



UNIVERSITY OF MILANO BICOCCA  
Doctorate School of Sciences  
Ph.D. Program in Computer Science  
XXVIII Cycle

**CONSTRUCTION AND MAINTENANCE OF DOMAIN SPECIFIC  
KNOWLEDGE GRAPHS FOR WEB DATA INTEGRATION**

Riccardo Porrini  
Doctoral Dissertation

**Supervisor:** Dr. Matteo Palmonari

**Tutor:** Prof. Dr. Enza Messina



To Vittorio.



# Abstract

A Knowledge Graph (KG) is a semantically organized, machine readable collection of types, entities, and relations holding between them. A KG helps in mitigating semantic heterogeneity in scenarios that require the integration of data from independent sources into a so called dataspace, realized through the establishment of mappings between the sources and the KG. Applications built on top of a dataspace provide advanced data access features to end-users based on the representation provided by the KG, obtained through the enrichment of the KG with domain specific facets. A facet is a specialized type of relation that models a salient characteristic of entities of particular domains (e.g., the vintage of wines) from an end-user perspective.

In order to enrich a KG with a salient and meaningful representation of data, domain experts in charge of maintaining the dataspace must be in possess of extensive knowledge about disparate domains (e.g., from wines to football players). From an end-user perspective, the difficulties in the definition of domain specific facets for dataspace significantly reduce the user-experience of data access features and thus the ability to fulfill the information needs of end-users. Remarkably, this problem has not been adequately studied in the literature, which mostly focuses on the enrichment of the KG with a generalist, coverage oriented, and not domain specific representation of data occurring in the dataspace.

Motivated by this challenge, this dissertation introduces automatic techniques to support domain experts in the enrichment of a KG with facets that provide a domain specific representation of data. Since facets are a specialized type of relations, the techniques proposed in this dissertation aim at extracting salient domain specific relations. The fundamental components of a dataspace, namely the KG and the mappings between sources and KG elements, are leveraged to elicitate such domain specific representation from specialized data sources of the dataspace, and to support domain experts with valuable information for the supervision of the process. Facets are extracted by leveraging already established mappings between specialized sources and the KG. After extraction, a domain specific interpretation of facets is provided by re-using relations already defined in the KG, to ensure tight integration of data. This dissertation introduces also a framework to profile the “status” of the KG, to support the supervision of domain experts in the above tasks.

Altogether, the contributions presented in this dissertation provide a set of automatic techniques to support domain experts in the evolution of the KG of a dataspace towards a domain specific, end-user oriented representation. Such techniques analyze and exploit the fundamental components of a dataspace (KG, mappings, and source data) with an effectiveness not achievable with state-of-the-art approaches, as shown by extensive evaluations conducted in both synthetic and real world scenarios.



# Contents

<b>Abstract</b>	<b>i</b>
<b>List of Figures</b>	<b>vii</b>
<b>List of Tables</b>	<b>ix</b>
<b>Acknowledgements</b>	<b>xi</b>
<b>1 Introduction</b>	<b>1</b>
1.1 An eCommerce Data Intensive Application . . . . .	3
1.2 Data Management Methodology for Dataspaces . . . . .	6
1.3 Limitations in the Support to Domain Experts . . . . .	7
1.4 Contributions and Outline . . . . .	8
<b>2 Dataspaces and Knowledge Graphs</b>	<b>13</b>
2.1 Dataspace Components . . . . .	13
2.1.1 Knowledge Graph . . . . .	13
2.1.2 Data Sources as Source Graphs . . . . .	19
2.1.3 Mappings . . . . .	21
2.2 Facets: End-user Oriented Representation . . . . .	23
2.3 Lifecycle of a Dataspace . . . . .	26
2.3.1 Example . . . . .	26
2.3.2 Model, Map and Materialize . . . . .	28
2.3.3 Role of Domain Experts in Data Management Tasks . . . . .	29
2.4 Summary . . . . .	31
<b>3 Literature Review</b>	<b>33</b>
3.1 Knowledge Graph Schema Enrichment . . . . .	33
3.1.1 Extraction from Structured Sources . . . . .	34
3.1.2 Extraction from Semi-Structured Sources . . . . .	35
3.1.3 Extraction from Unstructured Data Sources . . . . .	39
3.2 Mapping Discovery . . . . .	39
3.2.1 Mapping Structured Sources . . . . .	40
3.2.2 Mapping Semi-Structured Sources . . . . .	41
3.3 Knowledge Graph Profiling . . . . .	43
3.4 Summary . . . . .	45

