.0, etc. Only

two out of nine tools provide licensing information, RDFStats and Aether, which have respectively Apache Licensing and MIT License.

Usability is the degree to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use. The two metrics for usability used to evaluate profiling tools are understandability and learnability. As for the understandabilty the purpose of the system should be easily understandable while learnability is referred to the completeness of user documentation and explanation how to achieve common tasks. This is directly connected with user satisfaction on using this tool. To evaluate usability we used the Likert Scale [3], from 1 to 5, where 1 is Very Dissatisfied and 5 means Very Satisfied. Based on the documents and manuals how to install and use profiling tools, we assigned a value to measure their usability. Loupe and Roomba were the only tools that provide full documentation and tutorial on how to install and use them. LODStats and Aether provide some documentation but they are not complete, while ProLOD++ provide an insuficient documentation. We are completely missing documentation for ExpLOD, LODOP, RDFStats, and Semantic Sitemap.

Maintenance is related to the latest update of the tool. Roomba, Loupe and ProLOD++ are tools which were recently released and updated, while other tools such as Semantic Sitemap, RDFStats, Aether and ExpLOD are outdated and not maintained anymore.

Remarks for models and data profiling tools

- Most of the existing profiling tools are limited in reporting some basic statistics like the number of classes, properties, etc.
- Some of the existing tools do not provide any information about the availability and only a few of them are completely automatic.
- There exist some tools that do not provide any documented information in order to understand and be able to implement it if it is of our interest.
- Most of the tools are outdated and are not maintained anymore.

3.3 Topic-Based Profiling

Topical profiling has been studied in data mining [111], database [87], and information retrieval communities [73]. These methods find application in domains such as documents classification, contextual search, content management and review analysis [2, 97, 13, 125, 127]. Although topical profiling has been studied in other settings before, only a few methods exist for profiling LD datasets. These methods can be categorized based on the general learning approach that is employed into the categories *unsupervised* and *supervised*, where the first category does not rely on labeled input data, the latter is only applicable for labeled data.

The authors in [50] define the profile of datasets using semantic and statistical characteristics. They use statistics about vocabulary, property, and datatype usage, as well as statistics on property values, such as average strings length, for characterizing the topic of the datasets. For classification, they propose a feature/characteristic generation process, starting from the top discovered types of a dataset and generating property/value pairs. In order to integrate the property/-value pairs they consider the problem of vocabulary heterogeneity of the datasets by defining correspondences between terms in different vocabularies. This can be done by using ontology matching techniques. The authors intended to align only popular vocabularies. They have pointed out that it is essential to automate the feature generations and proposed the framework to do so, but do not evaluate their approach on real-world datasets. Also, considering only the most popular vocabularies, makes this framework not applicable to any dataset or belonging any kind of domain.

The authors in [37] propose the application of aggregation techniques to identify clusters of semantically related Linked Data given a target. Aggregation and abstraction techniques are applied to transform a basic flat view of Linked Data into a high-level thematic view of the same data. Linked Data aggregation is performed in two main steps; similarity evaluation and thematic clustering. This mechanism is the backbone of inCloud framework [38]. As an input, the system takes a keyword-based specification of a topic of interest, namely a real-world object/person, an event, a situation, or any similar subject that can be of interest for the user and returns a part of the graph related to the keyword in input. The authors claim that they evaluated the inCloud system by measuring user

satisfaction and system evaluation in terms of accuracy and scalability but do not provide any experimental data.

An approach to detect latent topics in entity-relationship graphs is introduced by [31]. This approach works in two phases: (1) A number of subgraphs having strong relations between classes are discovered from the whole graph, and (2) the subgraphs are combined to generate a larger subgraph, called summary, which is assumed to represent a latent topic. Topics are extracted from vertices and edges for elaborating the summary. This approach is evaluated using the *DBpedia* dataset. Their approach explicitly omits any kind of features based on textual representations and solely relies on the exploitation of the underlying graph. Thus, for datasets that do not have a rich graph, but with instances that are described with many literal values, this approach cannot be applied.

In [56], the authors propose an approach for creating dataset profiles represented by a weighted dataset-topic graph which is generated using the category graph and instances from *DBpedia*. In order to create such profiles, a processing pipeline that combines tailored techniques for dataset sampling, topic extraction from reference datasets, and relevance ranking is used. Topics are extracted using named-entity-recognition techniques, where the ranking of the topics is based on their normalized relevance score for a dataset. These profiles are represented in RDF using VOID vocabulary and Vocabulary of Links[12]. The accuracy for the dataset profiles is measured using normalized discounted cumulative gain which compares the ranking of the topics with the ideal ranking indicated by the ground truth.

Automatic identification of topic domains of the datasets utilizing the hierarchy within the *Freebase* dataset is presented in [81]. This hierarchy provides background knowledge and vocabulary for the topic labels. This approach is based on assigning *Freebase* types and domains to the instances in an input LD dataset. The main challenge in this approach is that it fails to identify the prominent topic domains if in *Freebase* there are no instances that match entities in the dataset.

Some approaches propose to model the documents as a mixture of topics, where each topic is treated as a probability distribution over words such as Latent Dirichlet Allocation (LDA) [27], Pachinko Allocation [86] or Probabilistic Latent Semantic Analysis (pLSA) [66]. As in [119], the authors present TAPIOCA[13], a Linked

[12]http://data.linkededucation.org/vol/
[13]http://aksw.org/Projects/Tapioca.html

Data search engine for determining the topical similarity between datasets. TAPIOCA takes as input the description of a dataset and searches for datasets with similar topic which are assumed to be good candidate for linking. Latent Dirichlet Allocation (LDA) is used to identify the topic or topics of RDF datasets. For the probabilistic topic-modelling based approach two types of information are used; instances and the structure of RDF datasets. The metadata comprises classes and properties used in the dataset, removing the classes and properties of most known vocabularies such as RDF, RDFS, OWL, SKOS and VOID because they do not provide any information about the topic. By extracting this structural metadata from a dataset, TAPIOCA transforms it into a description of the topical content of the dataset. In this work, the authors build a gold standard and make it available, but it is difficult to use it as the information is encoded. As described by the authors, the challenge is to search for a good number of topics and how to handle classes and properties in other languages rather than English. Thus, picking a good number of topics has a high influence on the models performance. Moreover, approaches that use LDA are very challenging to adapt in cases when a dataset has many topics. These approaches are very hard to be applied in LD datasets because of the lack of the description in natural language of the content of the dataset as will be described in chapter 4.

Remarks for topic-based profiling

- There is no existence of a gold standard for the multi-topic classification of LD datasets so the developed approaches can be comparable.
- Some approaches do not provide any experimental data, thus making it challenging to replicate the results.
- Many approaches for the topic classification are using only classes and properties in English language thus limiting their application for datasets with other language rather than English.
- Topics can be represented by relevant classes where the relevant classes are mined by considering mappings between classes of the most known vocabularies thus making this approach inapplicable for datasets using other vocabularies rather than the most known.
- Some approaches consider only textual representation thus making this inapplicable for datasets that do not have a graph rich of labels.

- One approach uses categories in the *Freebase* dataset as topic, by assigning its types and domains to the instances of the dataset for which we want to assign the topic, failing in the case where there are no instances that match entities in the dataset.
- Using probability distribution leads to the challenge of finding the good number of topics as this influences directly to the performance of the approach.

3.4 Schema-Based Profiling

Different approaches have been proposed for schema and data summarization. Most of them identify pieces of knowledge that are more relevant to the user, while the others do not represent the relations among instances but are limited in presenting the co-occurence of the most frequent types and properties. Here we will describe approaches that are explicitly proposed to summarize Linked Data and ontologies and to extract statistics about the data.

A first body of work has focused on summarization models aimed at identifying subsets of datasets or ontologies that are considered to be more relevant. The authors in [149] rank the axioms of an ontology based on their relevance to present to the user a view about the ontology. The evaluation of such approach is done using three small ontologies, containing few classes and properties. Rather, it is not known how this approach will perform for large ontologies such as *DBpedia*. Also, the authors do not have the intention to support further exploration of the ontology once user gets interested. RDF Digest [135] identifies the most relevant subset of a knowledge base (KB) including the distribution of instances in order to efficiently create summaries. This approach is tested only on RDFS ontologies and only on small ones. Both approaches are evaluated comparing the results of the summary with a gold standard summary, and in none of them the summary is evaluated in an application scenario.

As described in section 3.2, Loupe [92] is a tool used to inspect Linked Data by providing an overview of datasets content by means of patterns. Patterns describe relations between types and are used to construct summaries. For a big dataset the summary extracted by Loupe is also big as all types or relations are included. Moreover as we will discuss in chapter 5 it is not easy to navigate and explore such summaries.

In [61], the authors propose a graph-based approach called ELIS for visualizing and exploring induced schema for Linked Open Data. ELIS extracts patterns, called *schema-level patterns* as a combination between a set of subject types and a set of object types. These subject sets and object sets can be seen as nodes connected by a property. Using ELIS users are capable to visualize the induced schema. No evaluation of the induced schema is provided and there is no comparison between other induced schema approaches. Similarly, the approach in [141] induces a schema from data and their axioms represent stronger patterns to mine stronger constraints.

The approach called HEIDS is proposed in [40]. This approach summarizes dataset content and generates a hierarchical grouping of entities connected by relations. For the summarization, HEIDS considers coverage of dataset, height of hierarchy, cohesion within groups, overlap between groups, and homogeneity of groups. Two datasets are used to evaluate this approach and the authors measured empirically the quality of the provided summary compared to a baseline method and report the run time. For datasets whose instances do not share many property-value pairs it is difficult to create hierarchical summaries, resulting in a flat and big summary.

In [36], the authors consider vocabulary usage in the summarization process of an RDF graph and use information similar to knowledge patterns. A similar approach is also used in MashQL [71], a system proposed to query graph-based data (e.g., RDF) without prior knowledge about the structure of a data set. Knowledge pattern extraction from RDF data is also discussed in [113], but in the context of domain specific experiments and not with the purpose of defining a general Linked Data summarization framework.

SchemeEx extracts interesting theoretic measures for large datasets, by considering the co-occurrence of types and properties [77]. A data analysis approach on RDF data based on a warehouse-style analytic is proposed in [45]. This approach focuses on the efficiency of processing analytical queries which poses additional challenges due to their special characteristics such as complexity, evaluated on typically very large data sets, and long runtime. However, this approach requires the design of a data warehouse specially for a graph-structured RDF data. Linked Open Vocabularies, RDFStats [83] and LODStats [8] provide several statistics about the usage of vocabularies, types and properties but they do not represent the connections between types.

Remarks for schema-based profiling

- Some approaches for schema-based profiling are tested on ontologies or dataset that contain only few classes, properties or instances.
- Usually approaches that provide a summary of the content of the dataset include only the most relevant features thus giving a non complete view of the dataset.
- In none of the proposed approach the summary is evaluated in a user scenario application.
- The consideration of all relation between types, or generating summaries for datasets in which instances do not share many property/ value pairs result in including in the summary redundant information.
- There is a need in summarizing datasets considering the information in the dataset and its ontology as considering only one of those is not enough.
- Some summarization approaches work on specific domains thus can not be applied to all datasets.
- There exist some tools that require a dataset SPARQL endpoint to be able to extract the summary but not all datasets in the LD cloud provide one. Moreover, even for the datasets that have a SPARQL endpoint these approaches pose the challenge of efficiency.

3.5 Linkage-Based Profiling

As we mentioned in chapter 1, we focus on data linking at instance level and thus in the related work we present only those tools or techniques. The surveys [102] and [63] summarize the effort done by the community in the field of instance matching (the process of finding equivalent instances among different datasets). Similarity is usually performed on string bases. Often semi-automated approaches, which must be pre configured by the user may select from a wide range of similarity functions those suitable for the task at hand such as Silk [142]. The Silk system assumes a supervised matching scenario where the user specifies entities to link in a configuration file and selects an aggregation approach (weighted average,

max(min), Euclidean distance, or weighted product) for her task. Similar to Silk, the LIMES system [103] is a semi-automated approach that needs a configuration file to be set up.

The LiQuate framework [120] combines Bayesian Networks and rule-based systems to analyze the quality of data and links in the LD cloud. The Bayesian Networks model dependencies among resources, while queries among these models, represent the probability that different resources have redundant labels or that a link between two resources is missing while a probabilistic rule-based system is used to infer new links that associate equivalent resources. The LiQuate framework can be used to suggest ambiguities or possible incompleteness in the data or links and to resolve the ambiguities and incompleteness identified during the exploration of the Bayesian Network. The LiQuate framework deals with two incompleteness problems; link incompleteness and ambiguities between labels of resources and between sets of links. This is a semi-automated approach for which the last update was in 2013.

LINDA [30] is a system used to compute the similarity between two entities based on their neighbors. Two kinds of similarities are computed; apriori similarity and contextual similarity. Apriori similarity is based on literals and constraints and contextual similarity is computed on each iteration and considers the current state of similarity matrix. LINDA assumes each dataset to be already disambiguated addressing a narrow application.

A statistical and qualitative analysis of instance level equivalence in the LD cloud to automatically compute alignments at the classes level could be found in [46]. Adopting classical Jaccard methods to the ontology alignment task allow to improve the level of integration between datasets as this will help to resolve semantic heterogeneity. The authors used the Jaccard coefficient to measure the similarity between two classes when interpreted as sets of instances. They considered *DBpedia* as the source dataset and 6 target datasets, and extracted the sameAs links (RDF links that connect equivalent entities). Also, the authors extracted the classes' hierarchy where the behavior of classical Jaccard similarity measure was analyzed by studying the influence of hierarchical information in producing the alignments.

The authors in [106], introduced an approach to automatically detect redundant identifiers solely by matching the URIs of information resources. They used two

techniques to match URIs. The first is to tokenize the URI in all special characters and calculate the cosine similarity of all TF-IDF vectors [72] and the second technique is to use exact string matching techniques after dividing the URI into prefix, infix and suffix to detect duplicates. Their approach is limited only for string similarity and do not cover cases when the URI contains numerical information and blank nodes.

PARIS [129] is a probabilistic approach for the automatic alignment of ontologies. It aligns instances, relations and classes by measuring degrees of matching based on probability estimates. The PARIS system characterizes a standardized score between pairs of instances that represent how likely they are to be matched, and that relies on the matching scores of their compatible neighbors. The final scores are obtained by first initializing (and fixing) the scores on pairs of literals, and then propagating the updates through the relationship graph using a fixed point iteration. PARIS does not deal with structural heterogeneity and it assumes ontologies to be the same level of granularity. As demonstrated by [80], the implementation of PARIS is not easy.

SiGMa [80] is a framework used to automatically identifying corresponding entities among datasets and interlinking them. It works in two stages: first starts with a small seed matching assumed to be of good quality and after the algorithm incrementally augments the matching. The matching is done by using both structural information and properties of entities such as their string representation to define a modular score function. The experiments results are satisfactory and can be applied in large scale. As in the first step it assumes a small seed matching assumed to be of a good quality it is prone for errors and it does not allow correcting previous mistakes. It also needs parameter tuning for the learning to rank model and can not handle alignments other than 1-1.

Remarks for linkage-based profiling

- Most of the approaches in linkage-based profiling are semi-automatic needing the user intervention for setting up a configuration file
- Usually linkage frameworks use the classical similarity metrics that are applied on literal values but also structural information should be taken into consideration

- For those approaches that consider also structural heterogeneity, they assume that ontologies are of the same level of granularity while in practice ontologies can be of different granularity levels
- Frameworks that adopts ranking functions are difficult to implement and they need parameter tuning
- Some approaches consider the information of the local name in the URI of a resource for the similarity measures, but these approaches are limited and can be applied in only decoded string URIs

3.6 Summary

In this chapter we discussed the state-of-the-art approaches and tools in profiling Linked Data. As reflected in this chapter although there are huge achievements by the approaches and tools developed by the community, there are still some open issues. Most of the existing profiling tools generate basic statistics about a dataset thus covering only a specific task. The three aspects of profiling considered in this thesis have not been studied in a systematic way, thus there exist many open issues.

Because of the importance of the solutions for each open issue, profiling Linked Data is a hot topic. Although in this thesis we can not address all the above issues we resolve some of them as it will be described in the following chapters.

Chapter 4

Topic-Based Profiling

In this chapter we address the problem of automatic topic profiling to support users in classifying their data into one of the topical categories of the LD cloud by training machine learning classifiers with different features vectors. We provide an approach for classifying datasets at two levels of granularity; single-topic and multi-topic classification.

The chapter is structured as follows. Section 4.2 discusses the data model used to train our machine learning classifiers. We also provide a description for each topic and introduce some examples for datasets that belonge to each of them. In section 4.3 we introduce the approach; describing first the features we consider as input for our purpose. Then, we introduce the reader to three classification algorithms,kNN, Naive Bayes and J48 that we considered for our experiments. Moreover, because of the imbalance of the datasets belonging to each topical category we also describe different techniques for sampling. We considered two normalization technique for the features used by each dataset with the aim of reducing the influence of features that occur more often in the dataset. Results of the single-topic classification are provided in section 4.4.1 while the results of the experiments for the multi-topic classification are provided in section 4.4.2. We conclude this chapter with the discussion of the results achieved by our experiments in sections 4.4.1.3 and 4.4.2.3 .

4.1 Overview

The Web nowadays offers millions of datasets containing information on almost every area that might be interesting to someone. Often a user requires retrieving very fast all the available and useful information about a target, such as people, events, places, situations, etc. This information is important in order to take decisions, organize business work and plan future actions. The process of exploring Linked Data for a given target is long and not intuitive. Especially when the dataset do not provide information about its topic/s a lot of exploration steps are required in order to understand if the information contained in the dataset is useful or not. The decision of using such dataset is done through accessing the metadata that describe its content.

Data publishers should provide metadata that describe the characteristics of the datasets, for instance their topics, and more detailed information about their content as well as statistics [65]. These metadata are represented using the VOID vocabulary. Dataset-level metadata are also represented using Semantic Sitemaps. In the state of LD 2011[1], 63,05% (186 out of 295 data sources) do not provide either a VOID description or a Semantic Sitemap. Since the first proposal of Linked Data as a data publishing standard, the Linked Data cloud has grown to more than 1 000 datasets as of April 2014 [124]. The number of datasets that provide a VOID description from 2011 to 2014 has decreased from 32.20% to 14.69%. This is an important shortcoming of the current LD cloud. The datasets in the LD cloud 2014 belong to different domains, with social media, government data, and publications data being the most prominent areas [124]. For some dataset published as LD such as *Linked Movie Database*[2], or *GeoNames*, the metadata are completely missing, while for some others e.g., *DBpedia*[3], the topics it covers are not explicitly described. The covered topics might be easy to guess in cases where the dataset is well-known, but for relatively small and unknown datasets it is difficult to know its topic, e.g., what is the topic of *http://sadiframework.org/services/* dataset?

For many purposes, it is useful to have a classification of datasets according to their topical domain. Agents navigating through the Web of Linked Data should know the topic of a dataset discovered by following links in order to judge whether it is useful for the use case at hand or not. Furthermore, as shown in [124], it is often

[1] http://lod-cloud.net/state/#data-set-level-metadata
[2] http://www.linkedmdb.org/
[3] http://www.dbpedia.org

interesting to analyze characteristics of datasets clustered by topical domains, so that trends and best practices that exist only in a particular topical domain can be identified. Link discovery also can be supported by knowing the topic of the dataset. Usually datasets that share the same topic, probably share equivalent instances. Topical classification is also important for categorizing of the Linked Data cloud as in Fig. 1.1, which marks datasets according to their topical domain. Up till now, topical categories were manually assigned to LD datasets either by the publishers of the datasets themselves via the `datahub.io` dataset catalog or by the authors of the LD cloud.

The *topic* of a dataset can be understood as the dataset's subject, i.e. the subject or theme of a discourse or of one of its parts. As the LD cloud was manually created, for every dataset in the cloud the topic was assigned by either verifying its content or by accessing the metadata assigned by the publisher. Recalling the user scenario in section 1.1, the first criteria for our app developer to select the right datasets is to know their topics. Because the topic of the datasets was manually assigned, she needs an automatic approach to assign the topic to other datasets she finds on the Web. To support this problem, in this chapter, we investigate to which extent can we automatically classify datasets into the topical categories used within the LD cloud. We use the most recent LD cloud data collection [124] to train different classifiers for determining the topic of a dataset.

4.2 Data Model

The last crawl of Linked Data was performed in April 2014 by [124]. Authors used the *LD-Spider* crawler originally designed by [69], which follows dataset interlinks to crawl LD. The crawler seeds originate from three resources:

(1) Datasets from the *lod-cloud* group in `datahub.io` datasets catalog, as well as other datasets marked with Linked Data related tags within the same catalog.

(2) A sample from the Billion Triple Challenge 2012 dataset[4].

(3) Datasets advertised on the `public-lodw3.org` mailing list since 2011.

[4] `http://km.aifb.kit.edu/projects/btc-2012/`

The crawled data contained 900 129 documents describing 8 038 396 resources with altogether 188 million RDF triples. To group all the resources in datasets, it was assumed that all the data originating by one pay-level domain (PLD) belong to a single dataset. The gathered data originates from 1024 different datasets from the Web and is publicly available[5]. Fig. 4.1 shows the distribution of the number of resources and documents per dataset contained in the crawl.

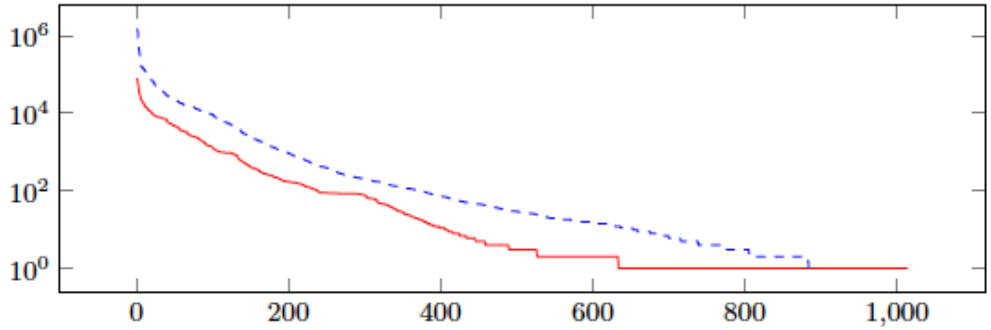


FIGURE 4.1: Distribution of the number of resources (—) and documents (- - -) (log scale) per dataset contained in the crawl

For creating the diagram, the datasets were manually annotated with one of the following topical categories: *media, government, publications, life sciences, geographic, cross-domain, user generated content,* and *social networking* [124].

Media category contains datasets providing information about films, music, TV and radio programmes, as well as printed media. Some datasets in this category are the `dbtune.org` music dataset, the *New York Times* dataset, and the *BBC radio and television program* datasets.

Government category contains Linked Data published by federal or local governments, including a lot of statistical datasets. Examples of the datasets in this category include the `data.gov.uk` and `opendatacommunities.org` dataset.

Publications category holds information library datasets, information about scientific publications and conferences, reading lists from universities, and citation database. Prominent datasets in this category include *German National Library* dataset, the *L3S DBLP dataset* and the *Open Library* dataset.

[5] `http://data.dws.informatik.uni-mannheim.de/lodcloud/2014/ISWC-RDB/`

Geographic category contains datasets like `geonames.org` and `linkedgeodata.org` comprising information about geographic entities, geopolitical divisions and points of interest.

Life science category comprises biological and biochemical information, drug-related data, and information about species and their habitats. Examples of datasets that belong to this category are *Drugbank FU-Berlin, Geospecies* and *Biomodels RDF*.

Cross-domain category includes general knowledge bases such as *DBpedia* or *UMBEL*, linguistic resources such as *WordNet* or *Lexvo*, as well as product data.