## CONTENTS

---

<b>4</b>	<b>Domain Specific Facet Extraction</b>	<b>49</b>
4.1	Overview . . . . .	49
4.2	Problem Definition . . . . .	50
4.3	Approach . . . . .	52
4.3.1	Value Extraction . . . . .	53
4.3.2	Value Clustering . . . . .	54
4.4	Faceted Assertions Generation . . . . .	57
4.5	Evaluation . . . . .	58
4.5.1	Gold Standard . . . . .	59
4.5.2	Evaluation Metrics . . . . .	59
4.5.3	Experimental Results . . . . .	61
4.6	Comparison with Prior Art . . . . .	64
4.7	Summary . . . . .	65
<b>5</b>	<b>Specificity-based Interpretation of Facets</b>	<b>67</b>
5.1	Overview . . . . .	67
5.2	Problem Definition . . . . .	68
5.3	Specificity Driven Semantic Similarity . . . . .	70
5.4	Facet Annotation . . . . .	72
5.4.1	Relation Filtering . . . . .	73
5.4.2	Relation Ranking . . . . .	75
5.5	Evaluation . . . . .	79
5.5.1	Knowledge Graphs . . . . .	79
5.5.2	Gold Standards . . . . .	80
5.5.3	Algorithms . . . . .	82
5.5.4	Evaluation Metrics . . . . .	82
5.5.5	Experimental Results . . . . .	83
5.6	Comparison with Prior Art . . . . .	88
5.7	Application to Table Annotation Problems . . . . .	89
5.8	Summary . . . . .	92
<b>6</b>	<b>Knowledge Graph Profiling</b>	<b>93</b>
6.1	Overview . . . . .	93
6.2	Summarization Model . . . . .	94
6.3	Summary Extraction . . . . .	97
6.4	Evaluation . . . . .	99
6.4.1	Compactness . . . . .	99
6.4.2	Informativeness . . . . .	100



6.5 Comparison with Prior Art . . . . .	104
6.6 Summary . . . . .	105
<b>7 Conclusion</b>	<b>107</b>
7.1 Summary of the Dissertation . . . . .	107
7.2 Future Work . . . . .	109
<b>A Product Autocomplete</b>	<b>113</b>
A.1 Overview . . . . .	113
A.2 Autocomplete Interfaces in eCommerce . . . . .	113
A.3 Problem Definition . . . . .	115
A.4 COMMA . . . . .	117
A.4.1 Indexing . . . . .	117
A.4.2 Syntactic and Semantic Filtering . . . . .	118
A.4.3 Ranking and Top-k Retrieval . . . . .	120
A.5 Implementation and Deployment . . . . .	124
A.6 Evaluation . . . . .	125
A.6.1 Effectiveness . . . . .	126
A.6.2 Efficiency . . . . .	130
A.6.3 Real World Experimentation . . . . .	132
A.7 Comparison with Prior Art . . . . .	133
A.8 Summary . . . . .	135
 <b>Bibliography</b>	 <b>137</b>



# List of Figures

1.1	The TrovaPrezzi Italian Comparison Shopping Engine. . . . .	3
1.2	The dataspace of a Comparison Shopping Engine. . . . .	5
1.3	A summary of the contributions of this dissertation. . . . .	9
2.1	The schematization of a dataspace. . . . .	14
2.2	A simple KG. . . . .	16
2.3	Examples of sources that exchange data by means of different formats. . . . .	20
2.4	Mappings at different granularities. . . . .	22
2.5	A example of facets built on top of a KG. . . . .	23
2.6	Examples of generalist and domain specific representation of KG instances. . . . .	26
2.7	Bootstrapping a KG. . . . .	27
2.8	Maintaining a KG. . . . .	28
2.9	Evolution of a dataspace over time. . . . .	30
3.1	An example of extraction of the KG schema from a relational database. . . . .	35
3.2	Two infoboxes from Wikipedia (as of February 2016). . . . .	36
3.3	An example of template for persons. . . . .	37
3.4	A Web page describing books. . . . .	38
3.5	Schematization of the input and the output of a table annotation approach. . . . .	42
3.6	Overview of a generic structured data summarization approach. . . . .	44
4.1	Overview of the proposed approach to domain specific facet extraction. . . . .	53
4.2	An example of mutually exclusive source types. . . . .	56
5.1	The facet annotation problem. . . . .	70
5.2	An example that illustrates the difference between specific and generic relations. . . . .	71
5.3	The Facet Annotation process. . . . .	73
5.4	Evaluation of DRC over the DBpedia dataset. . . . .	84
5.5	The effect of the <i>specificity</i> score on the <b>dbpedia-numbers</b> dataset. . . . .	85
5.6	Evaluation of DRC over the YAGO dataset. . . . .	86
5.7	The effect of the <i>specificity</i> score over the YAGO dataset. . . . .	87
5.8	A table with an uninformative subject column. . . . .	90
5.9	A screenshot of the column annotation feature of STAN. . . . .	91
6.1	A small KG and corresponding patterns. . . . .	96

## LIST OF FIGURES

---

6.2	The summarization workflow. . . . .	98
6.3	Distribution of the number of minimal types from the domain and range extracted for not specified relations of the <b>db2014-core</b> KG. . . . .	101
A.1	Overview of the matching algorithm. . . . .	118
A.2	The type grouping strategy. . . . .	124
A.3	COMMA evaluation deployment scenario. . . . .	125
A.4	nDCG for the 3 different configurations of COMMA compared to the baseline at different rank thresholds (all query types). . . . .	127
A.5	nDCG for the 3 different configurations of COMMA compared to the baseline at different rank thresholds. . . . .	128
A.6	Normalized DCG for different configurations of COMMA with different weights at fixed rank values. . . . .	130
A.7	nDCG at different rank thresholds for COMMA using two different ranking functions. . . . .	130
A.8	Average execution times with set of offers of different size. . . . .	131
A.9	Ratio between users AS result selection and eMarketplace search engine usage. . . . .	132

# List of Tables

- 4.1 Gold Standard statistics. . . . . 60
- 4.2 Effectiveness of TLD, LC and WP metrics. . . . . 62
- 4.3 Number of groups discovered by TLD, LC, and WP for each source type, compared to the gold standard. . . . . 62
- 4.4 Discovered ranges of domain specific facet ranges for TLD, WP and LC compared to manually discovered ones. Numbers after facet ranges indicate their cardinality. . . . . 63
  
- 5.1 Knowledge Graphs' statistics. . . . . 80
- 5.2 Gold Standards' statistics. . . . . 80
- 5.3 Example of facets from the gold standards. . . . . 81
  
- 6.1 KGs and summaries statistics. . . . . 99
- 6.2 Total number of relations with unspecified domain and range in each KG. 101
- 6.3 Results of the user study. . . . . 103



# Acknowledgements

It is difficult to express in few sentences how thankful I am to people who supported me in this great, though difficult journey. Starting from my supervisor, Matteo Palmonari, whose passion and hard working inspired me throughout my PhD. Thanks also to ITIS and LIntAr lab crews here in Bicocca, and the ADVIS lab crew at UIC, with whom I worked until late at night, shared jokes, meals, coffees, beers, and enlightening discussions (sometimes in this very exact order). I am also very thankful to all my colleagues and employers at 7Pixel, with a special mention to team Nimbus and team Eagle crews, who taught me how to write software “for real” and who gave me the chance to go for a PhD. To my family and friends who supported me in these years goes my greatest appreciation and love.

To Marta goes even more.





# 1

## Introduction

The rapid growth of structured and semi-structured data available on the Web unveiled new horizons and business opportunities, and lead to the development of Data Intensive applications in a variety of domains, from eCommerce to the Open Government context [28]. A Data Intensive application (DI application) aims at fulfilling the information needs of end-users by leveraging data acquired, integrated, and maintained over time. The established data management approach for DI applications accounts to the *pay-as-you-go* integration of *data sources* into a *dataspace* [39]. *Semantic heterogeneity* of data from different sources is one of the main challenges to be faced in the maintenance of dataspace [41]. Knowledge Graphs (KGs) emerged as an effective tool to cope with this challenge [49, 56].

A KG is a semantically organized, machine readable, graph-like collection of *entities*, *types*, and *relations* holding between them [133], and can be represented using different data models, such as RDF or the Relational Model. A KG acts also as a repository of machine readable knowledge used to integrate new data into the dataspace, via the establishment of *mappings* between the sources and the KG. A KG provides a machine readable *representation* of data that is the backbone upon which the user-experience of a DI application can be enhanced by implementing advanced data access features. Such features are usually supported by the enrichment of the KG with a set of *facets*: specialized relations aimed at model the *salient* characteristics of entities from specific domains (e.g., news, actors, or wine bottles), from an end-user perspective.

The effective maintenance of the dataspace, its KG and mappings, is a key asset for DI applications. The richness and quality of the representation provided by the KG influences not only the ability of a DI application to fulfill the information needs of end-users, but also the ability to integrate new sources into the dataspace [49]. Domain

## 1. INTRODUCTION

---

experts play a crucial role in this picture because they are in charge of performing and supervising the execution of crucial data management tasks, ensuring the equate quality of data provided to end-users [134, 135]. Such tasks includes the *modeling* and the *enrichment* of the schema of the KG (i.e., types, relations and facets), and the *discovery* of mappings between the sources and the KG. In order to proper maintain the dataspace, domain experts must be equipped with effective tools that not only provide automatic support for specific data management tasks, but also provide the necessary information for supervision and validation.

Several challenges arise in the maintenance of the KG and mappings in a dataspace. One of them accounts to the definition of meaningful and domain specific facets. Enriching the schema of the KG with domain specific facets so as to support the advanced data access features demanded by DI applications is crucial, but nevertheless hard. Such task requires an understanding of the salient characteristics of instances for a large number of domains, and thus is difficult and time consuming at a large scale. The majority of state-of-the-art approaches in this area focus on the enrichment of the KG schema with types and relations extracted from the sources (see Section 3.1), while few focus on the extraction of facets [100, 110, 151, 33, 60] for data presentation. Among such approaches, none specifically tackles the problem of extracting domain specific facets. As a result, the current state-of-the-art does not provide adequate support to domain experts in the enrichment of the schema of the KG with granular, domain specific representation.

Motivated by these challenges, this dissertation studies how to enrich the schema of a KG with domain specific facets extracted from a vast amount of structured sources, providing interactive methods to domain experts in charge of maintaining a dataspace to ensure the adequate quality of the data. To the best of our knowledge, we are among the first in focusing on the enrichment of a KG with domain specific facets. We propose an approach to extract facets from structured data integrated into a dataspace by leveraging mappings established between sources and KG types, as an effective way to enforce the domain specificity of the extracted facets (see Chapter 4). We then discuss an approach to provide a domain specific interpretation of extracted facets. We propose to re-use relations specified by the KG of the dataspace, or eventually by other external KGs, so as to reconcile the end-user oriented representation of instances provided by facets to the back-end oriented representation provided by KG relations, and thus refine the integration between sources (see Chapter 5). Finally, we introduce an approach to provide an overview of the specificity of relations of a KG, which constitutes a vital information for domain experts in charge of supervising the enrichment

## 1.1 An eCommerce Data Intensive Application

The screenshot displays the TrovaPrezzi website interface. At the top, there is a search bar with the text 'Il motore di ricerca per i tuoi acquisti' and a search button. Below the search bar, the search results are displayed in a table format. The table has four columns: 'Prodotto', 'Negozio', and 'Prezzo'. The search results are filtered by 'Televisori LCD, LED e Plasma' and show 9686 results. The first three results are:

Prodotto	Negozio	Prezzo
Sansui DXP700 COMBI/DVD 7 LCD SANSUI DXP700 - CODICE SCONTO ATTIVO - maintstore.com - Sansui - DXP700 COMBI/DVD 7 LCD SANSUI Disponibile	MaintStore Scheda negozio ★★★★★ 178 opinioni	63,04 € + 10,54 € sped. Totale: 73,58 € Vai al negozio
Toshiba Canvio Basic 1TB_USATO HDTB310EK3AA_USATO - Gar.EUROPA ***SPEDIZIONE IMMEDIATA*** Prodotto Usato pari al nuovo *** Prodotto testato dal nostro laboratorio. Sono inclusi tutti gli accessori originali: HDX 2.5 1TB Toshiba USB3.0 Canvio Basics black Toshiba Canvio Basic 1TB. Capacità hard disk: 1000 GB, Inte Disponibile	cytech Scheda negozio ★★★★★ 48 opinioni	67,67 € + 14,56 € sped. Totale: 82,23 € Vai al negozio
UNITED DVT7092H TV color 7" LCD Risoluzione: 1440x468 - Contrasto 150:1 Luminosità 300 cd/mq - Connessione Cuffie Tuner Digitale Terrestre integrato Disponibile	MediaMarkt Scheda negozio ★★★★★ 91 opinioni	67,99 € + 11,99 € sped. Totale: 79,98 € Vai al negozio

Figure 1.1: The TrovaPrezzi Italian Comparison Shopping Engine.

of the KG schema with domain specific facets and their domain specific interpretation (see Chapter 6).

In the remainder of this chapter we present a DI application from the eCommerce domain that constitutes a motivating example of the research work discussed in this dissertation (Section 1.1). We then sketch the general data management methodology for dataspace (Section 1.2), discuss some of the limitations of current approaches to the management of dataspace (Section 1.3), and provide an overview of the contributions discussed in the dissertation (Section 1.4).

## 1.1 An eCommerce Data Intensive Application

A Comparison Shopping Engine (CSE) is a particular type of DI application specific for the eCommerce domain. A CSE integrates product offers from a large set of eMarketplaces into a dataspace. End-users access a CSE in order to find offers and compare them along several possible dimensions, including their price or specific technical features (e.g., the screen size of smartphones). TrovaPrezzi<sup>1</sup> is one of the most accessed Italian CSEs. Active since year 2002, at the time of writing TrovaPrezzi integrates about 12 millions offers from about 3000 eMarketplaces, and is accessed by about 13 millions of unique end-users every month.

<sup>1</sup><http://www.trovaprezzi.it>

## 1. INTRODUCTION

---

### The TrovaPrezzi Dataspace

Figure 1.1 depicts a screenshot of a web page from TrovaPrezzi, which displays offers of a specific category (i.e., Televisions). From an end-user perspective, TrovaPrezzi provides two basic data access features: *type-based* and *facet-based* browsing. Types (or categories) provide a coarse-grained representation of all the offers of the TrovaPrezzi dataspace, and allow users to rapidly recall the “family” of offers they are interested in. Conversely, facets provide a fine-grained, domain specific representation of offers, which helps users to rapidly recall offers with specific characteristics (e.g., all TVs with a 40 inches screen, from Figure 1.1). The core of the TrovaPrezzi dataspace is a KG (exemplified in Figure 1.2), which includes products (e.g., Iphone6) modeled as KG entities, product categories (e.g., MobilePhone) modeled as KG types, generalist attributes common to all the offers (e.g., price) modeled as generalist facets, and technical features (i.e., displaySize) modeled as domain specific facets.

Figure 1.2 provides a schematization of the data management process adopted by the TrovaPrezzi CSE. Data sources (i.e., eMarketplaces) exchange their data by means of semi-structured CSV and XML formats. The integration process is driven and supervised by domain experts with the aid of tools, and is performed by establishing mappings between source and KG entities, types, and relations. When a new source requests the inclusion of its data into the dataspace, it is first asked to provide a CSV or XML dump of its offers, as for instance the one depicted in Figure 1.2. Then, domain experts incrementally establish mappings from the data source to the KG. Preliminary mappings are established in order to interpret the generalist characteristics of offers, such as the title or the price (i.e., the mapping between attr\_4 column and brand in Figure 1.2). Offers are integrated in the dataspace since this preliminary mapping: users can search and find them.

The integration is initially weak, because offers have no fine-grained, domain specific representation attached, being them characterized by very few generalist facets such as price or brand. As a result, at this stage TrovaPrezzi is able to provide basic search and browse features with somehow limited user-experience, but the absence of granular, domain specific representation does not allow to go much far beyond that. Then, mappings are established in order to uniformly categorize offers using KG types. At this stage offers start to be interpreted, for instance, as mobile phones. Then, domain experts start defining facets aimed at providing a domain specific representation of offers, and the integration is possibly further refined by establishing fine-grained mappings between instances. At this final stage, offers are tightly integrated.