User-generated content category contains data from portals that collect content generated by larger user communities. Examples include metadata about blogposts published as Linked Data by `wordpress.com`, data about open source software projects published by `apache.org`, scientific workflows published by `myexperiment.org`, and reviews published by `goodreads.com` or `revyu.com`.

Social networking category contains people profile as well as data describing the social ties among people. In this category individual FOAF profiles are included, as well as data about the interconnections among users of the distributed microblogging platform *StatusNet*.

The authors of the LD cloud make a distinction between the categories *user-generated content* and *social networking*. Datasets in the former category focus on the actual content while datasets in the later category focus on user profiles and social ties. Fig. 4.2 shows the distribution of categories over the dataset within the LD cloud.

As we can see from Fig. 4.2, the cloud is dominated by datasets from the *social networking* category, followed by *government* datasets. Only less than 25 datasets are included in the cloud for each of the domains *media* and *geographic*. The topical category is manually assigned to each dataset in the LD cloud thus we consider as a gold standard for our experiments. The imbalance needs to be taken into account for the later model learning, as some classification algorithms tend to predict better for stronger represented classes.

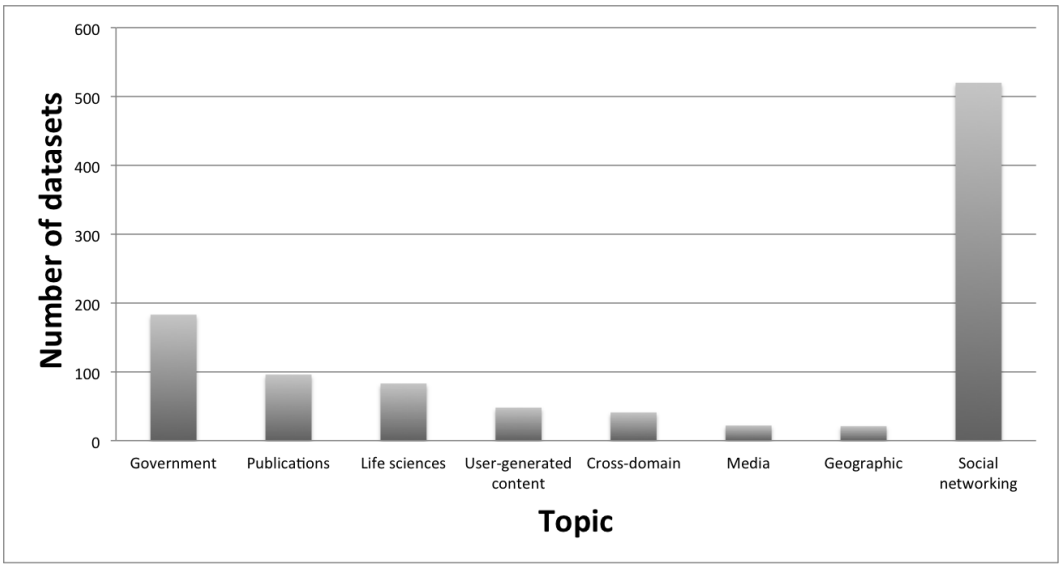


FIGURE 4.2: Topics Distribution within LD Cloud Datasets

Given a large RDF dataset with heterogeneous content, we want to derive the topic or topics that can be understood as the subject/s of the dataset by using different feature vectors that describe the characteristics of the data.

Definition 4.1. (Topical category) Given a set of RDF triples (s, p, o), a topic T is a set of labels $\{ l_1, l_2, ...l_k \}$ that describes the content of the dataset relating it with a specific area of the real world.

Definition 4.2. (Single-topic classification) Given a set $\{ D_1, D_2, ...D_N \}$ of datasets, where each D_i is associated with a feature vector $x_i = (x_{i1}, x_{i2}, ... x_{iM})$, the process of assigning only a single label l_j from the set of labels $\{ l_1, l_2, ...l_p \}$ to D_i, is called single-topic classification.

Definition 4.3. (Multi-topic classification) Given a set $\{ D_1, D_2, ...D_N \}$ of datasets, where each D_i is associated with a feature vector $x_i = (x_{i1}, x_{i2}, ... x_{iM})$, the process of assigning a subset of labels $l_k \subseteq$ L to D_i, where L = $\{ l_k : k = 1..p \}$ is the set of p possible labels, is called multi-topic classification.

In the LD cloud 2014 the datasets have only one topic, which was manually assigned. Throughout this thesis, we investigate the problem of assigning to LD datasets a single or multi topics. At the beginning we investigate to which extent can we automatically assign a single topic to each dataset. Considering the results of the first experiments about single-topic classification we investigate the problem of multi-topic classification of LD datasets.

4.3 Approach

For the topical extraction or classification of LD datasets several approaches have been proposed as in section 3.3. These approaches consider schema-level [50, 31, 81] or data-level information [37, 56] as input for the classification task. In [66] the topic extraction of RDF datasets is done through the use of schema and data level information. Similarly as in the cited works, we also investigated if schema or data-level information are good indicators for the topical assignment. We extracted different feature vector, for all datasets in the LD cloud and train different classification algorithms on top of them. In our work, we draw from the ideas of [50] of using schema-usage characteristics as features for the topical classification, but focus on LD datasets. [37] takes a keyword-based specification of a topic of interest and returns a part of the graph related to the keyword given as input through matching techniques, while in our approach we do not imply any matching algorithm, but use schema-based information to assign the topic. Differently from [31] which omits any kind of features based on textual representations and solely relies on the exploitation of the underlying graph, in our approach we extract all schema-level data. In this approach only strong relations between classes are discovered from the whole graph, while in our approach we do not consider the relation between classes but extract all classes and all properties used in the dataset. [56] extracts topics using named-entity-recognition techniques, the category graph and instances from *DBpedia*. In our approach we do not use any entity-recognition techniques but rather use schema-level information and different algorithms for the topic classification of LD datasets. The approach proposed by [66] uses LDA for the topical extraction of RDF datasets. For the probabilistic topic-modelling two types of information are used; instances and the structure of RDF datasets. This is a very challenging approach to adapt especially when the dataset belongs to many topics or the description of the dataset is in other languages rather than in English.

4.3.1 Feature Vectors

For each of the datasets contained in our collection, we created ten different feature vectors, which capture different aspects of the dataset. We made this information

available[6].

Vocabulary Usage (VOC): As vocabularies mostly describe a set of classes for a particular domain, e.g. `foaf` for describing persons, or `bibo` for bibliographic information, we assume that the vocabularies used by a dataset form a helpful indicator for determining the topical category of the dataset. We extract the predicates and classes which represent the set of terms of the vocabularies used by the dataset. We determine the vocabularies by aggregating the namespaces of these terms. We then summed up the number of occurrences resulting in a total of 1 453 vocabularies.

Class URIs (CUri): As a more fine-grained feature, the `rdfs:classes` and `owl:classes` which are used to describe entities within a dataset might provide useful information to determine the topical category of the dataset. Thus, we extracted all used classes of the datasets in the cloud and generated 914 attributes.

Property URIs (PUri): Beside the class information of an entity, another feature which will help is to have a look at the properties which are used to describe it. For example it might make a difference, if people in a dataset are annotated with `foaf:knows` statements or if her professional affiliation is provided. To leverage this information, we collected all the properties which are used within one dataset from the crawled data. This feature vector consists of 2 333 attributes.

Local Class Names (LCN): Different vocabularies might contain synonymous (or at least closely related) terms as described in section 1.2 that share the same local name and only differ in their namespaces, e.g. `foaf:Person` and `dbpedia:Person`. Creating correspondences between similar classes from different vocabularies reduces the diversity of features, but on the other side might increase the number of attributes which are used by more than one dataset. As we lack correspondences between all the vocabularies, we bypass this by using only the local names of the classes, meaning `vocab1:Country` and `vocab2:Country` are mapped to the same attribute. We used a simple regular expression to determine the local class name checking for `#`, `:` and `/` within the class URI. By focusing only on the local part of a class name, we increase the number of classes that are used by more than one dataset in

[6]https://github.com/Blespa/TopicalProfiling

comparison to CUri and thus generate 1 041 attributes for the LCN feature vector.

Local Property Names (LPN): Using the same assumption as for the LCN feature vector, we also extracted the local name of each property that is used by a dataset. This results in treating `vocab1:name` to `vocab2:name` as a single property. We used the same heuristic for the extraction as for the LCN feature vector and generated 2 073 different local property names which are used by more than one dataset, resulting in an increase of the number of attributes in comparison to the PUri feature vector.

Text from `rdfs:label` (LAB): Beside the vocabulary-level features, the names of the described entities might also indicate the topical domain of a dataset. We thus extracted objects (values) of `rdfs:label` properties, lower-cased them, and tokenized the values at space characters. We further excluded tokens shorter than three and longer than 25 characters. Afterward, we calculated the TF-IDF [72] value for each token while excluding tokens that appeared in less than 10 and in maximal 200 datasets, in order to reduce the influence of noise. This resulted in a feature vector consisting of 1 440 attributes. For LAB, we could only gather data for 455 datasets, as the remaining did not make use of the `rdfs:label` property.

Text from `rdfs:comment` (COM): We also extracted the values describing entities using the `rdfs:comment` property. We extracted all values of the comment property, and proceed in the same way as with the LAB feature. We lower-case all values and tokenize them at space characters and filtered out all values shorter than 3 characters and longer that 25 characters. This property is used by only 252 datasets, and not by the whole datasets in the cloud. For this feature we got 1 231 attributes. In difference from the LAB feature vector, we did not filter out tokens that were used by less than 10 datasets or more than 200 datasets. This was because the number of the datasets that were using the `rdfs:comment` was only 252 in whole LD cloud.

Vocabulary Description from LOV (VOCDES): LOV provides metadata about the vocabularies found in the LD cloud are provided. Among different metadata, it is also given the description in natural language for each vocabulary. From this description we can understand for which domain or topic we could

use this vocabulary. In the LOV website, there exist 581[7] different vocabularies. While in LD as described in the VOC feature vector there are 1453 different vocabularies. From 1438 vocabularies in LD, only 119 have a description in LOV, thus for 1319 vocabularies used in LD we do not have a description.

Top-Level Domains (TLD): Another feature which might help to assign datasets to topical categories is the top level domain of the dataset. For instance, government data is often hosted under the `.gov` top-level domain, whereas library data might be found more likely on `.edu` or `.org` top-level domains[8].

In & Outdegree (DEG): In addition to vocabulary-based and textual features, the number of outgoing RDF links to other datasets and incoming RDF links from other datasets could provide useful information for classifying the datasets. This feature could give a hint about the density of the linkage of a dataset, as well as the way the dataset is interconnected within the whole LD cloud ecosystem.

We extracted all the described features separately from the crawled data. We were able to gather all features (except for LAB and COM) for 1001 datasets.

4.3.2 Classification Approaches

Classification problem has been widely studied in the database [87], data mining [111], and information retrieval communities [73], and aims at finding regularities in patterns of empirical data (training data). The problem of classification is defined as follows: given a set of training records $\mathcal{D}$= $\{\mathcal{X}_1, \mathcal{X}_2,...\mathcal{X}_n\}$every record should be labeled with a class value drawn from a set of l different discrete values indexed by $\{1, 2,...l\}$. We choose to test different classification approaches, kNN, Naive Bayse and J48. Although there are tons of alternative classification algorithms available, we selected the ones for which the need for tuning is not too large, as for example the *support vector machines* because we do not want to overfit our learners by parameter tuning. The overfitting occurs when a model, does not fit the training data, thus is not reliable in making predictions.

[7]Numbers here refer to the version of LOV at the time when experiments for the topic classification were running in 2015.

[8]We restrict ourselves to top-level domains, and not public suffixes

k-Nearest Neighbour: kNN is one of the oldest non-parametric classification algorithms [15]. The training examples are vectors described by n dimensional numeric attributes. In kNN classification an object is classified by a majority vote of its neighbours, with the object being assigned to the class most common among its k nearest neighbours measured by a distance function. Choosing the right k value is done by inspecting the dataset first. In our experiments, based on the preliminary experiments on a comparable but disjunct set of data, we found that a k equal to 5 performs best. *Euclidean* measure is a good distance measure to use if data used as input are of similar type, e.g., all data are measured by the same metric such as heights and widths. While *Jaccard* distance is a good measure when the data in input are of different types, e.g., data are measured by different metrics such as age, weights, gender, etc. For this reason we used *Euclidean*-similarity for the binary term vectors and *Jaccard*-similarity for the relative term occurrence vectors as it will be described in 4.3.4.

J48 Decision Tree: Decision Trees are a powerful classification algorithms that run a hierarchical division of the underlying data. The most known algorithms for building decision trees are Classification and Regression Trees [33] and ID3 and C4.5 [115]. The decision tree is a tree with decision nodes which has two or more branches and leaf nodes that represents a classification or a decision. The splitting is based on the feature that gives the maximum information gain or uses entropy to calculate the homogeneity of a sample. The leaf node reached is considered the class label for that example. We use the *Weka* implementation of the C4.5 decision tree [114] called J48. Many algorithms try to prune their results. The idea behind pruning is that apart from producing fewer and more interpreted results, you reduce the risk of overfitting to the training data. We build a pruned tree, using the default settings of J48 with a confidence threshold of 0.25 with a minimum of 2 instances per leaf.

Naive Bayes: As a last classification algorithm, we use Naive Bayes. A Naive Bayesian [118] model is easy to build, with no complicated iterative parameter estimation which makes it particularly useful for very large datasets. It is based on Bayes theorem with independence assumptions between predictors. It considers each feature to contribute independently to the probability that this example is categorized as one of the labels. Naive Bayes classifier

assumes that the effect of the value of a predictor (x) on a given class (c) is independent to the values of other predictors. This assumption is called class conditional independence. Although this classifier is based on the assumption that all features are independent, which is mostly a rather poor assumption, Naive Bayes in practice has shown to be a well-performing approach for classification [146].

4.3.3 Sampling techniques

The training data is used to build a classification model, which relates the elements of a dataset that we want to classify to one of the categories. In order to measure the performance of the classification model built on the selected set of features we use cross-validation. Cross-validation is used to assess how the results of the classification algorithm will generalize to an independent dataset. The goal of using cross-validation is to define a dataset to test the model learnt by the classifier in the training phase, in order to avoid overfitting. In our experiments we used a 10-fold cross-validation, meaning that the sample is randomly partitioned into 10 equal sized subsamples. Nine of the 10 subsamples are used as training data, while one is used as validation data. The cross-validation process is then repeated 10 times (ten folds), with each of the 10 subsamples used exactly once as the validation data. The 10 results from the folds can after be averaged in order to produce a single estimation. As we described in section 4.1 the number of datasets per category is not balanced and over half of them are assigned to the *social networking* category. For this reason we explore the effect of balancing the training data. Even though there are different sampling techniques, as in [44], we explored only three of them:

Down sampling: We down sample the number of datasets used for training until each category is represented by the same number of datasets; this number is equal to the number of datasets within the smallest category. The smallest category is *geographic* with 21 datasets.

Up sampling: We up sample the datasets for each category until each category is at least represented by the number of datasets equal to the number of datasets of the largest category. The largest category is *social networking* with 520 datasets.

No sampling: We do not sample the datasets, thus we apply our approach in the data where each category is represented by the number of datasets as in the distribution of LD in Fig. 4.2.

The first sampling technique, reduces the chance to overfit a model into the direction of the larger represented classes, but it might also remove valuable information from the training set, as examples are removed and not taken into account for learning the model. The second sampling technique, ensures that all possible examples are taken into account and no information is lost for training, but by creating the same entity many times can result in emphasizing those particular data.

4.3.4 Normalization techniques

As the total number of occurrences of vocabularies and terms is heavily influenced by the distribution of entities within the crawl for each dataset, we apply two different normalization strategies to the values of the vocabulary-level features VOC, CUri, PUri, LCN, and LPN:

Binary version (bin): In this normalization technique the feature vectors of each feature vector consist of 0 and 1 indicating the presence and the absence of the vocabulary or term.

Relative Term Occurrence (rto): In this normalization technique the feature vectors of each feature vector captures the fraction of the vocabulary or term usage for each dataset.

In Table 4.1 shows an example on how we create the binary *(bin)* and relative term occurrence *(rto)* given the term occurence for a feature vector.

TABLE 4.1: Example of *bin* and *rto* normalization

Feature Vectors Version	Feature Vector (fv)			
	fv_1	fv_2	fv_3	fv_4
Term Occurrence	10	0	2	6
Binary (bin)	1	0	1	1
Relative Term Occurrence	0,5	0	0,1	0,4

4.4 Results

In this section we first report the results of our experiments using different feature vectors for single topic in 4.4.1. Afterward, we apply our classification algorithms with the goal of the multi-topic classification and report the results in section 4.4.2.

4.4.1 Single-topic classification

In this section we report the results for the experiments for single topic classification of LD datasets. We first report the results of our experiments training different feature vectors in separation 4.4.1.1. Afterward, we combine all feature vectors for both normalization techniques and again train our classification algorithms considering the three sampling techniques and report the results in section 4.4.1.2.

4.4.1.1 Results of Experiments on Single Feature Vectors

For the first experiment we learn a model to classify LD datasets in one of the eight categories described in 4.1. In this experiment we considered VOC, LCN, LPN, CUri, PUri, DEG, TLD and LAB feature vectors applying the approach described in section 4.3. For the above feature vectors, we trained the different classification techniques as in 4.3.2 with different sampling techniques as in 4.3.3 and different normalization techniques as in 4.3.4.

In order to evaluate the performance of the three classification techniques, we use 10-fold cross-validation and report the average accuracy. Table 4.2 reports the accuracy that is reached using the three different classification algorithms with and without sampling the training data. *Majority Class* is the performance of a default baseline classifier always predicting the largest category: *social networking*. As a general observation, the schema based feature vectors (VOC, LCN, LPN, CUri, PUri) perform on a similar level, LAB, TLD and DEG show a relatively low performance and in some cases are not at all able to beat the trivial baseline. Classification models based on the attributes of the LAB feature vector perform on average (without sampling) around 20% above the majority baseline, but predict still in half of all cases the wrong category. Algorithm-wise, the best results

are achieved using the decision tree (J48) without balancing (maximal accuracy 80.59% for LCN_{rto}) and the k-NN algorithm, also without balancing for the PUri_{bin} and LPN_{bin} feature vectors. Comparing the two balancing approaches, we see better results using the up sampling approach for almost all feature vectors (except VOC_{rto} and DEG). In most cases, the category-specific accuracy of the smaller categories is higher when using up sampling. Using down sampling the learnt models make more errors for predicting the larger categories. Furthermore, when comparing the results of the models trained on data without applying any sampling approach, with the best model trained on sampled data, the models applied on non sampled data are more accurate except for the VOC_{bin} feature vectors. We see that the balanced approaches are in general making more errors when trying to predict datasets for the larger categories, like *social networking* and *government*.

TABLE 4.2: Single-topic classification results on single feature vectors

Classification Approach	Accuracy in % VOC *bin*	VOC *rto*	CUri *bin*	CUri *rto*	PUri *bin*	PUri *rto*	LCN *bin*	LCN *rto*	LPN *bin*	LPN *rto*	LAB	TLD	DEG
Majority Class	51.85	51.85	51.85	51.85	51.85	51.85	51.85	51.85	51.85	51.85	51.85	51.85	51.85
k-NN (no sampling)	77.92	76.33	76.83	74.08	**79.81**	75.30	76.73	74.38	**79.80**	76.10	53.62	58.44	49.25
k-NN (downsampling)	64.74	66.33	68.49	60.67	71.80	62.70	68.39	65.35	73.10	62.80	19.57	30.77	29.88
k-NN (upsampling)	71.83	72.53	64.98	67.08	75.60	71.89	68.87	69.82	76.64	70.23	43.67	10.74	11.89
J48 (no sampling)	78.83	79.72	78.86	76.93	77.50	76.40	**80.59**	76.83	78.70	77.20	63.40	67.14	54.45
J48 (down sampling)	57.65	66.63	65.35	65.24	63.90	63.00	64.02	63.20	64.90	60.40	25.96	34.76	24.78
J-48 (up sampling)	76.53	77.63	74.13	76.60	75.29	75.19	77.50	75.92	75.91	74.46	52.64	45.35	29.47
Naive Bayes (no sampling)	34.97	44.26	75.61	57.93	78.90	75.70	77.74	60.77	78.70	76.30	40.00	11.99	22.88
Naive Bayes (down sampling)	64.63	69.14	64.73	62.39	68.10	66.60	70.33	61.58	68.50	69.10	33.62	20.88	15.99
Naive Bayes (up sampling)	77.53	44.26	74.98	55.94	77.78	76.12	76.02	58.67	76.54	75.71	37.82	45.66	14.19
Average (no sampling)	63.91	66.77	77.10	69.65	78.73	75.80	78.35	70.66	79.07	76.53	52.34	45.86	42.19
Average (down sampling)	62.34	67.34	66.19	62.77	67.93	64.10	67.58	63.38	68.83	64.10	26.38	28.80	23.55
Average (up sampling)	75.30	64.81	71.36	66.54	76.22	74.40	74.13	68.14	76.36	73.47	44.81	33.92	18.52

4.4.1.2 Results on Experiments of Combined Feature Vectors

In the second experiment, we combine all the feature vectors that we used in the first experiment and again train our classification models.

As before, we generate a *binary* and *relative term occurrence* version of the vocabulary-based features. In addition, we create a second set (*binary* and *relative term occurrence*), where we omit the attributes from the LAB feature vector, as we wanted to measure the influence of this particular feature, which is only available for less than half of the datasets. Furthermore, we create a combined set of feature vectors consisting of the three best performing feature vectors from the previous section.

Table 4.3 reports the results for the five different combined feature vectors:

ALL$_{rto}$: Combination of the attributes from all eight feature vectors, using the *rto* version of the vocabulary-based features (This feature vector is generated for 455 datasets).

ALL$_{bin}$: Combination of the attributes from all eight feature vectors, using the *bin* version of the vocabulary-based features (This feature vector is generated for 455 datasets).

NoLab$_{rto}$: Combination of the attributes from all feature, without the attributes of the LAB feature vectors, using the *rto* version.

NoLab$_{bin}$: Combination of the attributes from all feature, without the attributes of the LAB feature vectors, using the *bin* version.

Best3: Includes the attributes from the three best performing feature vectors from the previous section based on their average accuracy: PUri$_{bin}$, LCN$_{bin}$, and LPN$_{bin}$

We can observe that when selecting a larger set of feature vectors, our model is able to reach a slightly higher accuracy of 81.62% than using just the attributes from one feature vector (80.59%, LCN$_{bin}$). Still the trained model is unsure for certain decisions and has a stronger bias towards the categories *publications* and *social networking*.

TABLE 4.3: Single-topic classification results on combined feature vectors

Classification Approach	Accuracy in %				
	ALL_{rto}	ALL_{bin}	$NoLab_{rto}$	$NoLab_{bin}$	Best3
k-NN (no sampling)	74.93	71.73	76.93	72.63	75.23
k-NN (down sampling)	52.76	46.85	65.14	52.05	64.44
k-NN (up sampling)	74.23	67.03	71.03	68.13	73.14
J-48 (no sampling)	**80.02**	77.92	79.32	79.01	75.12
J-48 (down sampling)	63.24	63.74	65.34	65.43	65.03
J-48 (up sampling)	79.12	78.12	79.23	78.12	75.72
Naive Bayes (no sampling)	21.37	71.03	**80.32**	77.22	76.12
Naive Bayes (down sampling)	50.99	57.84	70.33	68.13	67.63
Naive Bayes (up sampling)	21.98	71.03	**81.62**	77.62	76.32

4.4.1.3 Discussion

In the following, we discuss the results achieved by our experiments and analyze the most frequent errors of the best performing approach for the single topic classification of LD datasets. The best performing approach is achieved by applying Naive Bayes trained on the attributes of the $NoLab_{bin}$ feature vector using up sampling. This approach achieved an accuracy of 81.62%. We take a closer look at the confusion matrix of the second experiment described in Table 4.4, where on the left side we list the predictions by the learnt model, while the heading names the actual topical category of the dataset. As observed in the table, there are three kinds of errors which occur more frequently than 10 times.