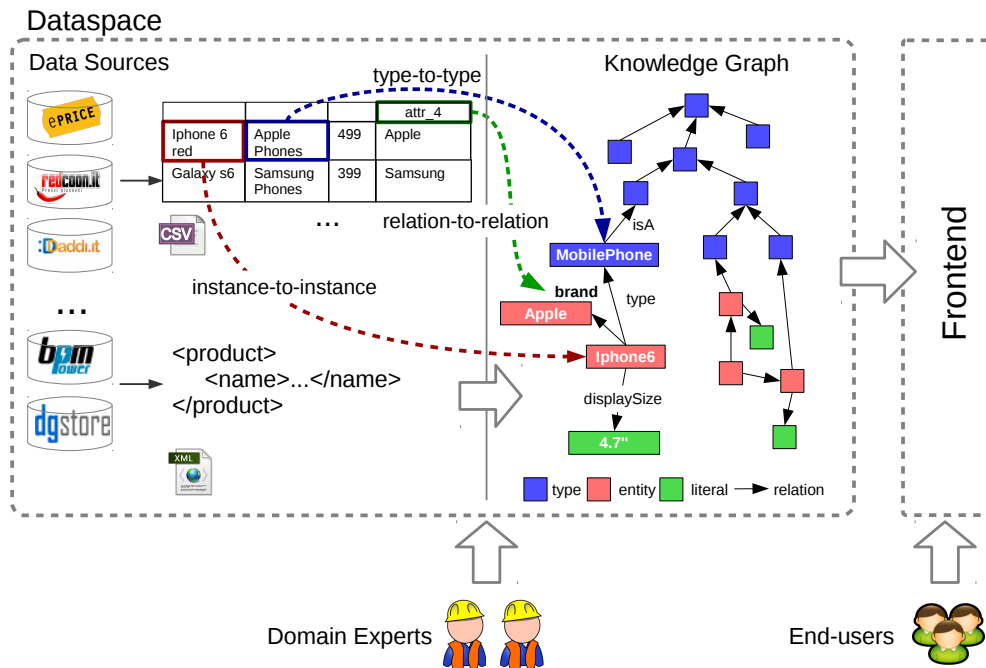


Figure 1.2: The dataspace of a Comparison Shopping Engine.

### Data Management Challenges

In the data management approach adopted by TrovaPrezzi the problem of semantic heterogeneity between sources is tackled by establishing mappings between the sources and the KG. The KG provides an end-user oriented representation of offers based on types, relations and facets. Enhanced user-experience built on top of this representation of data allows TrovaPrezzi to effectively fulfill the information needs of its end-users. Intuitively, the quality and the richness of such representation impact the user-experience of TrovaPrezzi, and consequently the business revenue originated from it.

At the time of writing, a dedicated team composed by 12 domain experts drives, supervises and validates the entire data management process described above through several tools maintained by two teams of 2 and 3 software engineers, including the author of this dissertation. The *modeling* of the schema of the KG (i.e., the definition of types, relations and facets for data presentation), and the *discovery* of mappings from the sources to the KG, are the main duties of the domain experts team, and are difficult and time demanding. The enrichment of the KG with meaningful facets at large scale requires a deep understanding of the salient characteristics of offers (e.g., wines are

## 1. INTRODUCTION

---

characterized by grape, type, provenance, and so on) for a large number of diverse domains. Also, the discovery of mappings is challenging because it requires the establishment of a huge amount of fine-grained mappings (i.e., millions of mappings, in the case of TrovaPrezzi) between specialized (i.e, domain specific) sources and the KG.

As of April 2016, about the 85% of offers integrated into the TrovaPrezzi dataspace is represented and presented to end-users only by means of generalist facets such as price or description. Efforts in providing rich and domain specific representation of offers focus on “mainstream” categories such as smartphones, tablets, or notebooks, and which account to about the 35% of product categories available in TrovaPrezzi, while more “niche” categories such as wines or ski equipment are deliberately left behind, due to the high maintenance costs. However, while mainstream categories alone contribute to a considerable portion of the revenue generated by TrovaPrezzi, niche categories constitute the “long tail” currently not fully exploited due to the lack of intelligent data management tools to support domain experts in the maintenance of domain specific representation of offers.

As a final remark, the challenges highlighted in this section apply not only to the specific case of TrovaPrezzi, but also to every more general scenario that requires the management and the integration of data from specialized data sources. For example, data released by public institutions in the Open Data and eGovernment context [88] is extremely domain specific and include, for instance, census, crime, but also pollution and public expenses related data<sup>2</sup>. In this context, the development of improved services to citizens based on such data requires a deep understanding of the underlying domain, that is difficult to achieve [50]. Remarkably, this difficulty constitutes one of the barriers in the adoption and reuse of Open Data available from public institutions [124].

### 1.2 Data Management Methodology for Dataspaces

---

Due to the large amount of heterogeneous data managed by DI applications like the CSE described in the last section, the maintenance of a dataspace (i.e., the KG and the mappings) follows an incremental *pay-as-you-go* process [77].

---

<sup>2</sup>see for instance <http://index.okfn.org/>

## 1.3 Limitations in the Support to Domain Experts

---

### Model, Map and Materialize

Three different but related data management tasks are continuously performed during the whole lifespan of a dataspace: the *modeling* of the schema of the KG, the definition of *mappings* between the sources and the KG, and *materialization* of integrated data into the dataspace. Modeling accounts to the enrichment of the KG with types, relations and facets for presenting data to end-users. A facet is a specialized type of relation that models a salient characteristic of entities of particular domains (i.e., types). The goal of the first two activities (i.e., modeling and mapping) is to enable the materialization phase, where mappings from the sources to KG types and relations defined in the modeling phase are automatically leveraged in order to transform source data and import it into the dataspace.