TABLE 4.4: Confusion Matrix for the $NoLAB_{bin}$ feature vector, with Naive Bayes classification algorithm, on up sampling.

prediction	true social networking	true cross-domain	true publications	true government	true life science	true media	true user-generated content	true geographic
social networking	489	4	5	10	2	4	**11**	1
cross-domain	1	10	3	1	1	0	1	1
publication	8	10	54	9	4	4	2	2
government	3	4	**14**	151	1	2	0	2
life science	5	3	**12**	0	72	2	5	5
media	6	3	4	1	1	7	2	0
user-generated content	6	1	1	2	0	2	26	0
geographic	1	5	1	5	1	0	0	8

The most common confusion occurs for the *publication* domain, where a larger number of datasets are predicted to belong to the *government* domain. A reason for this is that government datasets often contain metadata about government statistics which are represented using the same vocabularies and terms (e.g.

skos:Concept) that are also used in the publication domain. This makes it challenging for a vocabulary-based classifier to distinguish those two categories apart. In addition, for example the `http://mcu.es/` dataset the *Ministry of Culture in Spain* was manually labeled as *publication* within the LD cloud, whereas the model predicts *government* which turns out to be a borderline case in the gold standard (information on the LD cloud). A similar frequent problem is the prediction of *life science* for datasets in the *publication* category. This can be observed, e.g., for the `http://ns.nature.com/publications/`, which describe the publications in *Nature*. Those publications, however, are often in the life sciences field, which makes the labeling in the gold standard a borderline case.

The third most common confusion occurs between the *user-generated content* and the *social networking* domain. Here, the problem is in the shared use of similar vocabularies, such as *foaf*. At the same time, labeling a dataset as either one of the two is often not so simple. In [124], it has been defined that *social networking* datasets should focus on the presentation of people and their inter-relations, while *user-generated content* should have a stronger focus on the content. Datasets from personal blogs, such as `www.wordpress.com` however, can convey both aspects. Due to the labeling rule, these datasets are labeled as *user-generated content*, but our approach frequently classifies them as *social networking*.

In summary, while we observe some true classification errors, many of the mistakes made by our approach actually point at datasets which are difficult to classify, and which are rather borderline cases between two categories. For this reason as it will be described in section 4.4.2, we investigate the problem of multi-topic classification of LD datasets.

4.4.2 Multi-topic classification

In this section we report the results from the experiments for multi-topic classification of LD datasets. We first report the results of using the different feature vectors separately as for the single-topic classification in section 4.4.2.1. Afterward, we report the results of experiments combining attributes from multiple feature vectors in section 4.4.2.2.

4.4.2.1 Results of Experiments on Single Feature Vectors

In this section we report the results for classifying LD datasets in more than one topical category described in 4.1, that we define as multi-topic classification of LD datasets.

The objective of multi-label classification is to build models able to relate objects with a subset of labels, unlike single-label classification that predicts only a single label. Multi-label classification has two major challenges with respect to the single-label classification. The first challenge is related to the computational complexity of algorithms especially when the number of labels is large, then these approaches are not applicable in practice. While the second challenge is related to the independence of the labels and also some datasets might belong to an infinite number of labels. One of the biggest challenge in the community is to design new methods and algorithms that detect and exploit dependencies among labels [89].

[136] gives an overview of different algorithms used in the multi-label classification problem. The most straightforward approach for the multi-label classification is the Binary Relevance (BR) [137]. *BR* reduces the problem of multi-label classification to multiple binary classification problems. Its strategy involves training a single classifier per each label, with the objects of that label as positive examples and all other objects as negatives. The most important disadvantage of the *BR*, is the fact that it assumes labels to be independent. Although *BR* have many disadvantages, it is quite simple and intuitive. It is not computationally complex compared to other methods and is highly resistant to overfitting label combinations, since it does not expect examples to be associated with previously-observed combinations of labels [117]. For this reason it can handle irregular labeling and labels can be added or removed without affecting the rest of the model.

Also, for the multi-label classification problem one challenge is the lack of benchmark datasets. Since for the datasets in LD we lack the presence of a gold standard, we build one and make available for further research in this topic[9]. Also, the work presented in [119], built a gold standard for the multi classification of RDF datasets, but this work was done before the presentation of TAPIOCA framework.

As discussed in section 4.4.1.3 from the results of the first experiments, when we classify datasets into one topical category, we learnt that one of the problems of

[9]https://github.com/Blespa/TopicalProfiling

misclassification of LD datasets is that some categories or topics overlap between each other. For example, it is not clear what datasets should go into *social networking* and which ones into *user-generated content*. User-generated content can cover different topical domains thus to avoid misclassification of datasets we remove this category from the list of topical categories for LD datasets. We face the same problem classifying datasets into the *cross-domain* category and any other category. Because under the *cross-domain* category, also datasets in *life science* domain, or *media* domain can be categorized, we removed this category from the list of topics that we used for the multi-topic classification of LD datasets. From eight categories in the single topic experiments, in the multi-topic classification we have only six categories *life science, government, media, publications, social networking* and *geographic*.

For the problem of multi-topic classification of LD datasets we first build the gold standard. Two researchers independently classified datasets in the LD, into more than one category. We randomly select 200 datasets from the whole LD cloud. To assign more than one topical category to each dataset the researchers could access the descriptive metadata published into Mannheim Linked Data Catalog[10] which represents the metadata in the form of tags. Also, they had the possibility to take a deeper look inside the data itself. From the results, the researchers had an inter-rater agreement of 95.64%. Cases for which the assigned topics differ between the two researchers were further discussed with two professors.

TABLE 4.5: Distribution of number datasets per number of topics

Number of topics	1	2	3	4	5
Number of datasets	85	87	22	4	2

Table 4.5 shows the distribution of the number of datasets by the number of topics. As we can see, in our gold standard for the multi-topic classification, most of the datasets have one or two topics, while less than 3% of the datasets have more than four topics.

Multi-label classifiers can be evaluated from different points of view. Measures of evaluating the performance of the classifier can be grouped into two main groups: *example-based* or *label-based* [138]. The *example-based* measures compute the average differences of the actual and the predicted sets of labels over all examples. While the *label-based* measures decompose the evaluation with respect to each

[10] http://linkeddatacatalog.dws.informatik.uni-mannheim.de/

TABLE 4.6: Multi-topic classification results on single feature vectors

Classification Approach / Approach	Micro -averaged measure											
	CUri						LCN					
	bin			*rto*			*bin*			*rto*		
	P	*R*	*F*	*P*	*R*	*F*	*P*	*R*	*F*	*P*	*R*	*F*
k-NN (no sampling)	0.66	0.20	0.31	0.65	0.18	0.29	**0.68**	**0.21**	**0.32**	0.34	0.25	0.29
k-NN (downsampling)	0.58	0.21	0.31	0.55	0.02	0.28	0.53	0.22	0.31	0.68	0.19	0.30
k-NN (upsampling)	0.47	0.31	0.38	0.44	0.30	0.36	0.46	0.29	0.36	0.45	0.28	0.34
J48 (no sampling)	0.54	0.16	0.25	0.57	0.15	0.23	0.58	0.17	0.27	0.59	0.15	0.23
J48 (down sampling)	0.46	0.19	0.27	0.35	0.22	0.27	0.47	0.21	0.29	0.34	0.22	0.27
J-48 (up sampling)	0.50	0.20	0.28	0.51	0.18	0.26	0.50	0.21	0.29	0.52	0.18	0.27
Naive Bayes (no sampling)	0.41	0.53	0.46	0.45	0.41	0.43	**0.41**	**0.56**	**0.47**	0.45	0.40	0.42
Naive Bayes (down sampling)	0.35	0.46	0.39	0.41	0.41	0.41	0.38	0.42	0.40	0.39	0.41	0.40
Naive Bayes (up sampling)	0.41	0.53	0.46	0.45	0.41	0.43	**0.41**	**0.56**	**0.47**	0.45	0.40	0.42
Average (no sampling)	0.54	0.30	0.34	0.56	0.25	0.32	0.56	0.31	0.35	0.46	0.27	0.31
Average (down sampling)	0.46	0.29	0.32	0.44	0.22	0.32	0.46	0.28	0.33	0.47	0.27	0.32
Average (up sampling)	0.46	0.34	0.37	0.47	0.30	0.35	0.46	0.35	0.37	0.47	0.29	0.34

label. For *label-based* measures we can use two metrics; *macro-average* and *micro-average*. The *macro-average* averages the measures label-wise, while *micro-average* merges all label predictions and computes a single value over all of them. Macro-average measures give equal weight to each label, and are often dominated by the performance on rare labels. In contrast, micro-average metrics gives more weight to frequent labels. These two ways of measuring performance are complementary to each other, and both are informative [89]. For this experiment we will report the micro-average measure for precision (P), recall (R) and the harmonic mean between them, the f-measure (F).

Similarly, as for the single topic experiments, we also applied our classification algorithms on different feature vectors, taking into account also the different sampling and normalization techniques described in section 4.3.3 and 4.3.4. Also for the multi-topic classification of LD datasets we use a 10-fold cross-validation. For our first experiment we consider the LCN, LPN, CUri and PUri feature vectors as from the results of the experiments on the single topic classification they performed better with respect to the other feature vectors.

Table 4.6 and 4.7 show the micro-accuracy in terms of precision, recall and f-measure achieved by our classification algorithms. Algorithm-wise, the best results precision-wise are achieved using *k*-NN, without sampling with a P = 0.68, R = 0.21 and F = 0.32 trained on LCN binary, while for the best results for the harmonic mean between precision and recall are achieved for the same feature

TABLE 4.7: Multi-topic classification results on single feature vectors

Classification Approach	Micro -averaged measure											
	PURi						LPN					
	bin			*rto*			*bin*			*rto*		
Approach	*P*	*R*	*F*	*P*	*R*	*F*	*P*	*R*	*F*	*P*	*R*	*F*
k-NN (no sampling)	0.61	0.21	0.31	0.64	0.19	0.29	0.61	0.20	0.30	0.60	0.19	0.29
k-NN (downsampling)	0.52	0.22	0.31	0.58	0.19	0.29	0.55	0.22	0.32	0.56	0.21	0.30
k-NN (upsampling)	0.47	0.29	0.36	0.46	0.26	0.33	0.49	0.27	0.35	0.48	0.26	0.34
J48 (no sampling)	0.58	0.24	0.34	0.59	0.24	0.34	0.57	0.24	0.34	0.59	0.24	0.34
J48 (down sampling)	0.36	0.40	0.38	0.45	0.26	0.33	0.46	0.29	0.36	0.39	0.29	0.33
J-48 (up sampling)	0.53	0.27	0.35	0.55	0.27	0.36	0.56	0.29	0.39	0.54	0.27	0.36
Naive Bayes (no sampling)	0.61	0.21	0.31	0.64	0.19	0.29	0.61	0.20	0.30	0.60	0.19	0.29
Naive Bayes (down sampling)	0.52	0.22	0.31	0.58	0.19	0.29	0.55	0.22	0.32	0.56	0.21	0.30
Naive Bayes (up sampling)	0.47	0.29	0.36	0.46	0.26	0.33	0.49	0.27	0.35	0.48	0.26	0.34
Average (no sampling)	0.60	0.22	0.32	0,62	0,21	0,31	0,60	0,21	0,31	0.60	0,21	0,31
Average (down sampling)	0.47	0.28	0.33	0.54	0.21	0.30	0.52	0.24	0.33	0.50	0.24	0.31
Average (up sampling)	0.49	0.28	0.36	0.49	0.26	0.34	0.51	0.28	0.36	0.5	0.26	0.35

vector (LCN) training Naive Bayes on binary normalization technique. For the same feature vector and classification algorithm, the results achieved are in similar level for both sampling techniques; no sampling and up sampling; P=0.41, R=0.56 and F=0.47. Sampling-wise, the results achieved by the down-sampling are lower than the two other techniques. Also, normalization-wise there is a mixture in the results depending on the classification algorithm and the feature vector in the input.

4.4.2.2 Results of Experiments for Combined Feature Vectors

In the second experiment for the multi-topic classification of LD datasets we combine the feature vectors that we used in the first experiment and again train our classification algorithms. Table 4.8 shows the results of ALL feature vector and the combination of CUri, PUri, LCN and LPN.

From the results we can observe that when selecting a larger set of attributes, our model is not able to reach a higher performance than using only the attributes from one feature vector (P=0.68, R=0.21, F=0.32). Our model is precision-oriented and reach a satisfying precision but the recall is very low, which means that our model is not able to retrieve the right topic for the LD datasets. The highest performance for the experiments taking as input a combination of features is achieved

by training LCN and LPN binary vector as input for Naive Bayes on no sampling data P=0.42, R=0.48 and F=0.45.

TABLE 4.8: Multi-topic classification results on combined feature vectors

Classification Approach	Micro -averaged measure																	
	PUri & CUri						LPN & LCN						ALL					
	bin			*rto*			*bin*			*rto*			*bin*			*rto*		
Approach	*P*	*R*	*F*	*P*	*R*	*F*	*P*	*R*	*F*	*P*	*R*	*F*	*P*	*R*	*F*	*P*	*R*	*F*
k-NN (no sampling)	0.66	0.19	0.29	0.60	0.13	0.22	0.65	0.18	0.29	0.58	0.15	0.23	0.44	0.07	0.12	0.54	0.13	0.21
k-NN (downsampling)	0.53	0.23	0.32	0.56	0.16	0.25	0.53	0.23	0.32	0.51	0.19	0.28	0.42	0.08	0.13	0.51	0.14	0.22
k-NN (upsampling)	0.47	0.27	0.34	0.42	0.21	0.28	0.49	0.26	0.34	0.46	0.21	0.29	0.43	0.12	0.18	0.48	0.18	0.26
J48 (no sampling)	0.58	0.23	0.33	0.58	0.23	0.33	0.57	0.01	0.02	0.56	0.21	0.31	0.58	0.25	0.35	0.57	0.23	0.33
J48 (down sampling)	0.36	0.38	0.37	0.33	0.24	0.28	0.45	0.29	0.35	0.35	0.29	0.32	0.44	0.31	0.37	0.44	0.33	0.38
J-48 (up sampling)	0.54	0.27	0.36	0.55	0.27	0.36	0.53	0.27	0.35	0.52	0.24	0.33	0.58	0.25	0.35	0.51	0.25	0.34
Naive Bayes (no sampling)	0.51	0.39	0.44	0.50	0.34	0.41	0.42	0.48	0.45	0.54	0.34	0.41	0.54	0.34	0.42	0.52	0.31	0.39
Naive Bayes (down sampling)	0.42	0.43	0.43	0.38	0.40	0.39	0.40	0.40	0.40	0.37	0.42	0.40	0.41	0.42	0.41	0.38	0.41	0.39
Naive Bayes (up sampling)	0.51	0.39	0.44	0.50	0.34	0.41	**0.53**	**0.36**	**0.43**	0.54	0.34	0.41	0.55	0.32	0.40	0.52	0.31	0.39
Average (no sampling)	0.58	0.27	0,35	0.56	0.23	0.32	0.55	0.22	0.25	0.56	0.23	0.32	0.52	0.22	0.30	0.54	0.22	0.31
Average (down sampling)	0.44	0.35	0.37	0.42	0.27	0.31	0.46	0.31	0.36	0.41	0.30	0.33	0.42	0.27	0.30	0.44	0.29	0.33
Average (up sampling)	0.51	0.31	0.38	0.49	0.27	0.35	0.52	0.30	0.37	0.51	0.26	0.34	0.52	0.23	0.31	0.50	0.25	0.33

4.4.2.3 Discussion

In the following, we discuss the results achieved by our experiments on the multi-topic classification of LD datasets and analyse the most frequent errors of the best performing approach. As from the results in section 4.4.2.1 and section 4.4.2.2 for the multi-topic classification of LD datasets the best performing approach in terms of harmonic mean is achieved training the LCN using Naive Bayes on no sampling data with a performance of P=0.41, R=0.56 and F=0.47. Consider the problem of classifying the datasets with two topics, e.g., *media* and *social networking*. A representative example is the `bbc.co.uk/music` dataset, which in our gold standard is labeled with both topics. Our classifier predicts it as belonging to only media category. This dataset except of including music data, contains also other *social networking* data as a result of the possibility to sign up and create a profile, follow other people or comment in different music posts. For this reason we classify this dataset in our gold standard also as belonging to the social networking category. The classifier failed to classify the second topic because the vocabularies and classes used in this dataset belong mostly to the *bbc* vocabulary which is used only in datasets belonging to *bbc.co.uk* domain. Since the classifier learnt that the datasets from the *social networking* category make no use of such vocabulary, it is difficult for it to classify also the `bbc.co.uk/music` into the *social networking* category.

Consider the problem of classifying the datasets with three labels, e.g., *government, publication* and *geographic*. One of the datasets belonging to these topics is `europa.eu`. Our model classifies it as belonging to *publication* and *government*. The model was not able to predict *geographic* as the third topic. Even though this dataset contains some geographical data for all countries in the European Union, for example `http://europa.eu/european-union/about-eu/countries/member-countries/italy_en` the amount of geographic data with respect to the government and publication data is smaller. In this small amount of geographical data, the classifier could not find similar attributes as those used for training, considering them to be noise and not assigning a topic.

For the datasets that have more than three topics, it is even harder for the classifier to predict all labels, if there are few examples (instances) belonging to each topic and they use similar vocabularies to define also instances that belong to other topics.

Because of the results discussed above indicate that only schema-level data are not a good input to the classifiers, we also exploited the text information in these datasets. For this reason we extracted the LAB and COM feature vectors. Then we manually checked the text from LAB and COM feature vectors for the datasets in the gold standard to understand if this information could be a good input. We were able to find significant text only for 15 datasets (out of 200 in the gold standard) while for all the others, the text was not in English, or rather it contained acronyms, or was encoded. Because the number of datasets containing significant text is very low, we did not further continue testing LAB and COM feature vectors as input for the classifier for the multi-topic classification of LD datasets. Some examples for the LAB feature set: Name According To, hgnc:10916, Comment 1, Blog2RSS, 010A, etc. Besides the LAB and COM, also the VOCDES feature vector was not considered in our experiments. From 1438 vocabularies that are used in LD, only 119 have a description in LOV. From 119 vocabularies with a description, 90 of them are used in less than 10 datasets, while 5 of them are used in more than 200 datasets. For this reason we did not use the description of vocabularies in LOV as a feature vector for our classifiers.

4.5 Summary

In this chapter we discussed the problem of topic profiling. With the aim to automatically classify datasets in the LD cloud in one of the topical categories, we extracted different feature vectors and trained different classifiers. Furthermore, we investigated three sampling techniques and two normalization strategies. We exploited the problem of assigning a single-topic or more than one topic to LD dataset. For the single-topic we achieved an accuracy of around 82%. We did not achieve an accuracy of 100% because some datasets turn out to be borderlines between two labels. The performance of the classification model for the multi-topic classification showed that our approach is precision-oriented. The error analysis of the misclassified cases showed that this is a more difficult task than the single topic classification, because dataset use same or very similar feature vector to describe entities. Moreover, the distribution of the datasets for each topical category highly influences the classifier. The distribution of instances belonging to different topics within a dataset is highly influencing the classifier. If the dataset contains only a few instances belonging to a topic, our classifier consider this information as noise.

Chapter 5

Schema-Based Profiling

In this chapter we describe schema - based profiling by introducing an approach for dataset summarization. We propose ABSTAT, which is a framework used to summarize datasets by describing relations between types. The structure of this chapter is organized as follows: section 6.1 gives a general overview of the problem of summarizing dataset and why is schema-based profiling important. The summarization model is presented in section 5.2. The implementation of the model and the algorithm for the summarization are given in section 5.3. We evaluate ABSTAT's summaries from different perspectives and the experimental results are presented in section 5.4 while conclusions end this chapter in section 5.5.

5.1 Overview

The fundamental objective of the Semantic Web is the production of a common framework that allows information to be shared and reused across applications. As we show in chapter 4 the number of datasets published as Linked Data has increased up to 1014 as of April 2014 and continues increasing each day. Because of this increasing amount of information stored each day into databases or triple stores, users can no longer explore the data by mere visual inspection. Even knowing the topical category that a dataset belongs to, it does not give a complete understanding of what is inside these data, thus further exploration steps should be taken. A dataset might belong to the *media* category but this information does not tell how resources are described. The lack of having an overview or a summary

about the data that we want to investigate leads to inconspicuous relations in the data and therefore contributes to an absence of comprehension of the space being explored. The first step toward the data analysis and data exploration is data summarization. The main goal of dataset summarization is the semantic compression of the characteristics of a dataset [121]. The problem of data summarization has been studied by different communities such as *database* [121, 144], *text summarization* [52, 101], *graph summarization* [133, 99] and *ontology summarization* [149, 85]. In this chapter we present an ontology-driven dataset summarization approach. In section 3.4 we presented the state-of-the-art techniques and tools that deal with problems related to dataset summarization. As described in state of the art, for big and complex datasets, a user may find it difficult to understand to what extent a dataset covers a domain of interest and structures its content [92, 149, 135, 109, 74]. Exploration systems should be able to allow users to investigate the data by choosing the attributes and relationships that are of interest, and then make use of these features to produce small and informative summaries.

Given a LD dataset, users should be able to answer to questions such as: What types of resources are described in the dataset? What properties are used to describe the resources? What types of resources are linked and by means of what properties? How many resources have a certain type and how frequent is the use of a given property? Remarkably, difficulties in answering those questions have several consequences for data consumption, resulting in low adoption of many valuable but unknown datasets [124].

As described in section 2.3 Linked Data make use of ontologies to describe the semantics of their data. They are used to classify classes, describe possible relationships and define constraints on using them. However, answering the above questions by only looking at ontologies is not easy.

Recall the user scenario example in section 1.1. Mrs. Eamla could reduce the searching space for her purpose by applying topic-based profiling proposed in 4. By applying this approach, she could find that two datasets *LinkedBrianz* and *DBpedia* have at least *media* as one of their topics, thus these two datasets can be good candidate for building the app. She wants to further understand the semantics and the structure of the data in these datasets, thus she starts exploring the ontology, as ontologies describe the semantics of the data.

At the time of writing, *DBpedia* uses 685 (local) classes and 2795 properties. Ontologies used to model a large amount of diverse data in a flexible way are often underspecified. The domain is not specified for 259 properties of the *DBpedia Ontology*, while the range is not specified for 187 properties, e.g., the property dbo:authority does not have a domain or range defined in the ontology. In relatively expressive ontologies like the *Music Ontology*, some connections between classes may be specified by means of OWL axioms, e.g., qualified range restrictions, which may be difficult to understand for many data practitioners. For *LinkedBrianz* dataset the domain is underspecified for 13 (out of 34 properties) while the range is underspecified for 15 properties. Finally, the ontology does not tell how frequently a certain modelling patterns occurs in the dataset. Thus, by only looking at the ontology, Mrs. Eamla is not able to understand the semantics of the data found in both datasets.