### Role of Domain Experts

The contribution of domain experts is fundamental in the management of the dataspace, as they are in charge of incrementally drive, supervise, and validate all the data management tasks presented in this section. Through an effective supervision, domain experts incrementally enrich the KG with a richer and domain specific representation of instances, and establish mappings between elements of such representation and the sources. Providing automatic support for domain experts in these tasks is crucial for a proper and effective management of the dataspace.

## 1.3 Limitations in the Support to Domain Experts

---

Enriching the schema of the KG to provide a rich and domain specific representation of instances requires a deep understanding of the salient characteristics of instances for a large number of diverse domains, and thus is difficult and time consuming at a large scale. Intelligent tools are needed also to support domain experts in the establishment of mappings, to ultimately ensure an high quality of the data in the dataspace. Moreover, domain experts must be equipped not only with tools that actually enrich the schema of the KG and establish mappings, but also with the necessary information to proper supervise such tools and validate their results, in order to ensure the adequate quality of data.

Many different approaches can be potentially applied to support domain experts in

## 1. INTRODUCTION

---

the management of the dataspace (see Chapter 3 for an in depth discussion). However, such approaches have several limitations, among which there are:

1. **Extraction of domain specific facets.** Despite some work focused in the enrichment of the KG schema with facets [100, 110, 151, 33, 60], to the best of our knowledge no previous work focused on extracting domain specific facets. Enriching the KG schema with such facets is challenging because requires a granular analysis of the semantics of a vast amount of heterogeneous data with approaches capable to capture domain specificity. As a result, the current state-of-the-art supports domain experts in the enrichment of the schema of the KG with types, relations, and generalist facets, but not domain specific ones.
2. **Profiling of the KG.** There have been several efforts in the state-of-the-art to profile a KG with the goal of providing domain experts with the necessary information needed to supervise the data management process of dataspace [89]. However, such approaches either include only a portion of data estimated to be more relevant in order to understand and explore the structure of the KG, or include a set of statistics without capturing patterns in the KG, which can provide useful insights to domain experts in the execution of data management tasks for dataspace maintenance. Extracting a more complete profile of a KG is challenging because it requires to find the right balance between richness and compactness. A profile should be rich enough to represent the whole KG, and compact enough to avoid redundant information so as not to overwhelm domain experts.

### 1.4 Contributions and Outline

---

Motivated by the challenges and the limitations discussed in the last sections, this dissertation investigates how to enrich the schema of a KG with domain specific facets extracted from a vast amount of structured sources, providing interactive methods to domain experts in charge of maintaining a dataspace to ensure the adequate quality of the data. With the above research question in mind, the contributions of this dissertation are schematized in Figure 1.3 and summarized in the remaining of this section.

#### **Domain Specific Facet Extraction**

##### *Chapter 4*

We study how to support domain experts in the *extraction* of domain specific facets so as to enrich the KG schema with granular, domain specific representation of data.



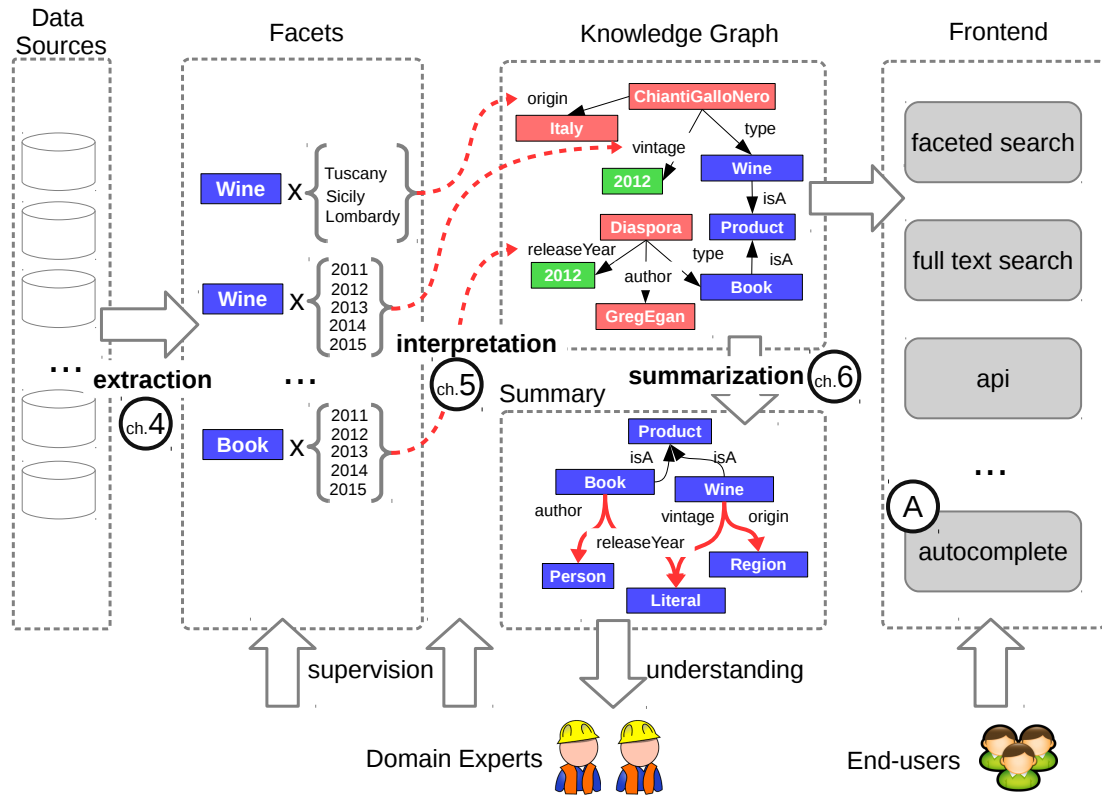


Figure 1.3: A summary of the contributions of this dissertation.