ABSTAT is an ontology-driven linked data summarization model proposed to mitigate the dataset understanding problem. In our view, a summary is aimed at providing a compact but complete representation of a dataset. With complete representation we refer to the fact that every relation between types that is not in the summary can be inferred. One distinguishing feature of ABSTAT is that it adopts a minimalization mechanism based on *minimal type patterns*. A minimal type pattern is a triple (C, P, D) that represents the occurrences of assertions $<a,P,b>$ in RDF data, such that C is a minimal type of the subject a and D is a minimal type of the object b. Minimalization is based on a *subtype graph* introduced to represent the data ontology. By considering patterns that are based on minimal types we are able to exclude several redundant patterns from the summary and to specify several formal characteristics of the summaries. As a consequence, summaries based on our model are rich enough to represent adequately the whole dataset, and small enough to avoid redundant information. The ABSTAT[1] framework supports users to query (via SPARQL), to search and to navigate the summaries through web interfaces. Other related work on data or ontology summarization have focused on complementary aspects of the summarization, such as the identification of *salient subsets* of knowledge bases (KB using different criteria [149, 74, 135, 109], e.g., connectivity. Other approaches do not represent connections between types as our model does [77, 8, 83].

[1] http://abstat.disco.unimib.it

5.2 Summarization Model

As we show in section 2.3, ontologies (or vocabularies) are used to specify the semantics of data modelled in RDF. Ontologies, which are usually represented in languages such as RDFS and OWL, specify the meanings of the elements of the ontology, e.g., classes, datatypes, properties, individuals, by means of logical axioms [128]. Although we do not focus on a specific ontological language, in this chapter we borrow the definition of a dataset from the definition of Knowledge Base (KB) in Description Logics (DL). In a Knowledge Base there are two components: a *terminology* defining the vocabulary of an application domain called TBox, and a set of *assertions* describing RDF resources in terms of this vocabulary called ABox [11].

Definition 5.1. (Dataset) A dataset is a couple $\Delta = (\mathcal{T}, \mathcal{A})$, where $\mathcal{T}$ is a set of terminological axioms, and $\mathcal{A}$ is a set of assertions.

The vocabulary of a dataset contains a set N^C of *types*, where with type we refer to either a class or a datatype, a set N^P of properties, a set of named individuals (resource identifiers) N^I and a set of literals L. We use symbols like C, C', ..., and D, D', ..., to denote types, symbols P, Q to denote properties, and symbols a,b to denote named individuals or literals. Types and properties are defined in the terminology and occur in assertions.

Observe that different datasets adopt different policies with respect to the inclusion of entailed assertions in the published assertions: for example, *DBpedia* explicitly includes the transitive closure of type inference in the published assertion set, while other datasets do not follow the same policy, e.g., *LinkedBrainz*. Our summarization model has to handle datasets that may have been published following different inference publication policies. However, we will briefly discuss the impact of different inference publication policies on the summarisation model in section 5.3.

Assertions in $\mathcal{A}$ are of two kinds: *typing assertions* of form $C(a)$, and *relational assertions* of form $P(a, b)$, where a is a named individual and b is either a named individual or a literal. We denote the sets of typing and relational assertions by $\mathcal{A}^C$ and $\mathcal{A}^P$ respectively. Assertions can be extracted directly from RDF data (even in absence of an input terminology). *Typing assertions* occur in a dataset as RDF triples $(x, \texttt{rdf:type}, C)$ where x and C are URIs, or can be derived from triples

$(x, P, y\hat{}\hat{}C)$ where y is a literal (in this case y is a typed literal), with C being its datatype. Without loss of generality, we say that x is an instance of a type C, denoted by $C(x)$, either x is a named individual or x is a typed literal. Every resource identifier that has no type is considered to be of type `owl:Thing` and every literal that has no type is considered to be of type `rdfs:Literal`. Observe that a literal occurring in a triple can have at most one type and at most one type assertion can be extracted for each triple. In contrast, an instance can be the subject of several typing assertions. A *relational assertion* $P(x, y)$ is any triple (x, P, y) such that $P \neq Q*$, where $Q*$ is either `rdf:type`, or one of the properties used to model a terminology (e.g. `rdfs:subClassOf`).

Abstract Knowledge Patterns (AKPs) are abstract representations of Knowledge Patterns, i.e., constraints over a piece of domain knowledge defined by axioms of a logical language, in the vein of Ontology Design Patterns [128].

Definition 5.2. (Abstract Knowledge Pattern (AKP)) An AKP is a triple (C, P, D) such that C and D are types and P is a property.

Intuitively, an AKP states that there are instances of type C that are linked to instances of a type D by a property P. For sake of clarity, we will use the term *pattern* to refer to an AKP in the rest of this thesis. In ABSTAT we represent a set of AKPs occurring in the dataset, which profiles the usage of the terminology. However, instead of representing every AKP occurring in the dataset, ABSTAT summaries include only a base of minimal type patterns, i.e., a subset of the patterns such that every other pattern can be derived using a subtype graph. In the following we better define these concepts and the ABSTAT principles.

Definition 5.3. (Pattern Occurrence) A pattern (C, P, D) *occurs* in a set of assertions $\mathcal{A}$ iff there exist some instances x and y such that $\{C(x), D(y), P(x, y)\} \subseteq \mathcal{A}$.

Patterns will also be denoted by the symbol π.

For datasets that publish the transitive closure of type inference (e.g., *DBpedia*), the set of all patterns occurring in an assertion set may be very large and include several redundant patterns. To reduce the number of patterns we use the observation that many patterns can be derived from other patterns if we use a *Subtype Graph* that represents types and their subtypes.

Definition 5.4. (Subtype Graph) A *subtype graph* is a graph $G = (\mathsf{N}^C, \preceq)$, where N^C is a set of type names (either class or datatype names) and $\preceq$ is a relation over N^C.

We always include two type names in N^C, namely `owl:Thing` and `rdfs:Literal`, such that every class is subtype of `owl:Thing` and every datatype is subtype of `rdfs:Literal`. One type can be subtype of none, one or more than one type.

Definition 5.5. (Minimal Type Pattern) A pattern (C, P, D) is a *minimal type pattern* for a relational assertion $P(a, b) \in \mathcal{A}$ and a terminology graph G iff (C, P, D) occurs in $\mathcal{A}$ and there does not exist a type C' such that $C'(a) \in \mathcal{A}$ and $C' \prec^G C$ or a type D' such that $D'(b) \in \mathcal{A}$ and $D' \prec^G D$.

Definition 5.6. (Minimal Type Pattern Base) A *minimal type pattern base* for a set of assertions $\mathcal{A}$ under a subtype graph G is a set of patterns $\widehat{\Pi}^{\mathcal{A},G}$ such that $\pi \in \widehat{\Pi}^{\mathcal{A},G}$ iff π is a minimal type pattern for some relation assertion in $\mathcal{A}$.

Observe that different minimal type patterns (C, P, D) can be defined for an assertion $P(a, b)$ if a and/or b have more than one minimal type. The minimal type pattern base excludes many patterns that can be inferred following the subtype relations and that are not minimal type for any assertion. In the graph represented in Figure 5.1 considering the assertion set $\mathcal{A} = \{P(a, b), C(a), A(a), F(b), D(b), A(b)\}$, there are six patterns occurring in $\mathcal{A}$, i.e., (C, P, D), (C, P, F), (C, P, A), (A, P, D), (A, P, F), (A, P, A). The minimal type pattern base for the dataset includes the patterns (E, Q, D), (E, R, T), (C, Q, D), (C, R, T) and (C, P, D) since E and C are minimal types of the instance c, while excluding patterns like (B, Q, D) or even (A, Q, A) since not B nor A are minimal types of any instance.

Definition 5.7. (Data Summary) A *summary* of a dataset $\Delta = (\mathcal{A}, \mathcal{T})$ is a triple $\Sigma^{\mathcal{A},\mathcal{T}} = (G, \Pi, S)$ such that: G is *Subtype Graph*, $\widehat{\Pi}^{\mathcal{A},G}$ is a *Minimal Type Pattern Base* for $\mathcal{A}$ under G, and S is a set of *statistics* about the elements of G and Π.

Statistics describe the occurrences of types, properties and patterns. They show how many instances have C as minimal type, how many relational assertions use a property P and how many instances that have C as minimal type are linked to instances that have D as minimal type by a property P.

The abstract representation provided by patterns has the practical advantage of characterizing the connections from/to instances of a given type. Moreover, the set

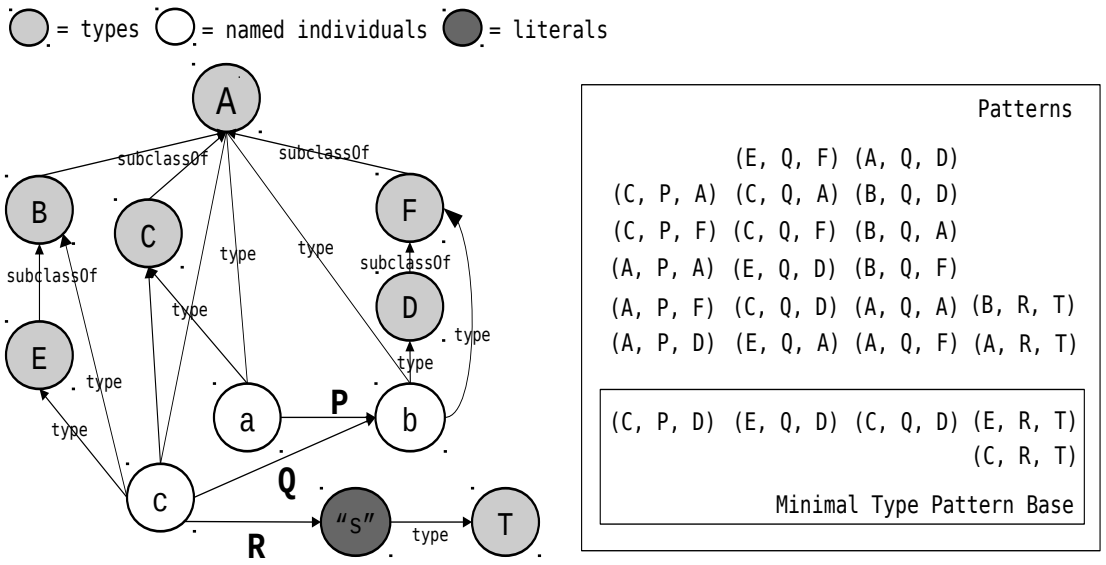


FIGURE 5.1: A small graph representing a dataset and the corresponding patterns.

of patterns containing a property describe the types of instances that they connect. The choice of using patterns as main components of a summary is motivated by the very goal of summarization: to represent the content of a dataset, that is, the set of its assertions.

5.3 Summary Extraction

Our summarization process, depicted in Fig. 5.2, takes in input an assertion set $\mathcal{A}$ and a terminology $\mathcal{T}$ and produces a summary $\Sigma^{\mathcal{A},\mathcal{T}}$. First, the typing assertion set $\mathcal{A}^C$ is isolated from the relational assertion set $\mathcal{A}^P$, while the subtype graph G is extracted from $\mathcal{T}$. Then, $\mathcal{A}^C$ is processed and the set of minimal types for each named individual is computed. Finally, $\mathcal{A}^P$ is processed in order to compute the minimal type patterns that will form the minimal pattern base $\widehat{\Pi}^{\mathcal{A},G}$. During each phase we keep track of the occurrence of types, properties and patterns, which will be included as statistics in the summary.

Subtype graph extraction. The subtype graph G^C is extracted by traversing all the subtype relations in $\mathcal{T}$. The subtype graph will be further enriched with types from external ontologies asserted in $\mathcal{A}^C$ while we compute minimal types of named individuals (i.e., *external* types). The subtype graph does not include equivalence relations.

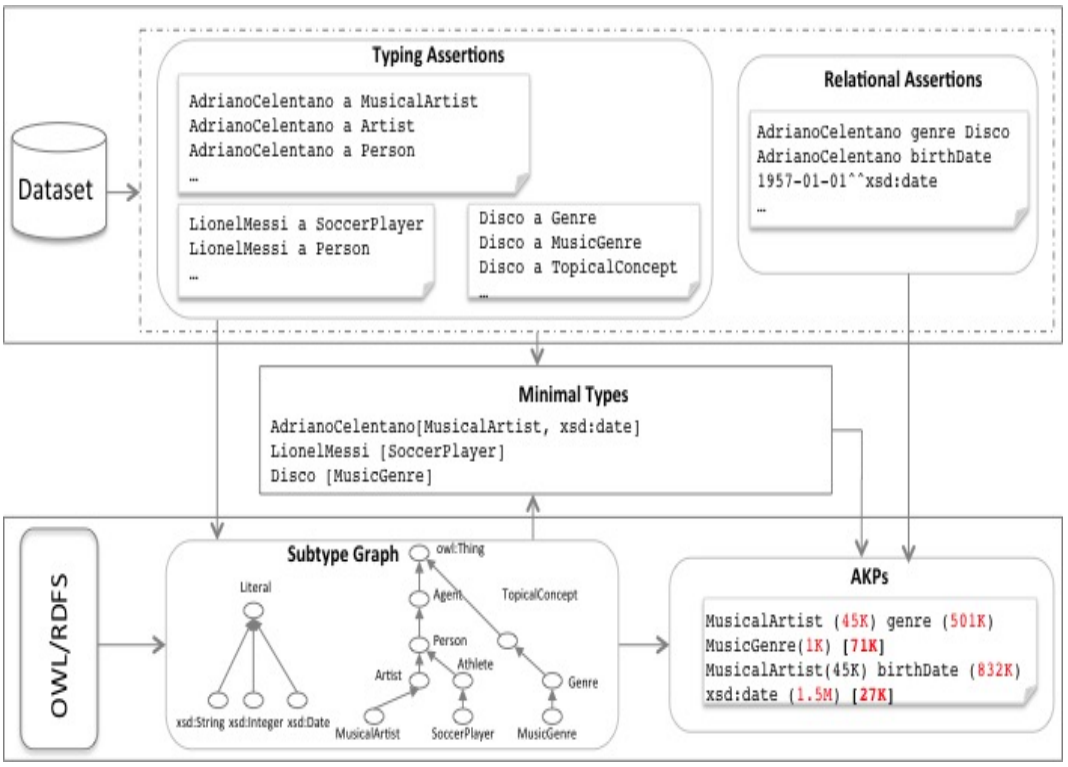


FIGURE 5.2: The summarization workflow.

For particular equivalence relations, it is possible to convert them in subtype relations, still preserving the partial ordering of the asserted type. We do not follow such approach because this can lead to counterintuitive or undesired inferences. For example, the `Village`, `PopulatedPlace` and `Place` types from the well known *DBpedia ontology* are equivalent to the type `Wikidata:Q532` (i.e., village). By including those equivalence relations in G^C, a correct but undesired inference could be that every `Place` is also a `Village` or that `Village`, `PopulatedPlace` and `Place` are equivalent. This real world example highlights an issue related to equivalence relations. Equivalence relations are often used to map named classes from different ontologies (e.g., by leveraging ontology alignment techniques) [55]. The existence of mappings leading to counterintuitive entailments can be explained by precise data management strategies. In the above mentioned equivalence relation, they may have been added to the ontology in order to map `Village` to `Wikidata:Q532` (i.e., villages with villages). Actually in ABSTAT we do not consider equivalence relations in the extraction of G. As a result, the extracted subtype graph G is complete not with respect to the whole terminology $\mathcal{T}$, but with respect to $\mathcal{T}$ without the equivalence relations.

Algorithm 1: Computation of the minimal types set for an instance x.

Input: $\mathcal{A}_x^C$ type assertions about the entity x, G^C subtype graph
Output: the set M_x the minimal types of x

```
1  M_x = {owl:Thing};
2  for C(x) ∈ A_x^C do
3      if C ∉ G^C then
4          G^C = G^C ∪ C
5      Sup = findSuperTypes(C, M, G^C);
6      if Sup ≠ ∅ then
7          M_x = M_x \ Sup;
8          M_x = M_x ∪ {C}
9      Sub = findSubTypes(C, M, G^C);
10     if Sub = ∅ then
11         M_x = M_x ∪ {C}
12 return M_x;
```

Minimal types computation. For each instance x, we compute the set M_x of minimal types with respect to the subtype graph G^C. Given x we select all typing assertions $C(x) \in \mathcal{A}^C$ and form the set $\mathcal{A}_x^C$ of typing assertions about x. Algorithm 1 presents the pseudocode for computing M_x. We first initialize M_x with the type `owl:Thing` (line 1), then we iteratively process all the type assertions. At each iteration we select a type C and remove from M_x all the supertypes of C according to G^C (lines 6-10). Then, if M_x does not contain any subtype of C according to G^C we can add C to M_x (lines 11-14). Notice that one preliminary step of the algorithm is to include C in G^C if it was not included during the subtype graph extraction phase (lines 3-5). Consequently, if a type C is not defined in the input terminology, is automatically considered as a minimal type for all the instances x. This approach allows us to handle instances of types from ontologies not included in the input terminology. At the moment, we do not retrieve such ontologies, but the minimal type pattern base with respect to the terminology graph G ensures that once they are added to the $\mathcal{T}$, the size of the summary will not increase.

Minimal type pattern base computation. For each relational assertion $P(x, y) \in \mathcal{A}^P$, we get the minimal types sets M_x and M_y. For all $C, D \in M_x, M_y$ we add a pattern (C, P, D) to the minimal types pattern base. If y is a literal value we consider its explicit type if present, `rdfs:Literal` otherwise. In this phase the subproperty graph is enriched with properties that are not defined by the terminology, but still occur in at least one pattern (i.e., *external* properties).

Summary storing and presentation. Once extracted, a summary $\Sigma^{\mathcal{A},\mathcal{T}}$ is

stored, indexed and made accessible through two user interfaces, i.e., ABSTAT-Browse and ABSTATSearch, and a SPARQL endpoint. The ABSTAT[2] homepage is shown in Fig. 5.3.

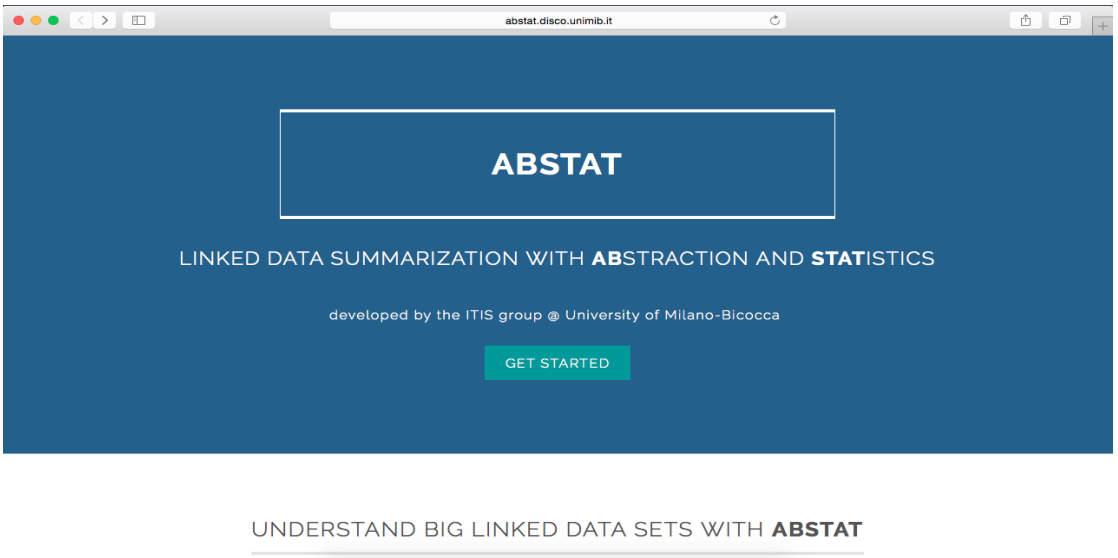


FIGURE 5.3: ABSTAT homepage

ABSTATBrowse[3] supports the interactive visualization of the summaries as shown in Fig. 5.4.

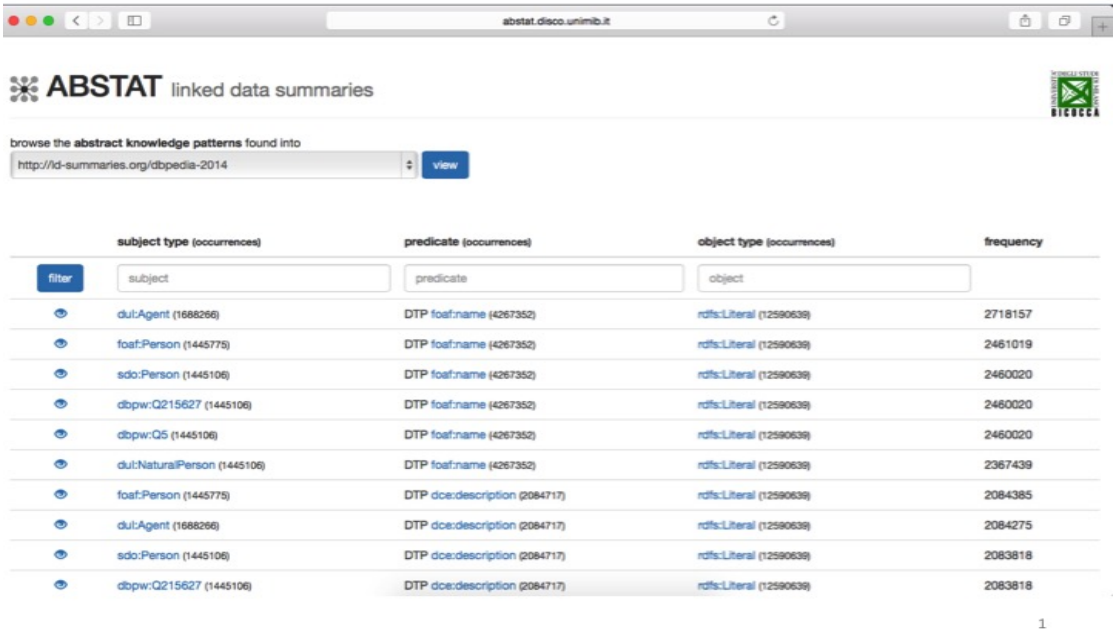


FIGURE 5.4: ABSTAT browse

[2] http://abstat.disco.unimib.it
[3] http://abstat.disco.unimib.it/browse

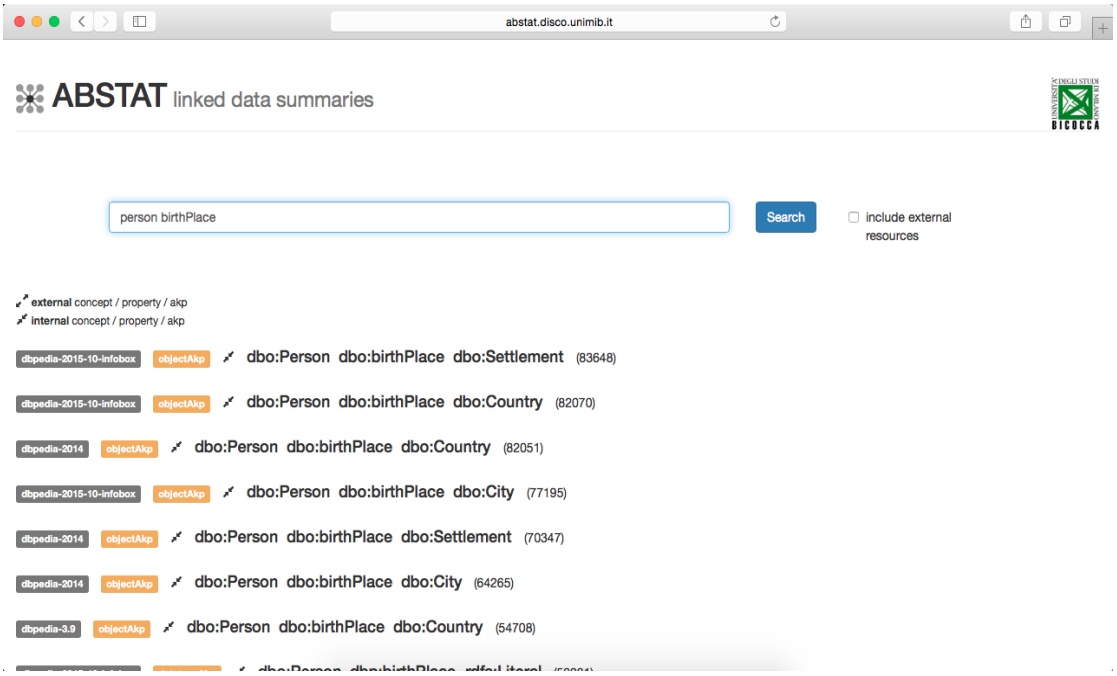


FIGURE 5.5: ABSTAT search

ABSTATSearch[4] implements a full-text search functionality over a set of summaries and is shown in Fig. 5.5. Types, properties and patterns are represented by means of their local names (e.g., `Person`, `birthPlace` or `Person birthPlace Country`) and conveniently tokenized, stemmed and indexed to support full text queries. Since the patterns are represented as triples, the study of principled and specialized matching and ranking techniques is an interesting extension that we leave for future work.

ABSTAT also provides a SPARQL endpoint[5] to query the summaries.

Recalling the user scenario, in the following we will describe how ABSTAT would support our app developer, Mrs. Eamla to understand the datsets for her need. She would access the ABSTAT homepage and select the ABSTATBrowse. After selecting the dataset, e.g., *LinkedBrainz*, she visualizes the AKPs and their occurrence. She can either navigate the summary or filter out the AKPs by subject and/or object types, and/or by property and look at their occurrence. In the filtering area ABSTAT enables a self filling feature to support her in identifying the types of interest (e.g., `mo:SoloMusicArtist`, `mo:MusicArtist`, `dbo:MusicalArtist` or `dbo:Band`).

[4] http://abstat.disco.unimib.it/search
[5] http://abstat.disco.unimib.it/sparql

She finds that *LinkedBrianz* contains 237 464 mo:MusicArtist while in *DBpedia 2014* there are 45 089 `dbo:MusicalArtist`, i.e., 80% less than in *LinkedBrianz*. On the other hand musical aritist are described in *DBpedia* with a richer and more diverse set of properties than in *LinkedBrianz*. Looking at these summaries is also possible to find errors in the datasets. There is only one occurrence of the AKP (`mo:MusicArtist, mo:member_of, mo:MusicArtist`) in *LinkedBrianz* which may depend on a mistyped instance. Several counterintuitive AKPs, such as (`dbo:Band, dbo:genre, dbo:Band`) and (`dbo:Band, dbo:instrument, dbo:Band`) which may reveal incorrect data, occur quite frequently also in *DBpedia*. In order to gain a comparable level of understanding of the two datasets without the help of ABSTAT, she would have had to (1) manually inspected the two ontologies and understood their semantics, and (2) written many SPARQL queries to get the statistics about their number of instances. Finally, she would have hardly spotted the incongruences highlighted by ABSTAT, with the risk of developing unreliable service for her app.

A summarization can be represented in different output formats. In the context of LD, it seems to be reasonable to use LD standards for modelling and storing the summarization as well. Thus, the output of our approach uses RDF for representing and serializing the summarization. This allows to write the summarization to a RDF file or directly to a triple store. One of the principles of LD is to reuse existing terms and vocabularies instead of inventing new ones. Because of this, we reuse SKOS[6] taxonomy.

To represent summaries in RDF we designed the *Linked Data Summaries* (LDS) ontology in OWL. The idea behind LDS is to provide a core vocabulary, which can be easily extended with additional summarization concepts or properties and further statistics about the summarization of the dataset.

5.4 Experimental Evaluation

The state-of-the-art of data exploration applications reveals the little attention accorded to the final users exploratory needs and capabilities when it comes to designing, developing and evaluating the applications. Studies in usability are scant and lacking in depth, reducing the work to a mere evaluation of technical aspects.

[6] https://www.w3.org/2004/02/skos/

TABLE 5.1: Datasets and summaries statistics.

	Relational	Typing	Assertions	Types (Ext.)	Properties (Ext.)	Patterns
db2014-core	~ 40.5M	~ 29.7M	~ 70.1M	869 (85)	1439 (15)	**171340**
db3.9-infobx	~ 96.3M	~ 19.7M	~ 116.4M	821 (58)	62572 (14)	**732418**
lb	~ 180.1M	~ 39.6M	~ 221.7M	21 (9)	33 (0)	**161**

In particular, the approaches based on minimalization still lack the evaluation from a cognitive perspective able to validate the proposed solutions.

We evaluate our summaries from different, orthogonal perspectives. We measure the *compactness* of ABSTAT summaries and compare the number of their patterns to the number of patterns extracted by Loupe [92], an approach similar to ours that does not use minimalization. The *informativeness* of our summaries are evaluated with two experiments. In the first experiment we show that our summaries provide useful insights about the semantics of properties, based on their usage within a dataset. In the second experiment, we conduct a preliminary user study to evaluate if the exploration of the summaries can help users in query completion tasks. In our evaluation we use the summaries extracted from three linked datasets: *DBpedia Core 2014* (**db2014-core**)[7], *DBpedia 3.9* (**db3.9-infobox.**)[8] and *LinkedBrainz* (**lb**). **db2014-core** and **db3.9-infobox** data sets are based on the *DBpedia ontology* while the **lb** dataset is based on the *Music ontology*. *DBpedia* and *LinkedBrainz* have complementary features and contain real and large data. For this reason they have been used, for example, in the evaluation of *Question and Answering systems* [88].

5.4.1 Compactness

Table 5.1 provides a quantitative overview of datasets and their summaries. To evaluate the compactness of a summary we measure the *reduction rate*, defined as the ratio between the number of patterns in a summary and the number of assertions from which the summary has been extracted.

Our model achieves a *reduction rate* of ∼0.002 for **db2014-core**, ∼0.006 for **db3.9-infobox**, and ∼6.72 $\times 10^{-7}$ for **lb**. Comparing the reduction rate obtained by our model with the one obtained by Loupe (∼0.01 for *db2014-core* and ∼7.1

[7]The *DBpedia 2014* version with mapping based property only.
[8]The *DBpedia Core 3.9* version plus automatically extracted properties.

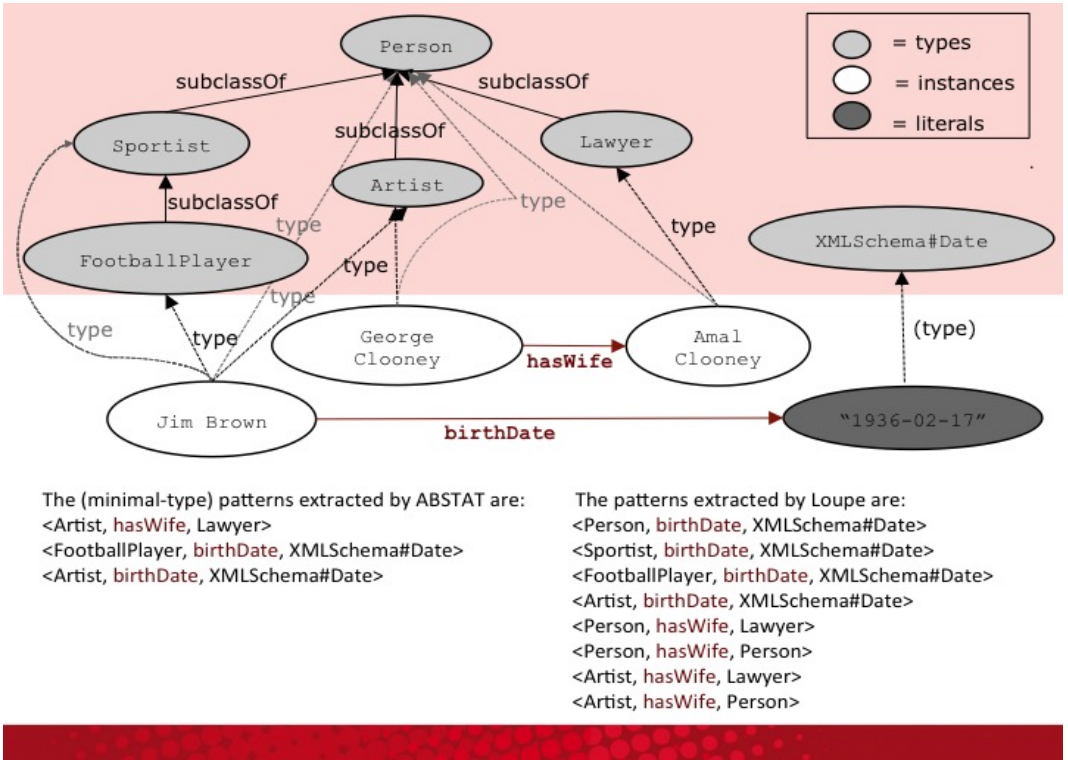


FIGURE 5.6: AKPs extraction

$\times 10^{-7}$ for *LinkedBrainz*)[9] we observe that the summaries computed by our model are more compact, as we only include minimal type patterns. Loupe instead, does not apply any minimalization technique thus its summaries are less compact. The effect of minimalization is more observable on *DBpedia* datasets, since the *DBpedia* terminology specifies a richer subtype graph and has more typing assertions. We observe also that 85 external types were added to the **db2014-core** subtype graph and 58 to the **db3.9-infobox** subtype graph during the minimal types computation phase as they were not part of the original terminology, and thus are considered by default as minimal types. While for *LinkedBrainz* the number of external types which are considered by default as minimal type is 9.

Fig. 5.6 shows how patterns are extracted by ABSTAT and Loupe. The summary extracted by ABSTAT is more compact with respect to the summary produced by Loupe, 3 and 8 patterns respectively.

[9]We do not provide the reduction rate for db3.9-infobox as this dataset is not summarized in Loupe

TABLE 5.2: Total number of properties with unspecified domain and range in each dataset.

	Domain (%)	Range (%)	Domain-Range (%)
db2014-core	259 (∼18%)	187 (∼13%)	48 (∼3.3%)
db3.9-infobox	61368 (∼98%)	61309 (∼98%)	61161 (∼97%)
lb	13 (∼39%)	15 (∼45%)	13 (∼39%)

5.4.2 Informativeness

Insights about the semantics of the properties. Our summaries convey valuable information on the semantics of properties for which the terminology does not provide any domain and/or range restrictions. Table 5.2 provides an overview of the total number of unspecified properties from the datasets. For example, around 18% of properties from **db2014-core** dataset have no domain restrictions while 13% have no range restrictions. Observe that this dataset is the most curated subset of *DBpedia* as it includes only triples generated by user validated mappings to *Wikipedia* templates. In contrast for **db3.9-infobox** dataset which includes also triples generated by information extraction algorithms, most of the properties (i.e., the ones from the `dbpepdia.org/property` namespace) are not specified within the terminology.

In general, underspecification may be the result of precise modelling choices, e.g., the property `dc:date` from the **lb** dataset. This property is intentionally not specified in order to favor its reuse, being the Dublin Core Elements (i.e., `dc`) a general purpose vocabulary. Another example is the `dbo:timeInSpace` property from the **db2014-core** dataset, whose domain is not specified in the corresponding terminology. However, this property is used in a specific way as demonstrated by patterns (`dbo:Astronaut`, `dbo:timeInSpace`, `xsd:double`) and (`dbo:SpaceShuttle` `dbo:timeInSpace`, `xsd:double`). Gaining such understanding of the semantics of the `dbo:timeInSpace` property by looking only at the terminology axioms is not possible.

We can push our analysis further to a more fine grained level. Fig. 5.7 provides an overview of the number of different minimal types that constitute the domain and range of unspecified properties extracted from the summary of the **db2014-core** dataset. The left part of the plot shows those properties whose semantics is less "clear", in the sense that their domain and range cover a higher number of different minimal types e.g., the `dbo:type` property. Surprisingly, the `dbo:religion` property is among them: its semantics is not as clear as one might think, as its

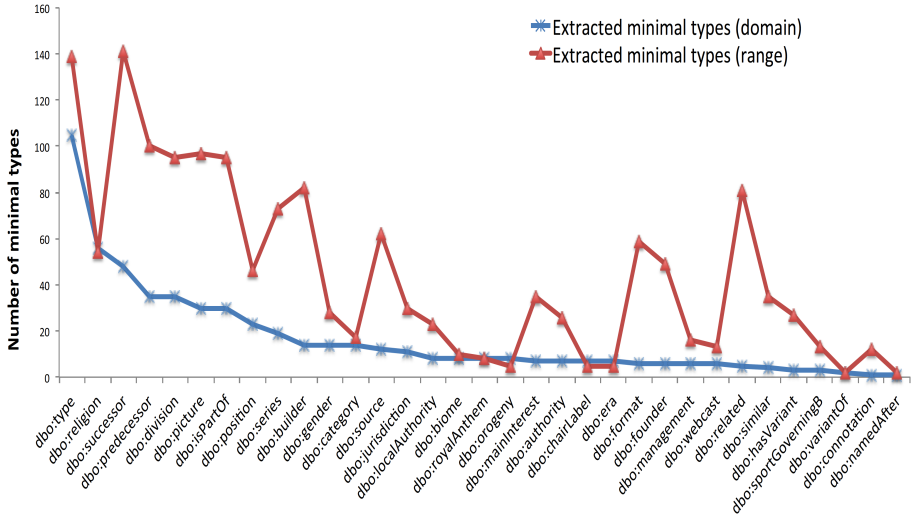


FIGURE 5.7: Distribution of the number of minimal types from the domain and range extracted for not specified properties of the **db2014-core** dataset.

range covers 54 disparate minimal types, such as `dbo:Organization`, `dbo:Sport` or `dbo:EthnicGroup`. Conversely, the property `dbo:variantOf`, whose semantics is intuitively harder to guess, is used within the dataset with a very specific meaning, as its domain and range covers only two minimal types: `dbo:Automobile` and `dbo:Colour`.

Small-scale user study. Formulating SPARQL queries is a task that requires prior knowledge about the dataset. ABSTAT could support users that lack such knowledge by providing valuable information about the content of the dataset. We designed a user study based on the assignment of cognitive tasks related to query completion. We selected a set of queries from the *Questions and Answering in Linked Open Data* benchmark[10] [139] for the **db3.9-infobox** dataset. The selected queries were taken from logs of the PowerAqua QA system and are believed to be representative of realistic information needs [88], although we cannot guarantee that they cover every possible information needed. We provided the participants the query in natural language and a "template" of the corresponding SPARQL query, with spaces intentionally left blank for properties and/or classes. For example, given the natural language specification *Give me all people that were born in Vienna and died in Berlin*, we asked participants to fill in the blank spaces. In Fig. 5.8 shows a screeshot of this query.

[10] http://greententacle.techfak.uni-bielefeld.de/~cunger/qald/

FIGURE 5.8: Example of the query for the user study

We selected five queries of increasing length, defined in terms of the number of triple patterns within the `WHERE` clause; one query of length one, two of length two and two of length three. Intuitively, the higher the query length, the more difficult it is to be completed.

We could use a limited number of queries because the tasks are time-consuming and fatigue-bias should be reduced [108]. The time needed to perform the 5 tasks are: min=18.4m, max=59.2m, avg=38.6m, + task description and training=20m. However, using 5 tasks is coherent with related work ("Typical experiments would have 20-60 participants, who are given 10-30 minutes of training, followed by all participants doing the same 2-20 tasks during a 1-3 hour session [108]). Overall 20 participants with no prior knowledge about the ABSTAT framework were selected. The number of users involved in our experiment is equal to or higher than the number of users used in related work in the Semantic Web Community (20 in [132], and 13 in [123]). Before queries completion task we profiled all the participants in terms of knowledge about SPARQL, data modelling, *DBpedia* dataset and ontology, so as to create two homogeneous groups: **abstat** and **control**. We trained only the participants from the first group for about 20 minutes on how to use ABSTAT.

TABLE 5.3: Results of the user study.

Group	**Avg. Completion Time (s)**	**Accuracy**
	query 1 - *How many employees does Google have?* - length 1	
abstat	**358.9**	**0.9**
control	380.6	0.8
	query 2 - *Give me all people that were born in Vienna and died in Berlin* - length 2	
abstat	356.3	**1**
control	**346.9**	0.8
	query 3 - *Which professional surfers were born in Australia?* - length 2	
abstat	476.6	0.6
control	**234.24**	**0.7**
	query 4 - *In which films directed by Gary Marshall was Julia Roberts starring?* - length 3	
abstat	**333.4**	0.9
control	445.6	0.9
	query 5 - *Give me all books by William Goldman with more than 300 pages* - length 3	
abstat	**233.4**	**1**
control	569.8	0.7

Both groups executed SPARQL queries against the **db3.9-infobox** dataset through the same interface (ABSTATQuery[11]) and were asked to submit the results they considered correct for each query. We measured the time spent to complete each query and the correcteness of the answers. The correcteness of the answers is calculated as the ratio between the number of correct answers to the given query agains the total number of answers. Table 5.3 provides the results of the performance of the users on the query completion task[12]. The time needed to perform the 5 queries from all partecipiants in average is 38.6m, while the minimum and the maximum time is 18.4m and 59.2m respectively. The independent *t-test*, showed that the time needed to correctly answer Q5, the most difficult query, was statistically significant for two groups. There was a significant effect between two groups, $t(16) = 10.32$, $p < .005$, with mean time for answering correctly to Q5 being significantly higher (+336s) for the control group than for abstat group.

Observe that the two used strategies to answer the queries by participants from the **control** group were: to directly access the public web page describing the *DBpedia* named individuals mentioned in the query and very few of them submitted explorative SPARQL queries to the endpoint. Most of the users searched on *Google* for some entity in the query, then consulted *DBpedia* web pages to find the correct answer. *DBpedia* is arguably the best searchable dataset, which is why this explorative approach was successful for relatively simple queries. However, this

[11] http://abstat.disco.unimib.it/search

[12] The raw data can be found at http://abstat.disco.unimib.it/downloads/user-study

explorative approach does not work with other non-indexed datasets (e.g., *LinkedBrainz*) and for complex queries. Instead, participants of the **abstat** group took advantage of the summary, obtaining huge benefits in terms of average completion time, accuracy, or both. Moreover, they achieved increasing accuracy over queries at increasing difficulty, still performing the tasks faster. We interpret the latter trend as a classical cognitive pattern, as the participants became more familiar with ABSTATBrowse and ABSTATSearch web interfaces.

The noticeable exception is query 3. In particular, participants from the **abstat** group completed the query in about twice the time of participants from **control** group. This is due to the fact that the individual `Surfing` (which is used as object of the property `dbo:occupation`) is classified with no type other than `owl:Thing`. As a consequence, participants from the **abstat** group went trough a more time consuming trial and error process in order to guess the right type and property. Participants from the **abstat** group finally came to the right answer, but after a longer time. This issue might be solved by applying state-of-the-art approaches for type inference on source RDF data [107] and suggest possible improvements of ABSTAT for example including values for classes that are defined by closed and relatively small instance sets.

5.5 Summary

Getting an understanding of the shape and nature of the data from large Linked Datasets is a complex and a challenging task. In this chapter, we presented ABSTAT a minimalization-based summarization model to support dataset understanding. Based on the results of our experiments we show that our summarization framework is able to provide both *compact* and *informative* summaries for a given dataset. We showed that using ABSTAT framework the summaries are more compact than the ones generated from other models and they also help the user to gain insights about the semantics of underspecified properties in the ontology. The results of our preliminary experiment showed that exploring ABSTAT' summaries help users formulating SPARQL queries both in terms of time and accuracy.

Chapter 6

Linkage-Based Profiling

In this chapter we investigate the problem of finding equivalent instances among different datasets automatically as called in this thesis linkage-profiling. The rest of this chapter is structured as follows: section 6.1 gives a general overview of the problem with examples to support the motivation behind this work. The approach is presented in section 6.2 where we describe each phase of the pipeline for linkage-profiling. To evaluate the framework we conducted two experiments as presented in section 6.3 while conclusions end this chapter in section 6.4.

6.1 Overview

The idea behind Linked Data (LD) is that datasets should be linked in order to promote interoperability and integration among large data collections on the Web [25]. In chapter 4 we presented the approach to identify automatically the topical category for RDF datasets, while in chapter 5 we introduced ABSTAT a tool which helps users understand the content of big RDF datasets. Data interlinking focuses on identifying equivalent instances by determining the similarity between their instance descriptions to represent the fact that they refer to the same real world object in a given domain.

In the previous two chapters we supported our app developer in selecting the data that are of her request. After assigning the topic to datasets she could reduce her searching space. While using ABSTAT she could select only some of the datasets that share the same topic, the ones for which the resources were described with a

richer set of properties. In her final selection, she found that there are equivalent instances describing music artists in *DBpedia* and *LinkedBrianz*. Even though these datasets describe same instances the information they provide is not the same. For her purpose, Mrs. Eamla wants to collect all the available information for each music artists. She can collect this information by applying some instance matching techniques that measure the similarity between instances descriptors.

This equivalence between two instances often is represented by using the standard `owl:sameAs` property. According to OWL semantics, if two instances are connected by the `sameAs` property, every statement including one instance can be rewritten by replacing one with the other instance [16], because in OWL the semantic of this property is: *an owl:sameAs statement indicates that two URI references actually refer to the same thing*. This means that according to W3C definition[1] these two instance share the same properties. Although there are different properties that link equivalent instances we analyze only the most used property `owl:sameAs` within the LD cloud [124].

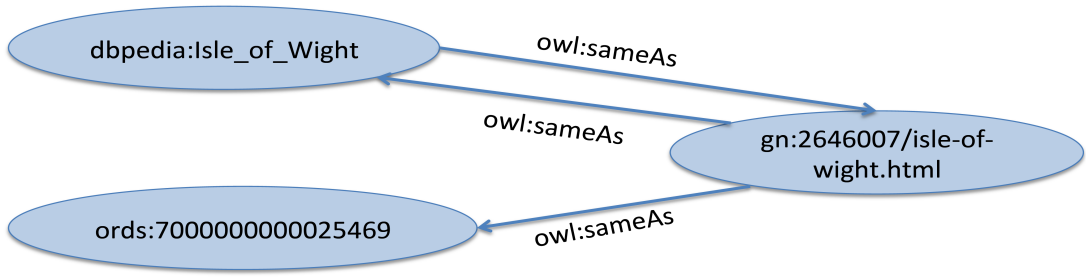


FIGURE 6.1: Example of the use of `owl:sameAs` property between three datasets

Definition 6.1. (owl:sameAs statement) An `owl:sameAs` statement is an RDF triple which connects two RDF resources by means of an owl:sameAs property.

In Fig. 6.1 it is shown an example of a piece of RDF graph which connects via the `owl:sameAs` property three instances belonging to *DBpedia*, *GeoNames* and *OrdnanceSurvey* dataset. Using this information, one can answer queries that join geographical data from *DBpedia* with other data in the *GeoNames* or *OrdnanceSurvey* dataset. Obviously, this form of linking instances from different dataset at Web-scale has enormous potential.

[1] `http://www.w3.org/TR/owl-ref/`

In the context of LD 2014 [124], we count 1 532 323 `owl:sameAs` statements. In the dump of LD, *DBpedia* is the dataset with the highest number of `owl:sameAs` triples (792 268) followed by rdfize.com (215 716), ontologycentral.com (166 020) and linked-statistics.org (144 543). A huge number of links exist between *DBpedia* and other datasets, but however, these links are trivial as *DBpedia* is build upon *Wikipedia* which by itself contain articles by collaborative authoring of its users [95]. From this first analysis we may deduce that datasets in the LD cloud are sparsely connected due to the fact that, usually, data publishers are not aware about the content of the datasets and thus the task of linking equivalent instances is not straightforward since it requires previous knowledge about the content of the datasets. For instance, in LD cloud the resource *nyt:8818483249778538299*1[2] from the *NYTimes* dataset is linked through the `owl:sameAs` property with *dbpedia:Senegal*[3]. The first resource describes Woods Hole, a place in the town of Falmouth in Barnstable County, Massachusetts, US, while the second describes Senegal, a country in Africa. These two resources have wrong sameAs links because they do not represent the same entity in the real world, thus can not be considered to be equivalent. Also in the current LD cloud *gn:2964180/gaillimh.html*[4] belonging to *GeoNames* dataset and the resource in *LinkedGeoData lgdo:node582043319*[5] are not linked with the `owl:sameAs` property even though they refer to the same city in Ireland, Galway. In order to make use of Linked Data, data consumers require them to be of a high quality and built on top applications that explore and produce trustworthy results.