Facets are specialized relations aimed at model the salient characteristics of entities from specific domains (e.g., news, actors, or wine bottles), and thus the technical problem tackled in this contribution accounts to the extraction of salient domain specific relations. The proposed approach leverages the information already present in the dataspace, in the form of mappings established between source and KG types, to suggest meaningful facets specific for a given KG type.

### Specificity-based Interpretation of Facets

#### Chapter 5

We then focus on providing a domain specific interpretation of extracted facets, by *annotating* them with KG relations. Given a facet, the proposed approach derives a set of candidate relations from KG and ranks them considering how much their semantics is similar to the semantics of the facet, focusing on domain specificity. By re-using relations specified by the KG of the dataspace, we reconcile the end-user oriented representation of instances provided by facets to the back-end oriented representation provided by KG relations, thus refining the integration between sources.

## 1. INTRODUCTION

---

### **Knowledge Graph Profiling**

#### *Chapter 6*

We propose a KG *summarization* model that provides an abstract view of KG relations, in the form of Abstract Knowledge Patterns (e.g., (Wines, origin, Region)) and their corresponding occurrence statistics extracted from the KG. With such knowledge at hand, domain experts can properly supervise the extraction and annotation of facets and validate their result, deciding for example to annotate a facet with a relation that they consider more domain specific with respect to automatically suggested ones. The summarization model has been included in ABSTAT, a framework capable to profile a KG by providing an abstract representation of it.

### **Case Study: Product Autocomplete**

#### *Appendix A*

We describe a case study from the eCommerce domain where the rich domain specific representation is leveraged in order to provide an advanced Product Autocomplete feature for eMarketplaces. The Product Autocompletion system, which is named COMMA, is an example of the impact of the core contributions of this dissertation, deployed and evaluated on the real world scenario of an Italian eMarketplace.

### **Outline of the Dissertation**

The remainder of the dissertation is organized as follows. In Chapter 2, we provide a formal definition of a dataspace and its components: the KG, the data sources and the corresponding mappings between them. We also describe the Data Management methodology adopted in the management of a dataspace and stress the importance of supporting domain experts in the enrichment of the KG with a domain specific representation of data. In Chapter 3 we survey the state-of-the-art in management of dataspace. Chapters 4, 5, 6 present the core contributions of this dissertation right before Chapter 7, where we conclude the dissertation with a summary of the research goals achieved and contributions, and outline the potential future work and impact of the proposed techniques.

### **Note**

The work described in this dissertation has been done in collaboration with Matteo Palmonari, Brando Preda, Anisa Rula, Blerina Spahiu, Andrea Maurino, Carlo Batini

and Giuseppe Vizzari, and partially appeared in [99, 108, 107, 98, 127]. The domain specific facet extraction technique described in Chapter 4 is currently running in production within the TrovaPrezzi CSE. The facet interpretation technique described in Chapter 5 is included in STAN<sup>3</sup>, a tool for the interpretation of generic tabular data (see Section 5.7). Finally, the KG summarization framework described in Chapter 6 can be accessed online through the ABSTAT web application<sup>4</sup>.

---

<sup>3</sup><http://stan.disco.unimib.it> - <https://github.com/brando91/STAN>

<sup>4</sup><http://abstat.disco.unimib.it> - <https://github.com/rporrini/abstat>



# 2

## Dataspaces and Knowledge Graphs

### 2.1 Dataspace Components

---

We now provide a more formal definition of a dataspace. A dataspace  $\mathcal{D}$  is composed by a Knowledge Graph  $\mathcal{K}$ , a set of data sources  $\mathcal{S}$  and a set of mappings  $\mathcal{M}$  between  $\mathcal{K}$  and data sources in  $\mathcal{S}$ :

$$\mathcal{D} = \langle \mathcal{K}, \mathcal{S}, \mathcal{M} \rangle .$$

Figure 2.1 provides a schematization of a dataspace  $\mathcal{D}$ . As a remark, in this section we do not aim at providing a comprehensive formal definition of a dataspace. Instead, we focus on the concepts that are more relevant to this dissertation: the KG, the data sources and the mappings between them. For simplicity and without loss of generality, we use First Order Logic (FOL) statements to model all the components of the dataspace described in the following sections. By using FOL, we discuss this model without being bounded to any specific representation. In practice, however, different data models can be adopted to represent the KG, the sources and the mappings, such as RDFS, OWL, or the Relational Model, by using well-defined subsets of FOL (e.g., Description Logics [6] as for OWL2).