The problem of finding equivalent instances among heterogeneous data sources is a well studied problem in the Semantic Web Community. This task is performed on the basis of the evaluation of the degree of similarity among descriptions of instances. Two survey papers review and summarize the approaches on data interlinking [122, 116]. The problem of discovering equivalent instances in different datasets is quite well known in record linkage [62] and ontology matching community [55, 90]. The record linkage problem tries to find all equivalent records among different databases for the same domain. [51] describes the problem of record linkage for de-dublication, data scrubbing or other forms of data scrubbing. However these techniques can not be applied for RDF data for several reasons. First of all, LD is a large source of data in difference from record linkage data warehouse

[2] ny - http://data.nytimes.com/
[3] dbpedia - http://dbpedia.org/resource/
[4] gn - http://geonames.org
[5] lgdo - http://linkegeodata.org/tiplify/

scenario. Secondly, record linkage techniques work best in domain specific settings where similarities measure can be easily customized, thus can not be applied in LD due to their heterogeneous nature. Finally, because of the loose-schema of RDF datasets it is much harder to find appropriate similarity functions.

The work presented in this chapter attempts to exploit the actual state of `owl:sameAs` links in the cloud and investigate to which extent we can automatically find other equivalent instances in the datasets without prior knowledge about their content. This will help applications built on top of LD datasets to discover more links for the same instance, thus enriching its information with other properties or information found in other datasets. To achieve this goal, we developed a framework to identify ambiguities and suggest possible inconsistencies and incompleteness. The framework implements two similarity metrics one for string similarity and one for numeric similarity, to analyze the quality of existing links and propose new ones to resolve the identified ambiguities. First, we extract all properties for instances which have an `owl:sameAs` property between two datasets in the cloud and transform them into tables. Secondly, we calculate a similarity score comparing each row between tables of datasets that we want to find equivalent instances. We consider a similarity threshold greater than 0.9 for the instances to be categorised as equivalent and test our framework on *GeoNames* which is linked with 13 other datasets in the LD cloud.

6.2 Approach

In the "Web of Data" an increasing number of `owl:sameAs` statements have been published to support merging distributed descriptions of equivalent RDF resources. Although these statements are just binary relations, when all of these `owl:sameAs` statements are taken together, they form a very large directed graph connecting RDF resources to each other [49]. As described in section 6.1 there are many cases where `owl:sameAs` statements are wrong or they are missing.

Although much effort is done during the recent years as shown in section 3.5, still not all challenges described in section 6.1 have been addressed. In contrast to [142, 103, 120], our approach implements an automated workflow which can be applied to a wide range of domains and is considered totally unsupervised. Our approach that is considered complementary to Silk and LIMES, take as input

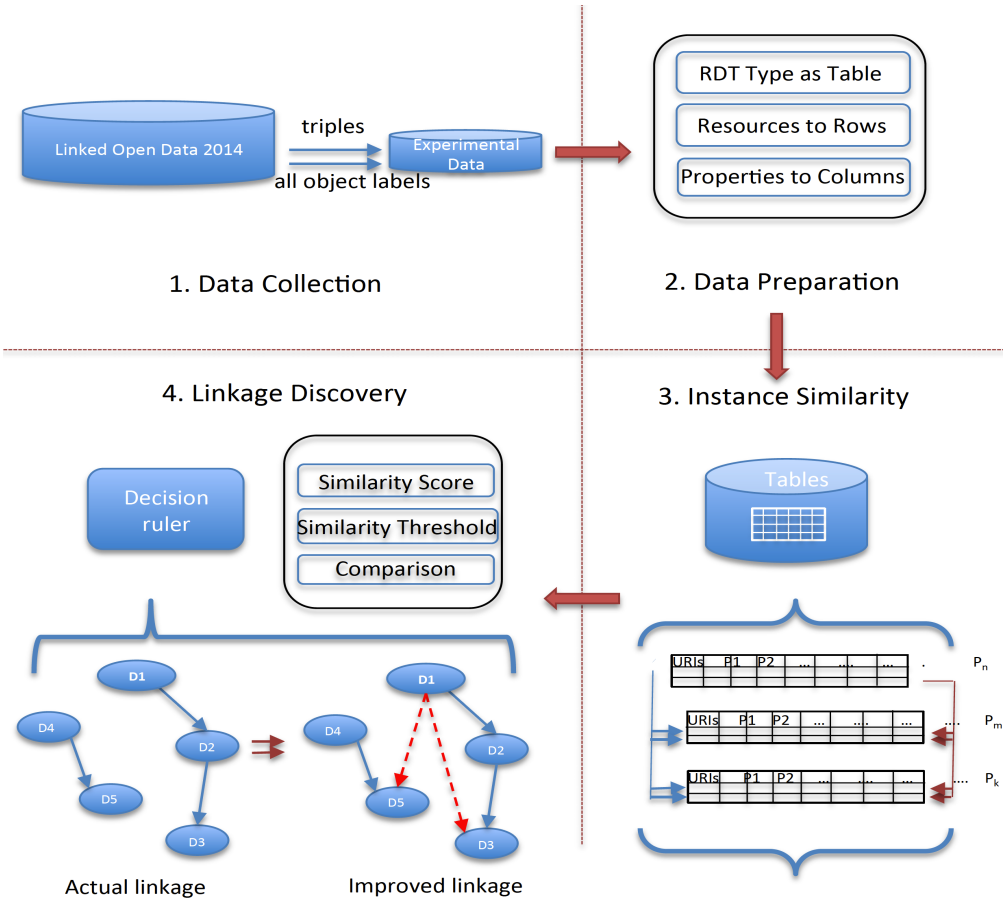


FIGURE 6.2: Pipeline for finding equivalent instances in LD

not only two datasets but one against all datasets in the LD cloud. LINDA [30] assumes each dataset to be already disambiguated while in our approach we do not make such an assumption thus addressing a more widely application. While the approach proposed in [106] is limited only for string similarity and do not cover cases when the URI contains numerical information and blank nodes, our approach covers both cases.

In this chapter we propose an approach to automatically find equivalent instances among datasets in the LD cloud as shown in Fig. 6.2.

Our approach consists of four phases: i) Data Collection; ii) Data Preparation; iii) Similarity Model; and iv) Linkage Discovery. In the following we describe each phase in detail.

6.2.1 Data Collection

As a first step toward the evaluation of existing `owl:sameAs` statements we collect triples for datasets from the LD cloud 2014. While the subject is considered to be internal to the dataset for which we want to find `owl:sameAs` statements, the object may be internal or external to this dataset. For each RDF triple belonging to a dataset that we want to find equivalent instances, we check if the object is a resource or not. In case of a resource, the description of that resource occurring at the subject position is also collected. This is done in order to have the complete information describing both resources respectively in the subject and object position linked through an `owl:sameAs` property.

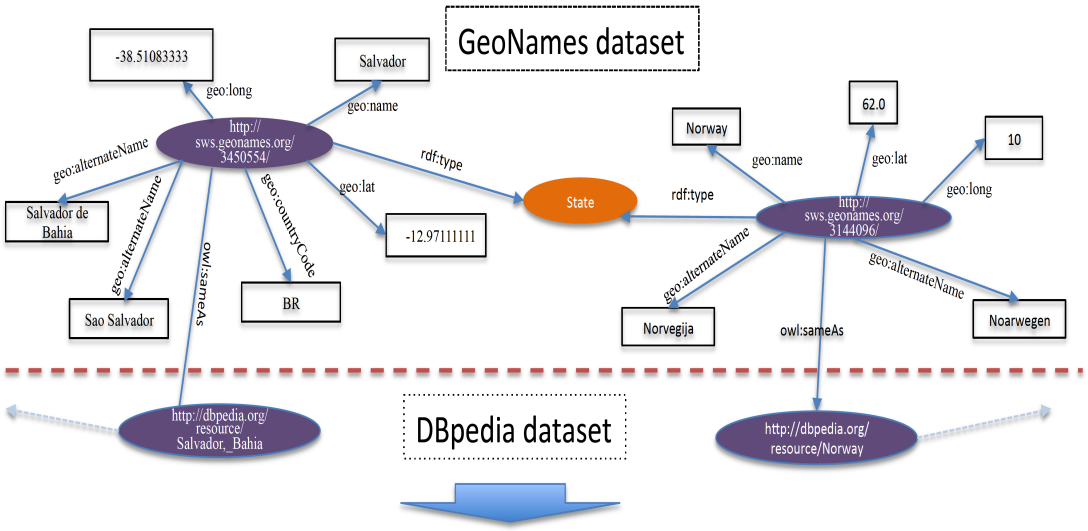
6.2.2 Data Preparation

As our aim is to automate the process of finding equivalent instances among datasets it is more easy to work with tables rather than with triples. Similarity between instances can be seen as a problem of finding similar rows between different tables of different datasets. The idea of "DBpedia as Tables"[6] inspired us to analogously transform RDF triple to tables for each dataset. Then the problem of finding equivalent instances is transformed into the problem of finding similar rows between tables, or the so-called table matching. We create one table for each class found in the dataset. Fig. 6.3 shows how to transform RDF to table. As an example, consider two instances of the class `State` from *GeoNames* dataset, named `Salvador` and `Norway` having many properties (illustrated by arrows) and their corresponding values (illustrated by squares). We transpose this information into tables where in the first row, the local name of the properties is placed, while the first column contains the URIs of the instances and the remaining cells contain the values for each property for the corresponding instance. We consider some criteria to favor an instance as a potential candidate for sameAs linkage:

Number of properties Once we built the tables we check if instances have more than four properties. If not, they are removed from the tables.

LabelLike group We create the LabelLike group which comprises the following properties: *label, name, title, text, comment, subject* and *abstract*. For each

[6] http://wiki.dbpedia.org/services-resources/downloads/dbpedia-tables



Instance URI	rdf:type	geo:alternateName	geo:name	owl:sameAs	geo:lat	geo:long	...	geo:countryCode
http://sws.geonames.org/3450554/	State	Salvador de Bahia|Sao Salvador|	Salvador	http://dbpedia.org/resource/Salvador,_Bahia	-12.97111111	-38.51083333	...	BR
http://sws.geonames.org/3144096/	State	Norvegija|Noarwegen	Norway	http://dbpedia.org/resource/Norway	62.0	10	...	null
...	...	...	...	...	...	...	...	...

FIGURE 6.3: Linked Data as Tables

instance, we check if it has at least one property belonging to the Label-Like group. If instances do not have one of those properties, they are not considered as good candiate for sameAs linkage.

We considered these two criteria because it is very difficult even for humans to find equivalent instances if they do not share at least one text value for one of the properties in LabelLike group and if the number of properties used to describe them is very low.

6.2.3 Similarity Model

All string property values are tokenized at special characters such as: /, _, :, ;, # and at capital letters. We use two formulas to calculate the similarity for properties value: String Similarity and Numeric Similarity.

6.2.3.1 String Similarity

Different similarity measures are proposed in literature [90, 126]. To find the similarity between two instaces we calculate the similarity score.

Definition 6.2. (Similarity score) Similarity score S denotes the similarity between two instances I_1 and l_2 and it is a value between 0 and 1.

The *Edit distance* formulated by Levenshtein [84] is a well-established method for finding the difference between two strings. It measures the minimum number of token insertions, deletions, and substitutions required to transform one string into another. Based on this metric we propose a lexical similarity measure. For each cell containing a string value we calculate the similarity score using the formula 6.1.

$$S_{string}(s,l) = \frac{\sum_{i=1}^{|s|} Max\{Edit(s[i], l[1..|l|])\}}{|s| + |l| - \sum_{i=1}^{|s|} Max\{Edit(s[i], l[1..|l|])\}} \tag{6.1}$$

where s and l are string sets, with s referring to the smallest set and l refering to the largest set. $S(s,l)$ gives the similarity score between sets s and l, having string value. $Edit(s[i], l[1..|l|])$ calculates the similarity between s[i], where i=0,...n and n is the number of strings in the set s to all elements in l by using Levenshtein distance metrics. Remember that one property can have more than one value. $Max\{Edit(s[i], l[1..|l|])\}$ has a value from [0,1]. In Fig. 6.4, the property *geo:alternateName* has two values. In cases when a property has more than one value the string similarity score is calculated for each of them. In this example two values of the property alternateName, from *GeoNames* dataset are, *Salvador de Bahia | Sao Salvador*. In *LinkedGeo* dataset the value of the property *label* is *Salvador*. In the above formula the smallest set s(i) is *Salvador* equal to 1, while the largest set l(l) is *Salvador de Bahia | Sao Salvador* equal to 2 (note that this property has two values). The arrows in Fig. 6.4 show the flow of the matching process.

We use Levenshtein distance ($Edit(s[i], l[1..|l|])$) to measure the similarity score between the values *Salvador de Bahia* and *Sao Salvador* from *GeoNames* and *Salvador* from *LinkedGeoData*, respectively 0.3 and 0.67. In the numerator part of the formula we select the maximum value between them, which in our example is 0.67. The denominator is equal to 2.23 (as *s=2*, *l=1* and *Max Edit = 0.67*). The similarity score between the values for the property *alternateName* and *label* is 0.3 (0.67/2.23). In the same way, we iterate through all the values of the cells. Note that we do not make an aligment between the headers of the tables when we calculate the similarity score.

6.2.3.2 Numeric Similarity

For each property having a numeric value we used the following formula to measure the similarity score:

$$S_{numeric}(n_1, n_2) = \begin{cases} 0, & if \quad |n_1 - n_2| > Min\{RanOf(n_1), RanOf(n_2)\} \\ 1 - \dfrac{|n_1 - n_2|}{Min\{RanOf(n_1), RanOf(n_2)\}}, \end{cases}$$

where n_1 is the numerical value of the property in one table, and n_2 is the numerical value of the property in the other table. The *RanOf* gives the value range of the numerical values in the column containing the property value for which we want to find the similarity score. This formula helps us to compare different numerical values ranges. As we are not aware about the properties that are being compared, sometimes the comparison is not straightforward. For instance, comparing only numerical values can be ambiguous, e.g. the coordinates with population. Suppose to find in a table a cell with the value 34.6458201 and a cell in the other table containing the value 346,458,201. Using the formula 6.2.3.2 for numeric similarity we can deduce that the similarity is 0 because n_1 - n_2 is 346458166,3541799 which is greater than the minimum value range of both columns. Therefore, these two cells can not be compared. The similarity score between –12.97111111 from *GeoNames* and –12.9816356E1 from *LinkedGeoData* using formula 6.2.3.2, is 0.998.

6.2.3.3 Aggregation

To calculate the similarity score between two instances (two different rows in tables), we consider only property values, for which the similarity score is greater than 0.9. Thus, in the example above to calculate the similarity score between the first instance of the *GeoNames* dataset and the first instance of the *LinkedGeoData* dataset, we consider only the cells with the values *Salvador*, –12.9816356E1 and –3.8482077100000005E1. Respectively for these values, the similarity score is, 1, 0.998, 0.999. To calculate the similarity score for these two instances we aggregate the similarity score of each cell weighting all values using the geometric progression of 75% increase. We use this aggregation model to reward the properties for which the similarity value is grater than 0.9. If only one cell has the similarity score

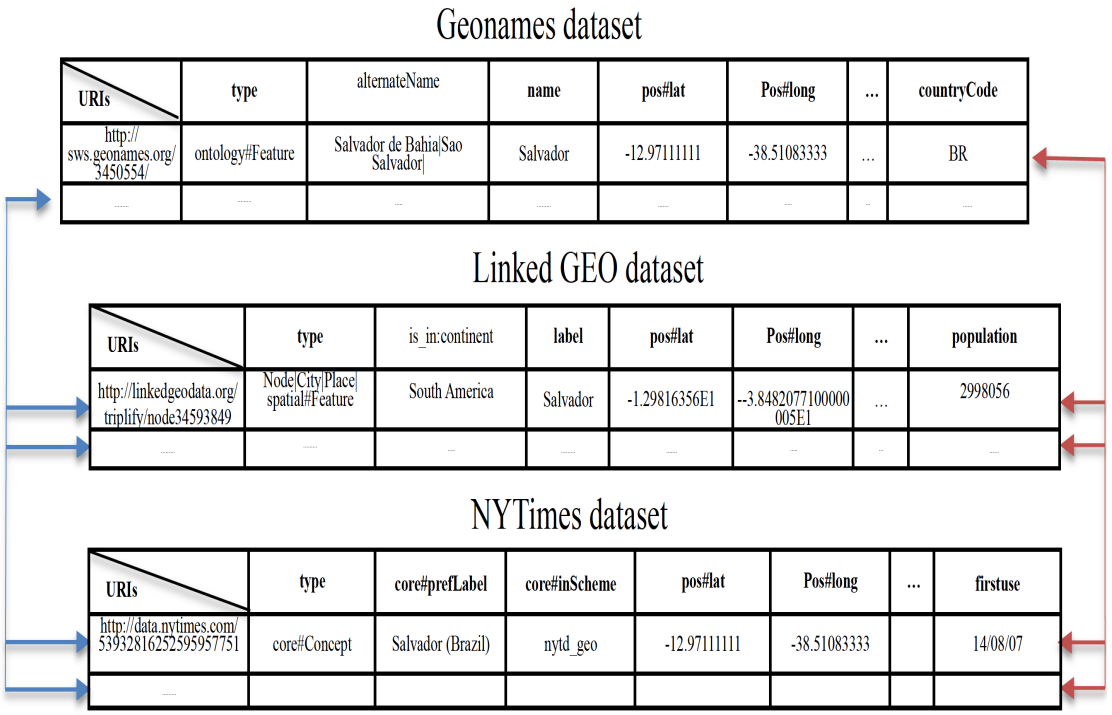


FIGURE 6.4: Instance Similarity Finding

greater than 0.9 then for the aggregation, this score is multiplied by 0.75. If two cells have similarity score greater than 0.9, then their score is multiplied by (0.75 + 0.75*0.25)/2. If three columns have similarity score greater than 0.9 their score is multiplied by (0.75 + 0.75*0.25 + 0.75*0.0625)/3. The similarity score of the first instance of the *GeoNames* dataset and the first instance of the *LinkedGeoData* dataset is equal to ((1+0.998+0.999)/3)*0.9843 = 0.9833. Note that the number of properties value with similarity score greater than 0.9 and the aggreegated score are tunnable parameters. The more properties with similarity score greater than 0.9 can contribute to the aggregated score, the more confident we are to categorise these instances as equivalent. Also, the greater the threshold for cell similarity, the more confident we are to categorise these two instance as equivalent.

6.2.4 Linkage Discovery

After calculating the similarity score for each instance, we consider as sameAs instances, those for which the aggregated similarity score is greater than 0.9. We trained different values for this threshold as shown in Fig. 6.5 and we can observe that for a threshold equal to 0.9 our approach reaches the best performance, where

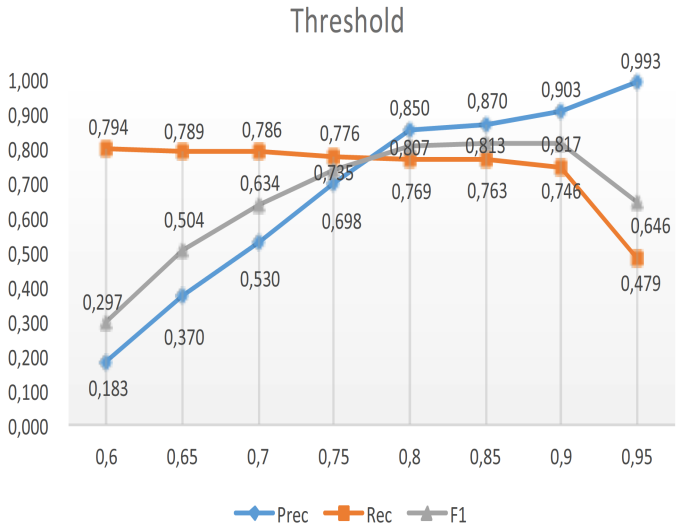


FIGURE 6.5: Framework performance tunning similarity threshold

precision has the highest value with respect to recall and f-measure. Our framework is precision oriented. In this phase of the approach, we discover sameAs statements, filtering only those instances with aggregated similarity score greater than 0.9.

6.3 Experimental Evaluation

In this section we will describe the experimental evaluation of our approach for finding equivalent instances among datasets without prior knowledge about their content.

6.3.1 Dataset and Gold Standard

Dataset. To evaluate our framework we used the datasets and the data interlinking information in LD cloud 2014. To evaluate our approach we consider *GeoNames* from the geographic domain as the dataset for which we want to find similar instances. The number of outdegree links (number of outgoing links (see section 2.2.1) aggregated for dataset) from this dataset to other datasets is 20, while the number of indegree links (number of incoming links (see section 2.2.1) aggregated for dataset) from all the other datasets to *GeoNames* is 134 [124]. In the LD cloud, *GeoNames* has 7135 `owl:sameAs` links (incoming and outgoing).

Gold Standard. We consider as Gold Standard (GS) the `owl:sameAs` links between *GeoNames* and other datasets that already exist in LD cloud[7]. As the LD cloud was manually built we can consider its information as a gold standard. The first column of the Table 6.1 shows the distribution of owl:sameAs links for the experimental datasets in the current state of the LD cloud.

6.3.2 Results

As described in Section 6.2, we initially extract 5,890 triples having an `owl:sameAs` property, where the subject is from *GeoNames* dataset and the object belongs to other datasets. For each triple we check if the object is a resource or not. In case it is a resource, we also extract the information for that resource appearing in the subject position of any triple in the cloud. We extracted this information because we want to asure that we have all the information describing the resources that we want to find the similarity score. During the *Data collection phase* of our approach we collected 587,985 triples. The second step is the *Data preparation phase* in which we check for each instance in our experimental data the number of properties it has. We do not consider those instances for which the number of properties is smaller than four and do not have a property from the LabelLike group. After applying these criteria in our experimental data we have 1,798 `owl:sameAs` that link to 610 distinct instances from *GeoNames* and 1,798 instances belonging to the target datasets. Triples are then transposed into tables as described in 6.2.2. For the *GeoNames* dataset we could have 4 tables (meaning that all sameAs links that exist in the GS (LD dump) from this dataset belong to only four classes) and for all the other datasets we could generate 133 tables (meaning that all sameAs links that exist in the others datasets belong to 133 classes). We conducted two experiments to evaluate our framework. In the first experiment we consider as target only those datasets which *GeoNames* has at least one `owl:sameAs` statement, while in the second experiment we randomly select and add in the experimental data triples from the target datasets, such that there are no *owl:sameAs* statements between them and *GeoNames*.

[7] nytimes.com, europa.eu, geovocab.org, linkedmdb.org, didactalia.net, linkedgeodata.org, lexvo.org, dbpedia.org, 270a.info, lenka.no

GeoNames with other datasets where at least one `owl:sameAs` statement exist In the first experiment we evaluate the framework for finding equivalent instances between *GeoNames* and all the other datasets where at least one `owl:sameAs` statement exists. In the Gold Standard there are 1,798 *owl:sameAs* instances between these datasets. Our framework generates 1,333 links as True Positive, 127 links as False Positive and 465 links as False Negative. In terms of precision, recall and f- measure the framework returns the following results: Precision (P) = 0.91, Recall (R) = 0.74 and f-measure (F) = 0.82.

GeoNames with other datasets where at least one or no `owl:sameAs` statement exists In the second experiment we evaluate the framework for finding equivalent instances between *GeoNames* with all the others datasets adding some noise in the experimental data. The noise consist of triples from 13 different datasets. We added triples from the datasets from the first experiment and also triples from three other datasets (ordnancesurvey.co.uk; fao.org and ucd.ie), where no `owl:sameAs` statemets exist in the Gold Standard. We add these triples to evaluate if our framework would be able to find equivalent instances between *GeoNames* dataset and the triples considered to be noise. Our framework generate 1,333 links as True Positive and 277 as False Positive. Table 6.1 shows the distribution of the links generated by the framework for each dataset. In terms of precision, recall and f- measure the framework returns the following results: Precision (P) = 0.81, Recall (R) = 0.74 and f-measure (F) = 0.77. As an observation, in the second experiment the performance of our framework decreases as a result of an increasing number of False Positive.

6.3.3 Discussion

In the following we will analyse in more detail the results from our framework, focusing in the False Positive. In order to evaluate the performance of our approach we manually check if the links generated as False Positive were correct or not. As a checking result, from 127 as False Positive from the first experiment, 99 were correct and 28 were incorrect mappings, meaning that the total number of True Positive is 1,432 and the number of False Positive is 28. Manually checking from 277 False Positive mappings from the second experiment, 206 links were correct so the number of real True Positive found by the framework is 1,539, while the number of real False Positive is 71. From this verification we prove that our framework could

TABLE 6.1: Distribution of sameAs links in Gold Standard and framework results

Dataset	GS	TP	FP	TP*	FP*
nytimes.com	497	460	4	462	2
europa.eu	719	679	97	770	6
geovocab.org	16	0	91	65	25
linkedmdb.org	11	9	0	9	0
didactalia.net	10	0	0	0	0
linkedgeodata.org	45	31	18	46	3
lexvo.org	97	18	1	19	0
dbpedia.org	227	127	23	131	19
270a.info	175	9	0	9	0
lenka.no	1	0	1	1	0
ordnancesurvey.co.uk	0	0	30	14	16
fao.org	0	0	10	10	0
ucd.ie	0	0	2	2	0
Total	1798	1264	277	1538	71

Gold Standard (GS), True Positive (TP), False Positive (FP), Verified True Positive (TP*), Verified False Positive (FP*).

find 14 similar links between *GeoNames* and ordnancesurvey.co.uk, among which there are no links in the LD cloud, thus improving the linkage information. We found that the resource e.g *gn:2110425/nauru.html* should be linked to *gv:0–170*[8] as both refer to the island of Nauru and *gn:2652355/cornwall.html* should be linked to *ords:7000000000043750*[9] as both refer to the county of Cornwell in England. This information currently is missing in the LD cloud. While if we check for two resources classified as equivalent *gn:6324733/st_john_s.html* from *GeoNames* dataset and *ords:7000000000019514* from *OrdnanceSurvey* we see that these two recources refer to different places eventhough they share the same name. In the information that we have in the cloud, these two resources share three properties for the name (LabelLike group) and one of the coordinates is similar as well. These four properties contribute to the similarity score categorising these two resources as equivalent. Another misclassification is between *gn:2618425/copenhagen.html* and *dbpedia:Copenhagen_Municipality*. Because these resources share many properties our framework classifies them as equivalent. While we observe some true classification errors, many of the mistakes made by our framework point to fact that many resources are described with similar properties so it is difficult also for humans without prior knowledge to classify them as equivalent. Our framework can be used also to check the quality of URIs in a dataset. In the dataset *OrdnanceSurvey*, the resource *Isle of Wight* is described with two URIs, *ords:7000000000025469* and *ords:7000000000025195*. Also in *LinkedGeoData* we find that the resource for the

[8]gv : http://gadm.geovocab.org/id/
[9]http://data.ordnancesurvey.co.uk/doc/

city of Vienna has two different URIs, *lgdo:node240034347* and *lgdo:node17328659*. We were able to find this information from the report of our approach where the same instance of a dataset were categorized as equivalent with two other instances, both belonging to the same dataset. This information can be used to check the overall quality of the URIs defined in a dataset.

6.4 Summary

In this chapter we discussed the problem of finding similar instances among datasets without a prior knowledge about their content. To do so we proposed a framework which can automatically find equivalent instances in the LD cloud without a prior knowledge about the class they belong to and the properties they share. The results show that this framework is very useful to find equivalent pairs between datasets not only in the same category but also with other datasets despite the category they belong to.

The analysis of the limitations of our framework, i.e., the cases where the equivalent instances found were wrong, point to the current information in LD, where usually instances even though describing different things, their property values are similar.

Chapter 7

Conclusions

As the interconnectivity of information systems increases and more and more information becomes available on the Web, the problem of understanding which to select is gaining a huge importance. Despite this fact, the ability of data consumers to select the right dataset for their need is not adjusting its accessibility.

With the adoption of Linked Data best practices, datasets assure that the structure and the semantics of the data are made explicit. Even though the adoption of Linked Data best practices has many advantages, the consumption is still limited as they lack descriptive metadata published along with the dataset.

Data profiling techniques extract descriptive metadata supporting users in understanding the content of a dataset. Throughout this thesis we analyzed the problem of profiling Linked Data in terms of topic-based, schema-based summarization and linkage-base profiling. The profile generated by the techniques described in this thesis is fundamental not only for data understanding problem and decision making but also for other use cases such as; ontology integration, identification of quality issues, query optimization, schema discovery, data visualization, data analytics and entity summarization. Moreover, our profiles can be applied by any user requiring minimal effort, because they can be executed in automatic for every dataset despite the domain they belong to.

We conclude this thesis by summing up the contributions and provide an outlook on further research directions for future work.

7.1 Summary of Contributions

In this section, we revisit each contribution in the field of profiling Linked Data and summarize the solution for the problems and challenges identified at the begining of this thesis.

Detailed literature review. We addressed the issue of providing a comprehensive and systematic literature review about tools and techniques used to profile Linked Data. First, we considered the challenges that LD poses with respect to previous works on profiling, in particular techniques used to profile relational datasets. The aim of this study was to create a crystallization point of the state of profiling tools and techniques that are available at the moment of writing this thesis. Even though as shown in chapter 3 there are some attempts in developing profiling frameworks that can cover different profiling tasks, the number of profiling tools is less than 10 and they cover specific profiling tasks. Additionally, we provide a comparison of these tools considering six criteria such as; availability, automation, scalability, licensing, usability and maintenance. Because there are only few tools and the techniques in this field covering a specific task, we can not consider profiling Linked Data a mature enough research field. One of the reasons for this could be the infancy of the research area. This study provides a broader context of topical, schema and linkage based profiling that will be beneficial to a wide range of applications and data consumers in order to identify which tool to use for her use case.

Automatic topic classification of LD datasets. As part of this thesis we have introduced an automatic, supervised method for classifying LD datasets. We employ supervised learning techniques as we want to exploit the existing topical annotation of the datasets in the cloud. We investigated the problem of single-topic classification and multi-topic classification. The results of our experiments indicate that features derived from the vocabularies used in a dataset are the best indicators for its single-topic classification of LD datasets, yielding an accuracy of around 82%.

The analysis of the limitations of our approach, i.e., the cases where the automatic classification deviates from the manually labeled one, points to a problem with the current LD cloud data collection: all datasets are labeled

with exactly *one* topical domain, although sometimes two or more categories would be equally appropriate. One such example are datasets which hold information for publications in the life science domain, which can be equally labeled as *publications* and *life sciences*. Thus, the LD dataset classification problem might be more suitably formulated as a *multi-label classification* problem [136]. The experiments for the multi-topic classification have shown that features derived from the vocabularies used in a dataset are the best indicators, yielding a performance with an f-measure of $F = 0.47$. A particular challenge for the classification is the heavy imbalance of the dataset, with roughly half of the data belonging to the *social networking* domain. Moreover, some datasets belonging to a specific topic such as *bbc.co.uk* belonging to the *media* category, make use of specific vocabularies such as *bbc vocabulary*. Because our learning classifier learn the model on specific vocabularies, it fails to assign the same topical category also to other datasets beloning to the same category but not using such vocabulary.

Building the gold standard for multi-topic classification of LD datasets. In chapter 4, we investigate the problem of single-topic classification and multi-topic classification. For the single-topic classification we considered as gold standard the information that we found in the cloud. We remind the reader that the topic of these datasets was manually assigned either by the data publishers or by the creator of the LD cloud. For the multi-topic classification, we lack a gold standard so that we could evaluate our approach. For this reason we build the gold standard for the multi-topic classification and made available for further research in this field.

Evaluation of schema-based summarization for LD datasets. Understanding the data from large datasets in the Linked Data cloud is a complex and a challenging task. We propose a minimalization-based summarization model to support dataset understanding and exploration. Experimental evaluation showed that our summarization framework is able to provide both *compact* and *informative* summaries for a given dataset. We showed that using ABSTAT framework the summaries are more compact than the ones generated from other models in the state-of-the-art. Moreover, the summaries provided by ABSTAT are informative becuase they not only support users gaining insights about the semantics of the underspecified properties in the ontology, but also to verify if the restrictions in the ontology are hold

in the data. The results of our preliminary user study showed that ABSTAT could support users formulating SPARQL queries both in terms of time and accuracy.

Automatic similar linkage discovery framework. For the linkage-based profiling we propose a framework which can automatically find equivalent instances in the LD cloud without a prior knowledge about the type they belong to and the properties they share. The results from the experiments show that this framework is very useful to find equivalent instances between datasets not only in the same category but also with other datasets despite the category they belong to. Moreover, this framework is also useful to identify quality issues by identifing wrong links among datasets and by identifying cases when same instances are modeled with different URIs within the same dataset. The performance of this framework returned the following results: P= 0.91, R= 0.74 and F= 0.82.

The analysis of the limitations of our framework, i.e., the cases where the equivalent instances found were wrong, point to the current information in LD, where usually instances even though describing different things, their property values are similar.

7.2 Future Directions

The work presented in this thesis opens up several directions for future research as it will be described in this section.

One profiling tool for several profiling tasks. We plan to implement the work presented in this thesis to ABSTAT tool, in order to make it available as a profiling tool which can cover different tasks.

Profiling survey. We introduced in this thesis a detailed review about the three aspects of profiling: topic, schema-based dataset summarization and linkage profiling. As profiling is the umbrella term for the process of generating descriptive metadata and statistics, then the survey should be extended in covering all profiling tasks. This will help data publishers and consumer to select the profiling tools and techniques that are relevant for their profiling task. We plan to extend the survey in order to cover all aspects of profiling

and make a comparison between different profiling tools in a user scenario application.

Topic-based profiling. In this thesis we investigated the problem of topic classification of LD datasets. The single-topic classification did not reach an accuracy of 100% because the classification is influenced by the heavy imbalance of the dataset, with roughly half of the data belonging to the *social networking* category. We plan to apply a two-stage classifier, which first tries to separate the larger category from the rest, while the second classifier then try to make a prediction for the remaining categories.

The results of the single-topic classification are satisfactory, but in most cases a dataset belong to more than one topic. For this we investigated the problem of multi-topic classification. The results of multi-topic showed that the performance was not as good as for the single-topic. When regarding the problem as a multi-label problem, the corresponding approach would be a *classifier chains*, which make a prediction for one category after the other, taking the prediction for the first category into account as features for the remaining classifications [148]. Another direction is the application of stacking, nested stacking or dependent binary methods [94].

Schema-based profiling. In this thesis we introduced ABSTAT, an ontology-based dataset summarization tool which help data consumers to understand the dataset at hand. Several are the future directions. We plan to run the user study in large scale, including more users with different background characteristics in order to analyse in details which is the target group of users for which ABSTAT is more useful. At the time of writing this thesis we are conducting the experiment in large scale, including more users from the Semantic Web community, by inviting them to take part of this survey through means of public mailing lists and social groups[1].

We plan to complement our coverage-oriented approach with relevance-oriented summarization methods based on connectivity analysis. Another interesting direction was highlighted by our user study, that is the inference of specific types for untyped instances found in the data set. We are also planning to consider the inheritance of properties to produce even more compact summaries. Finally, we envision a complete analysis of the most important dataset available in the LD cloud.

[1] https://lists.w3.org/Archives/Public/public-lod/2016Dec/0003.html

Linkage-based profiling. We presented an approach for identifying in automatic equivalent instances among dataset without a prior knowledge about their content. As a future work we plan to run the framework in the whole LD cloud, considering not only the instances connected by the `owl:sameAs` property but all the instances. We also plan to verify the True Negative links generated by the approach in order to identify quality problems between the instances already connected with the `owl:sameAs` property in the cloud. Because our approach compare the descriptors of all instances among them, in the current settings our approach does not scale well. For this reason, we plan to apply some blocking and filtering mechanisms to select potentially candidates for linking.

Glossary

AKP Abstract Knowledge Patterns. 87, 93, 94

COM Text from rdfs:comment Feature Set. 63, 64, 80

CSV Comma Separated Values. 1, 20, 34

CUri Class URIs Feature Set. 62, 67, 68, 76, 77

DCAT Data Catalogue Vocabulary. 39

DCTerms DCMI (Dublin Core Metadata Initiative) Metadata Terms. 11

DEG In and Outdegree Feature Set. 64, 68, 69

DL Description Logics. 30, 86

FOAF Friend-of-a-Friend Vocabulary. 11, 59

HTML Hypertext Markup Language. 1, 20

HTTP Hypertext Transfer Protocol. 20, 31, 33, 34

IRI Internationalized Resource Identifier. 24

KB Knowledge Base. 25, 48, 85, 86

KP Knowledge Patterns. 4, 42

LAB Text from rdfs:label Feature Set. 63, 64, 68, 71, 80

LCN Local Type Names Feature Set. 62, 63, 67–69, 71, 76, 77, 79

LD Linked Data. 1–17, 35, 45–47, 50, 51, 55–57, 59–61, 63, 64, 67, 68, 72–77, 79, 80, 84, 94, 103–107, 113, 114, 116, 117, 120–124

LDA Latent Dirichlet Allocation. 46, 47, 61

LDS Linked Data Summaries. 94

LOD Linked Open Data. 34, 35

LOV DCMI (Linked Open Vocabularies. 11, 12, 63, 64, 80

LPN Local Property Names Feature Set. 63, 67–69, 71, 76, 77

OWL Web Ontology Language. 11–13, 19–21, 28–30, 42, 47, 85, 86, 94, 104

PLD pay-level domain. 58

PUri Property URIs Feature Set. 62, 63, 67–69, 71, 76, 77

RDF Resource Description Framework. 1, 2, 4, 7, 9–11, 13, 16, 19–26, 28, 31–34, 39–41, 46, 47, 49, 58, 60, 61, 64, 74, 85, 86, 94, 100, 103–106, 108

RDFS Resource Description Framework Schema. 11, 12, 19, 21, 28–30, 38, 47, 48, 86

SKOS Simple Knowledge Organization System. 47, 94

SPARQL SPARQL Protocol and RDF Query Language. 1, 12, 16, 19–21, 31–34, 40, 41, 50, 85, 92–94, 97–99, 101

TLD Top-Level Domains Feature Set. 64, 68

URI Uniform Resource Identifiers. 20, 23, 25, 33, 34, 40, 51–53, 86, 104, 107, 108, 116, 122

URL Uniform Resource Locator. 31, 41

VOC Vocabulary Usage Feature Set. 62, 63, 67–69

VOCDES Vocabulary Description from LOV Feature Set. 63, 80

VOID Vocabulary of Interlinked Datasets. 10, 40, 41, 46, 47, 56

W3C World Wide Web Consortium. 1, 20, 21, 30, 31, 34, 39, 104

WWW World Wide Web. 1, 19

XML Extensible Markup Language. 1, 20, 23, 25, 41

Appendix A

Published Work

Parts of the work presented in this thesis have been published in the proceedings of international conferences and refereed workshops. Publications relating to this work are listed below:

International Journals

- Gianluigi Viscusi, Blerina Spahiu, Andrea Maurino, Carlo Batini: *Compliance with open government data policies: An empirical assessment of Italian local public administrations.* Information Polity (2014).

International Conferences

- Blerina Spahiu, Riccardo Porrini, Matteo Palmonari, Anisa Rula, Andrea Maurino: *ABSTAT: Ontology-Driven Linked Data Summaries with Pattern Minimalization.* In Extended Semantic Web Conference (ESWC) (Satellite Events) (2016).
- Cheng Xie, Melisachew Wudage Chekol, Blerina Spahiu, Hongming Cai: *Leveraging Structural Information in Ontology Matching.*In the 30th IEEE International Conference on Advanced Information Networking and Applications (AINA) (2016).
- Matteo Palmonari, Anisa Rula, Riccardo Porrini, Andrea Maurino, Blerina Spahiu, Vincenzo Ferme: *ABSTAT: Linked Data Summaries with ABstraction and STATistics.* In the Extended Semantic Web Conference (ESWC) (Satellite Events) (2015).

Refereed Workshops

- Blerina Spahiu, Cheng Xie, Anisa Rula, Andrea Maurino, Hongming Cai: *Profiling similarity links in Linked Open Data.* In the International Conference on Data Engineering (ICDE) Workshops (2016).
- Robert Meusel, Blerina Spahiu, Christian Bizer, Heiko Paulheim: *Towards Automatic Topical Classification of LOD Datasets.* LDOW Workshop. In the World Wide Web (WWW) (2015).
- Blerina Spahiu: *Profiling Linked (Open) Data.* DC proposal at International Semantic Web Conference (ISWC) (2015).
- Cheng Xie, Dominique Ritze, Blerina Spahiu, Hongming Cai: *Instance-based property matching in linked open data environment.* Ontology Matching at International Semantic Web Conference (ISWC) (2015).

Bibliography

[1] Abedjan, Z., Golab, L., and Naumann, F. Profiling relational data: a survey. *The VLDB JournalThe International Journal on Very Large Data Bases 24*, 4 (2015), 557–581.

[2] Aggarwal, C. C., and Zhai, C. A survey of text clustering algorithms. In *Mining Text Data*. 2012, pp. 77–128.

[3] Albaum, G. The likert scale revisited: an alternate version. *Journal of the Market Research Society 39*, 2 (1997), 331–332.

[4] Alexander, K., Cyganiak, R., Hausenblas, M., and Zhao, J. Describing linked datasets. In *LDOW* (2009).

[5] Araujo, S., Houben, G.-J., Schwabe, D., and Hidders, J. Fusion–visually exploring and eliciting relationships in linked data. In *International Semantic Web Conference* (2010), Springer, pp. 1–15.

[6] Arcan, M., Dragoni, M., and Buitelaar, P. Translating ontologies in real-world settings. In *International Semantic Web Conference* (2016), Springer, pp. 241–256.

[7] Assaf, A., Troncy, R., and Senart, A. Roomba: An extensible framework to validate and build dataset profiles. In *The 2nd International Workshop on Dataset PROFIling and fEderated Search for Linked Data (PROFILES '15) co-located with ESWC 2015, Portorož, Slovenia, May 31 - June 1, 2015.* (2015), pp. 32–46.

[8] Auer, S., Demter, J., Martin, M., and Lehmann, J. LODStats - An Extensible Framework for High-Performance Dataset Analytics. In *EKAW (2)* (2012).

[9] Auer, S., and Lehmann, J. Creating knowledge out of interlinked data. *Semantic Web 1*, 1, 2 (2010), 97–104.

[10] Auer, S., Lehmann, J., and Ngomo, A.-C. N. Introduction to linked data and its lifecycle on the web. In *Reasoning Web. Semantic Technologies for the Web of Data*. Springer, 2011, pp. 1–75.

[11] Baader, F. *The description logic handbook: Theory, implementation and applications*. Cambridge university press, 2003.

[12] Bao, J. rdf: Plainliteral: A datatype for rdf plain literals. *World Wide Web Consortium (W3C) Recommendation 27 October 2009* (2009).

[13] Basu, T., and Murthy, C. A. Effective text classification by a supervised feature selection approach. In *12th IEEE International Conference on Data Mining Workshops, ICDM Workshops, Brussels, Belgium, December 10, 2012* (2012), pp. 918–925.

[14] Batini, C., Palmonari, M., and Viscusi, G. The many faces of information and their impact on information quality. In *Proceedings of the 17th International Conference on Information Quality, IQ 2012, Paris, France, November 16-17, 2012.* (2012), pp. 212–228.

[15] Bay, S. D. Combining nearest neighbor classifiers through multiple feature subsets. In *ICML* (1998), vol. 98, Citeseer, pp. 37–45.

[16] Bechhofer, S., v. H. F. H. J. H. I. M. D. L. P.-S. P. F., and A, S. L. Owl web ontology language reference. *Tech. rep., W3C, http://www.w3.org/TR/owl-ref/, February 2004* (2004).

[17] Beckett, D. Modernising semantic web markup. *Proceedings of XML Europe 2004* (2004).

[18] Beckett, D. Rdf test cases - n-triples. *Tech. rep., W3C Recommendation, 2004* (2004).

[19] Beckett, D. Rdf/xml syntax specification (revised). *Tech. rep., World Wide Web Consortium, 2004.* (2004).

[20] Benington, J., and Moore, M. H. *Public value: Theory and practice*. Palgrave Macmillan, 2010.

[21] Berners-Lee, T. Linked data. In *https://www.w3.org/DesignIssues/LinkedData.html* (2006).

[22] Berners-Lee, T., Cailliau, R., and Groff, J. The world-wide web. *Computer Networks and ISDN Systems 25*, 4-5 (1992), 454–459.

[23] Berners-Lee, T., Cailliau, R., Luotonen, A., Nielsen, H. F., and Secret, A. The world-wide web. *Commun. ACM 37*, 8 (1994), 76–82.

[24] Berners-Lee, T., Hendler, J., Lassila, O., et al. The semantic web. *Scientific american 284*, 5 (2001), 28–37.

[25] Berners-Lee, T., Hollenbach, J., Lu, K., Presbrey, J., and Schraefel, M. Tabulator redux: Browsing and writing linked data.

[26] Bizer, C., Heath, T., and Berners-Lee, T. Linked data - the story so far. *Int. J. Semantic Web Inf. Syst. 5*, 3 (2009), 1–22.

[27] Blei, D. M., Ng, A. Y., and Jordan, M. I. Latent dirichlet allocation. *Journal of Machine Learning Research 3* (2003), 993–1022.

[28] Bleiholder, J., and Naumann, F. Data fusion. *ACM Computing Surveys (CSUR) 41*, 1 (2009), 1.

[29] Blomqvist, E., Zhang, Z., Gentile, A. L., Augenstein, I., and Ciravegna, F. Statistical knowledge patterns for characterising linked data. In *Proceedings of the 4th Workshop on Ontology and Semantic Web Patterns co-located with ISWC 2013, Sydney, Australia, October 21, 2013.*

[30] Böhm, C., de Melo, G., Naumann, F., and Weikum, G. LINDA: distributed web-of-data-scale entity matching. In *21st ACM International Conference on Information and Knowledge Management, CIKM'12, Maui, HI, USA, October 29 - November 02, 2012* (2012), pp. 2104–2108.

[31] Böhm, C., Kasneci, G., and Naumann, F. Latent topics in graph-structured data. In *21st ACM International Conference on Information and Knowledge Management, CIKM'12, Maui, HI, USA, October 29 - November 02, 2012* (2012), pp. 2663–2666.

[32] Böhm, C., Naumann, F., Abedjan, Z., Fenz, D., Grütze, T., Hefenbrock, D., Pohl, M., and Sonnabend, D. Profiling linked open

data with prolod. In *Workshops Proceedings of the 26th ICDE 2010, March 1-6, 2010, Long Beach, California, USA*.

[33] Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. A. *Classification and regression trees*. CRC press, 1984.

[34] Brickley, D., and Guha, R. V. Rdf/xml syntax specification (revised). *RDF vocabulary description language 1.0: RDF schema. Tech. rep., W3C, 2004. http://www.w3.org/TR/2004/ REC-rdf-schema-20040210/.* (2004).

[35] Broekstra, J., Kampman, A., and Van Harmelen, F. Sesame: A generic architecture for storing and querying rdf and rdf schema. In *International semantic web conference* (2002), Springer, pp. 54–68.

[36] Campinas, S., Perry, T. E., Ceccarelli, D., Delbru, R., and Tummarello, G. Introducing RDF Graph Summary with Application to Assisted SPARQL Formulation. In *DEXA* (2012).

[37] Castano, S., Ferrara, A., and Montanelli, S. Thematic exploration of linked data. In *Proceedings of the First International Workshop on Searching and Integrating New Web Data Sources - Very Large Data Search, Seattle, WA, USA, September 2, 2011* (2011), pp. 11–16.

[38] Castano, S., Ferrara, A., and Montanelli, S. Structured data clouding across multiple webs. *Inf. Syst. 37*, 4 (2012), 352–371.

[39] Chen, K., Kannan, A., Madhavan, J., and Halevy, A. Y. Exploring schema repositories with schemr. *SIGMOD Record 40*, 1 (2011), 11–16.

[40] Cheng, G., Jin, C., and Qu, Y. HIEDS: A generic and efficient approach to hierarchical dataset summarization. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016* (2016), pp. 3705–3711.

[41] Cheng, G., Tran, T., and Qu, Y. RELIN: relatedness and informativeness-based centrality for entity summarization. In *The Semantic Web - ISWC 2011 Bonn, Germany, October 23-27, 2011*, pp. 114–129.

[42] Cimiano, P. *Ontologies*. Springer, 2006.

[43] Clark, K. G., F. L., and Torres, E. Sparql protocol for rdf. world wide web consortium, recommendation rec-rdf-sparqlprotocol-20080115.

http://www.w3.org/tr/rdf-sparql-protocol/. *REC-rdf-sparqlprotocol-20080115. http://www.w3.org/TR/rdf-sparql-protocol/* (2008).

[44] Cochran, W. G. *Sampling techniques.* John Wiley & Sons, 2007.

[45] Colazzo, D., Goasdoué, F., Manolescu, I., and Roatiş, A. RDF Analytics: Lenses over Semantic Graphs. In *WWW* (2014).

[46] Correndo, G., Penta, A., Gibbins, N., and Shadbolt, N. Statistical analysis of the owl: sameas network for aligning concepts in the linking open data cloud. In *Database and Expert Systems Applications - 23rd International Conference, DEXA 2012, Vienna, Austria, September 3-6, 2012. Proceedings, Part II* (2012), pp. 215–230.

[47] Cyganiak, R., Stenzhorn, H., Delbru, R., Decker, S., and Tummarello, G. Semantic sitemaps: Efficient and flexible access to datasets on the semantic web. In *European Semantic Web Conference* (2008), Springer, pp. 690–704.

[48] Data, C. *An introduction to database systems.* Addison-Wesley publ., 1975.

[49] Ding, L., Shinavier, J., Shangguan, Z., and McGuinness, D. L. Sameas networks and beyond: analyzing deployment status and implications of owl: sameas in linked data. In *International Semantic Web Conference* (2010), Springer, pp. 145–160.

[50] Ellefi, M. B., Bellahsene, Z., Scharffe, F., and Todorov, K. Towards semantic dataset profiling. In *Proceedings of the 1st International Workshop on Dataset PROFIling & fEderated Search for Linked Data co-located with the 11th Extended Semantic Web Conference, PROFILES@ESWC 2014, Anissaras, Crete, Greece, May 26, 2014.* (2014).

[51] Elmagarmid, A. K., Ipeirotis, P. G., and Verykios, V. S. Duplicate record detection: A survey. *IEEE Transactions on knowledge and data engineering 19*, 1 (2007), 1–16.

[52] Erkan, G., and Radev, D. R. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research 22* (2004), 457–479.

[53] Erling, O., and Mikhailov, I. Rdf support in the virtuoso dbms. In *Networked Knowledge-Networked Media.* Springer, 2009, pp. 7–24.

[54] Euzenat, J., Meilicke, C., Stuckenschmidt, H., Shvaiko, P., and Trojahn, C. Ontology alignment evaluation initiative: six years of experience. In *Journal on data semantics XV*. Springer, 2011, pp. 158–192.

[55] Euzenat, J., Shvaiko, P., et al. *Ontology matching*, vol. 18. Springer, 2007.

[56] Fetahu, B., Dietze, S., Nunes, B. P., Casanova, M. A., Taibi, D., and Nejdl, W. A scalable approach for efficiently generating structured dataset topic profiles. In *The Semantic Web: Trends and Challenges - 11th International Conference, ESWC 2014, Anissaras, Crete, Greece, May 25-29, 2014. Proceedings* (2014), pp. 519–534.

[57] Fielding, R., Irvine, U., Gettys, J., Compaq/W3C, Mogul, J., Compaq, Frystyk, H., W3C/MIT, Masinter, L., Xerox, Leach, P., Microsoft, Berners-Lee, T., and W3C/MIT. Hypertext transfer protocol – http/1.1. *Request For Comments* (7 July 2006).

[58] Fluit, C., Sabou, M., and Van Harmelen, F. Supporting user tasks through visualisation of light-weight ontologies. In *Handbook on ontologies*. Springer, 2004, pp. 415–432.

[59] Forchhammer, B., Jentzsch, A., and Naumann, F. Lodop-multi-query optimization for linked data profiling queries. In *PROFILES@ ESWC* (2014).

[60] Gangemi, A., and Presutti, V. Towards a pattern science for the semantic web. *Semantic Web 1*, 1-2 (2010), 61–68.

[61] Gottron, T., Knauf, M., Scherp, A., and Schaible, J. ELLIS: interactive exploration of linked data on the level of induced schema patterns. In *Proceedings of the 2nd International Workshop on Summarizing and Presenting Entities and Ontologies (SumPre 2016) co-located with the 13th Extended Semantic Web Conference (ESWC 2016), Anissaras, Greece, May 30, 2016.* (2016).

[62] Gu, L., Baxter, R., Vickers, D., and Rainsford, C. Record linkage: Current practice and future directions. *CSIRO Mathematical and Information Sciences Technical Report 3* (2003), 83.

[63] Gunaratna, K., Lalithsena, S., and Sheth, A. Alignment and dataset identification of linked data in semantic web. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 4*, 2 (2014), 139–151.

[64] Halb, W., and Hausenblas, M. select * where { : I : trust : you }. In *Proceedings of the International Workshop on Interacting with Multimedia Content in the Social Semantic Web (IMC-SSW'08) Koblenz, Germany, December 3, 2008.* (2008).

[65] Heath, T., and Bizer, C. Linked data: Evolving the web into a global data space. *Synthesis lectures on the semantic web: theory and technology 1*, 1 (2011), 1–136.

[66] Hofmann, T. Probabilistic latent semantic analysis. *CoRR abs/1301.6705* (2013).

[67] Hu, W., Chen, J., Zhang, H., and Qu, Y. How matchable are four thousand ontologies on the semantic web. In *The Semantic Web: Research and Applications, ESWC 2011, Heraklion, Crete, Greece, May 29-June 2, 2011* (2011), pp. 290–304.

[68] Idreos, S., Papaemmanouil, O., and Chaudhuri, S. Overview of data exploration techniques. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data* (2015), ACM, pp. 277–281.

[69] Isele, R., Umbrich, J., Bizer, C., and Harth, A. LDSpider: An open-source crawling framework for the web of linked data. In *Proc. ISWC '10 –Posters and Demos* (2010).

[70] Jain, P., Hitzler, P., Yeh, P. Z., Verma, K., and Sheth, A. P. Linked data is merely more data. In *AAAI Spring Symposium: linked data meets artificial intelligence* (2010), vol. 11.

[71] Jarrar, M., and Dikaiakos, M. A Query Formulation Language for the Data Web. *IEEE Trans. Knowl. Data Eng 24*, 5 (2012).

[72] Joachims, T. A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. Tech. rep., DTIC Document, 1996.

[73] Joachims, T. *Learning to classify text using support vector machines: Methods, theory and algorithms.* Kluwer Academic Publishers, 2002.

[74] Khatchadourian, S., and Consens, M. P. Explod: Summary-based exploration of interlinking and RDF usage in the linked open data cloud. In *ESWC 2010, Heraklion, Crete, Greece, May 30 - June 3, 2010* (2010), pp. 272–287.

[75] Kitchenham, B. Procedures for performing systematic reviews. *Keele, UK, Keele University 33*, 2004 (2004), 1–26.

[76] Klyne, G., and Carroll, J. J. Resource description framework (rdf): Concepts and abstract syntax. *World Wide Web Consortium, Recommendation REC-rdf-concepts-20040210* (February, 2004).

[77] Konrath, M., Gottron, T., Staab, S., and Scherp, A. SchemEX - Efficient construction of a data catalogue by stream-based indexing of linked data. *J. Web Sem. 16* (2012).

[78] Kontokostas, D., Zaveri, A., Auer, S., and Lehmann, J. Triplecheckmate: A tool for crowdsourcing the quality assessment of linked data. In *Knowledge Engineering and the Semantic Web - 4th International Conference, KESW 2013, St. Petersburg, Russia, October 7-9, 2013. Proceedings* (2013), pp. 265–272.

[79] Koudas, N., Sarawagi, S., and Srivastava, D. Record linkage: similarity measures and algorithms. In *Proceedings of the 2006 ACM SIGMOD international conference on Management of data* (2006), ACM, pp. 802–803.

[80] Lacoste-Julien, S., Palla, K., Davies, A., Kasneci, G., Graepel, T., and Ghahramani, Z. Sigma: Simple greedy matching for aligning large knowledge bases. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining* (2013), ACM, pp. 572–580.

[81] Lalithsena, S., Hitzler, P., Sheth, A. P., and Jain, P. Automatic domain identification for linked open data. In *2013 IEEE/WIC/ACM International Conferences on Web Intelligence, WI 2013, Atlanta, GA, USA, November 17-20, 2013* (2013), pp. 205–212.

[82] Langegger, A., and Wöss, W. Rdfstats - an extensible RDF statistics generator and library. In *Database and Expert Systems Applications, DEXA, International Workshops, Linz, Austria, August 31-September 4, 2009* (2009), pp. 79–83.

[83] Langegger, A., and Wöss, W. RDFStats - An Extensible RDF Statistics Generator and Library. In *DEXA* (2009).

[84] Levenshtein, V. I. Binary codes capable of correcting deletions, insertions and reversals. In *Soviet physics doklady* (1966), vol. 10, p. 707.

[85] Li, N., Motta, E., and d'Aquin, M. Ontology summarization: an analysis and an evaluation.

[86] Li, W., and McCallum, A. Pachinko allocation: Dag-structured mixture models of topic correlations. In *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006* (2006), pp. 577–584.

[87] Lim, T.-S., Loh, W.-Y., and Shih, Y.-S. A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. *Machine learning 40*, 3 (2000), 203–228.

[88] Lopez, V., Unger, C., Cimiano, P., and Motta, E. Evaluating question answering over linked data. *Web Semantics: Science, Services and Agents on the World Wide Web 21* (2013), 3–13.

[89] Luaces, O., Díez, J., Barranquero, J., del Coz, J. J., and Bahamonde, A. Binary relevance efficacy for multilabel classification. *Progress in Artificial Intelligence 1*, 4 (2012), 303–313.

[90] Maedche, A., and Staab, S. Measuring similarity between ontologies. In *International Conference on Knowledge Engineering and Knowledge Management* (2002), Springer, pp. 251–263.

[91] Mäkelä, E. Aether–generating and viewing extended void statistical descriptions of rdf datasets. In *European Semantic Web Conference* (2014), Springer, pp. 429–433.

[92] Mihindukulasooriya, N., Poveda Villalon, M., Garcia-Castro, R., and Gomez-Perez, A. Loupe - An Online Tool for Inspecting Datasets in the Linked Data Cloud. In *ISWC Posters & Demonstrations* (2015).

[93] Millard, I. C., Glaser, H., Salvadores, M., and Shadbolt, N. Consuming multiple linked data sources: Challenges and experiences. In *Proceedings of the First International Conference on Consuming Linked Data-Volume 665* (2010), CEUR-WS. org, pp. 37–48.

[94] Montañes, E., Senge, R., Barranquero, J., Quevedo, J. R., del Coz, J. J., and Hüllermeier, E. Dependent binary relevance models for multi-label classification. *Pattern Recognition 47*, 3 (2014), 1494–1508.

[95] Morsey, M., Lehmann, J., Auer, S., Stadler, C., and Hellmann, S. Dbpedia and the live extraction of structured data from wikipedia. *Program 46*, 2 (2012), 157–181.

[96] Motik, B., Patel-Schneider, P. F., Parsia, B., Bock, C., Fokoue, A., Haase, P., Hoekstra, R., Horrocks, I., Ruttenberg, A., Sattler, U., et al. Owl 2 web ontology language: Structural specification and functional-style syntax. *W3C recommendation 27*, 65 (2009), 159.

[97] Nam, J., Kim, J., Loza Mencía, E., Gurevych, I., and Fürnkranz, J. Large-scale multi-label text classification - revisiting neural networks. In *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2014, Nancy, France, September 15-19, 2014. Proceedings, Part II* (2014), pp. 437–452.

[98] Naumann, F. Data profiling revisited. *SIGMOD Record 42*, 4 (2013), 40–49.

[99] Navlakha, S., Rastogi, R., and Shrivastava, N. Graph summarization with bounded error. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data* (2008), ACM, pp. 419–432.

[100] Necula, S.-C. Using resource description format in integrating and querying web data sources for semantic portals. In *Research Conference on Metadata and Semantic Research* (2011), Springer, pp. 194–200.

[101] Nenkova, A., and McKeown, K. A survey of text summarization techniques. In *Mining text data*. Springer, 2012, pp. 43–76.

[102] Nentwig, M., Hartung, M., Ngonga Ngomo, A.-C., and Rahm, E. A survey of current link discovery frameworks. *Semantic Web*, Preprint (2015), 1–18.

[103] Ngomo, A. N., and Auer, S. LIMES - A time-efficient approach for large-scale link discovery on the web of data. In *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011* (2011), pp. 2312–2317.

[104] Nuzzolese, A. G., Gangemi, A., Presutti, V., and Ciancarini, P. Encyclopedic knowledge patterns from wikipedia links. In *The Semantic Web - ISWC 2011 Bonn, Germany, October 23-27, 2011, Proceedings, Part I* (2011), pp. 520–536.

[105] Nuzzolese, A. G., Presutti, V., Gangemi, A., Musetti, A., and Ciancarini, P. Aemoo: exploring knowledge on the web. In *Web Science 2013 (co-located with ECRC), WebSci '13, Paris, France, May 2-4, 2013* (2013), pp. 272–275.

[106] Papadakis, G., Demartini, G., Fankhauser, P., and Kärger, P. The missing links: discovering hidden same-as links among a billion of triples. In *iiWAS'2010 - The 12th International Conference on Information Integration and Web-based Applications and Services, 8-10 November 2010, Paris, France* (2010), pp. 453–460.

[107] Paulheim, H., and Bizer, C. Type Inference on Noisy RDF Data. In *ISWC* (2013).

[108] Perer, A., and Shneiderman, B. Integrating statistics and visualization: case studies of gaining clarity during exploratory data analysis. In *Proceedings of the SIGCHI conference on Human Factors in computing systems* (2008), ACM, pp. 265–274.

[109] Peroni, S., Motta, E., and d'Aquin, M. Identifying Key Concepts in an Ontology, through the Integration of Cognitive Principles with Statistical and Topological Measures. In *ASWC* (2008).

[110] Peter Haase, Jeen Broekstra, A. E., and Volz, R. A comparison of rdf query languages. *In Proceedings of the 3rd International Semantic Web Conference* (2004), 502–517.

[111] Phyu, T. N. Survey of classification techniques in data mining. In *Proceedings of the International MultiConference of Engineers and Computer Scientists* (2009), vol. 1, pp. 18–20.

[112] Polleres, A., Hogan, A., Delbru, R., and Umbrich, J. Rdfs and owl reasoning for linked data. In *Reasoning Web. Semantic Technologies for Intelligent Data Access*. Springer, 2013, pp. 91–149.

[113] Presutti, V., Aroyo, L., Adamou, A., Schopman, B. A. C., Gangemi, A., and Schreiber, G. Extracting core knowledge from linked data. In *Proceedings of the COLD 2011, Bonn, Germany, October 23, 2011* (2011).

[114] Quinlan, J. R. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[115] Quinlan, J. R. *C4. 5: programs for machine learning*. Elsevier, 2014.

[116] Rajabi, E., Sicilia, M.-A., and Sanchez-Alonso, S. An empirical study on the evaluation of interlinking tools on the web of data. *Journal of Information Science 40*, 5 (2014), 637–648.

[117] Read, J., Pfahringer, B., Holmes, G., and Frank, E. Classifier chains for multi-label classification. *Machine learning 85*, 3 (2011), 333–359.

[118] Rish, I. An empirical study of the naive bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence* (2001), vol. 3, IBM New York, pp. 41–46.

[119] Röder, M., Ngomo, A.-C. N., Ermilov, I., and Both, A. Detecting similar linked datasets using topic modelling. In *International Semantic Web Conference* (2016), Springer, pp. 3–19.

[120] Ruckhaus, E., Vidal, M., Castillo, S., Burguillos, O., and Baldizan, O. Analyzing linked data quality with liquate. In *The Semantic Web: ESWC 2014 Satellite Events - ESWC 2014 Satellite Events, Anissaras, Crete, Greece, May 25-29, 2014, Revised Selected Papers* (2014), pp. 488–493.

[121] Saint-Paul, R., Raschia, G., and Mouaddib, N. General purpose database summarization. In *Proceedings of the 31st international conference on Very large data bases* (2005), VLDB Endowment, pp. 733–744.

[122] Scharffe, F., Ferrara, A., and Nikolov, A. Data linking for the semantic web. *International Journal on Semantic Web and Information Systems 7*, 3 (2011), 46–76.

[123] Scheuermann, A., Motta, E., Mulholland, P., Gangemi, A., and Presutti, V. An empirical perspective on representing time. In *Proceedings*

of the seventh international conference on Knowledge capture (2013), ACM, pp. 89–96.

[124] Schmachtenberg, M., Bizer, C., and Paulheim, H. Adoption of the linked data best practices in different topical domains. In *The Semantic Web - ISWC 2014, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part I* (2014), pp. 245–260.

[125] Shivane, P., and Rajani, R. A survey on effective quality enhancement of text clustering & classification using metadata.

[126] Shvaiko, P., and Euzenat, J. A survey of schema-based matching approaches. In *Journal on data semantics IV*. Springer, 2005, pp. 146–171.

[127] Song, G., Ye, Y., Du, X., Huang, X., and Bie, S. Short text classification: A survey. *Journal of Multimedia 9*, 5 (2014), 635–643.

[128] Staab, S., and Studer, R. *Handbook on ontologies*. Springer Science & Business Media, 2010.

[129] Suchanek, F. M., Abiteboul, S., and Senellart, P. Paris: Probabilistic alignment of relations, instances, and schema. *Proceedings of the VLDB Endowment 5*, 3 (2011), 157–168.

[130] ten Cate, B., Kolaitis, P. G., and Tan, W. C. Schema mappings and data examples. In *Joint 2013 EDBT/ICDT Conferences, EDBT '13 Proceedings, Genoa, Italy, March 18-22, 2013* (2013), pp. 777–780.

[131] Thalhammer, A., Toma, I., Roa-Valverde, A. J., and Fensel, D. Leveraging usage data for linked data movie entity summarization. *CoRR abs/1204.2718* (2012).

[132] Thellmann, K., Galkin, M., Orlandi, F., and Auer, S. Linkdaviz–automatic binding of linked data to visualizations. In *International Semantic Web Conference* (2015), Springer, pp. 147–162.

[133] Tian, Y., Hankins, R. A., and Patel, J. M. Efficient aggregation for graph summarization. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data* (2008), ACM, pp. 567–580.

[134] Tonon, A., Catasta, M., Demartini, G., and Cudré-Mauroux, P. Fixing the domain and range of properties in linked data by context disambiguation, 2015.

[135] Troullinou, G., Kondylakis, H., Daskalaki, E., and Plexousakis, D. RDF Digest: Efficient Summarization of RDF/S KBs. In *ESWC* (2015).

[136] Tsoumakas, G., and Katakis, I. Multi-label classification: An overview. *Dept. of Informatics, Aristotle University of Thessaloniki, Greece* (2006).

[137] Tsoumakas, G., Katakis, I., and Vlahavas, I. Mining multi-label data. In *Data mining and knowledge discovery handbook*. Springer, 2009, pp. 667–685.

[138] Tsoumakas, G., Katakis, I., and Vlahavas, I. Data mining and knowledge discovery handbook. *Mining multi-label data* (2010).

[139] Unger, C., Forascu, C., Lopez, V., Ngomo, A. N., Cabrio, E., Cimiano, P., and Walter, S. Question Answering over Linked Data (QALD-4). In *CLEF* (2014).

[140] Vandenbussche, P.-Y., Atemezing, G. A., Poveda-Villalón, M., and Vatant, B. Linked open vocabularies (lov): a gateway to reusable semantic vocabularies on the web. *Semantic Web*, Preprint (2015), 1–16.

[141] Völker, J., and Niepert, M. Statistical schema induction. In *The Semantic Web: Research and Applications*. Springer, 2011, pp. 124–138.

[142] Volz, J., Bizer, C., Gaedke, M., and Kobilarov, G. Silk - A link discovery framework for the web of data. In *Proceedings of the WWW2009 Workshop on Linked Data on the Web, LDOW 2009, Madrid, Spain, April 20, 2009.* (2009).

[143] Xiang, Z., Zheng, J., Lin, Y., and He, Y. Ontorat: automatic generation of new ontology terms, annotations, and axioms based on ontology design patterns. *Journal of biomedical semantics 6*, 1 (2015), 1.

[144] Yang, X., Procopiuc, C. M., and Srivastava, D. Summarizing relational databases. *Proceedings of the VLDB Endowment 2*, 1 (2009), 634–645.

[145] Zaveri, A., Rula, A., Maurino, A., Pietrobon, R., Lehmann, J., and Auer, S. Quality assessment for linked data: A survey. *Semantic Web 7*, 1 (2015), 63–93.

[146] Zhang, H. The optimality of naive bayes. *AA 1*, 2 (2004), 3.

[147] Zhang, K., Wang, H., Tran, D. T., and Yu, Y. Zoomrdf: semantic fisheye zooming on RDF data. In *Proceedings of the 19th International Conference on World Wide Web, WWW 2010, Raleigh, North Carolina, USA, April 26-30, 2010* (2010), pp. 1329–1332.

[148] Zhang, M.-L., and Zhou, Z.-H. A review on multi-label learning algorithms. *IEEE transactions on knowledge and data engineering 26*, 8 (2014), 1819–1837.

[149] Zhang, X., Cheng, G., and Qu, Y. Ontology summarization based on rdf sentence graph. In *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007* (2007), pp. 707–716.

[150] Zhao, L., and Ichise, R. Instance-based ontological knowledge acquisition. In *The Semantic Web: Semantics and Big Data, 10th International Conference, ESWC 2013, Montpellier, France, May 26-30, 2013. Proceedings* (2013), pp. 155–169.

[151] Zimmermann, A. Ontology recommendation for the data publishers. *for the Semantic Web, Aachen, Germany* (2010), 95.