#### 2.1.1 Knowledge Graph

A Knowledge Graph (KG) is constituted by a *signature*  $\mathcal{N}$ , a set of *terminological axioms*  $\mathcal{T}$  and a set of *assertions*  $\mathcal{A}$

$$\mathcal{K} = \langle \mathcal{N}, \mathcal{T}, \mathcal{A} \rangle .$$

## 2. DATASPACES AND KNOWLEDGE GRAPHS

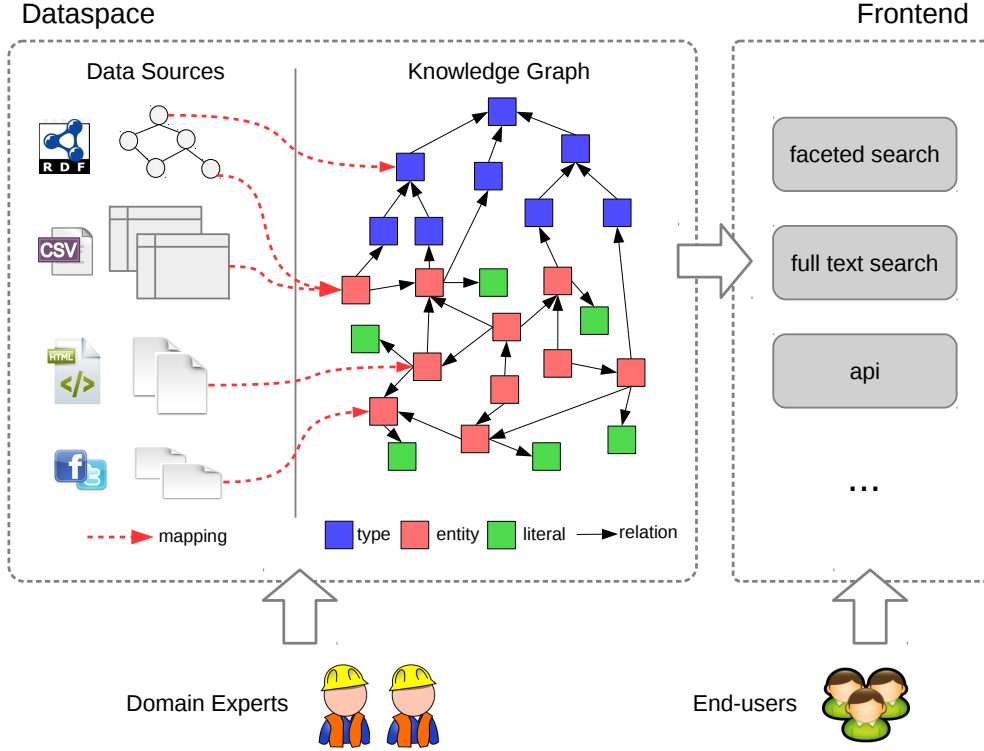


Figure 2.1: The schematization of a dataspace.

### Instances, Types and Relations

The signature  $\mathcal{N}$  of a KG defines the vocabulary of a DI application domain and is composed by a set  $\mathcal{N}^I$  of *individuals* (or constant symbols), a set  $\mathcal{N}^C$  of *unary predicate* symbols and a set  $\mathcal{N}^P$  of a *binary predicate* symbols

$$\mathcal{N} = \langle \mathcal{N}^I, \mathcal{N}^C, \mathcal{N}^P \rangle .$$

Adhering to the standard FOL formalism, given a signature  $\mathcal{N}$  an *interpretation*  $\mathcal{I}$  is a pair  $\langle \mathcal{D}, \mathcal{I} \rangle$  where  $\mathcal{D}$  is any nonempty set of objects, called the *domain* of the interpretation, and  $\mathcal{I}$  is a mapping, called the *interpretation mapping*, from the non-logical symbols (constants, predicates, and function symbols) to functions and relations over  $\mathcal{D}$  [20]. Signature elements are interpreted under the standard FOL semantics as:

- *Instances* -  $\mathcal{N}^I$ . Instances are of two different types: entities (e.g., the instance that describes the city of Chicago), and literals (e.g., "2" or "chicago"); we denote instances with constants symbols like  $a, b, \dots$

- *Types* -  $\mathcal{N}^C$ . Provide a categorization of instances, as for example a type that represents mobile phones; types are denoted with unary predicate symbols like  $C, D, \dots$
- *Relations* -  $\mathcal{N}^P$ . Conceptualize reciprocal relationships between instances, as for example the relation that holds between a mobile phone and its operating system; relations are denoted with binary predicate symbols like  $P, Q, \dots$

Instances, types and relations are associated to a set of *lexicalizations*. A lexicalization is a sequence of natural language tokens that briefly characterizes an instance  $a$ , relation  $P$  or type  $C$ , respectively. We define a special function  $lex(x)$  which tracks the correspondence between a signature element  $x$  (instance, type or relation) and its lexicalizations.

### Assertions

The set  $\mathcal{A}$  contains assertions in the form of FOL statements about instances. An assertion is a ground atomic formula built with the signature  $\mathcal{N}$ . We distinguish between two kinds of assertions: *typing* and *relational* assertions. Typing assertions are in the form

$$C(a),$$

stating that  $a$  is an instance of the type  $C$ . Similarly, a *relational* assertion in the form

$$P(a, b)$$

states that there is a relation named  $P$  which holds between a *subject* instance  $a$  and an *object* instance  $b$ . Through all this dissertation, we will denote with  $\mathcal{A}^C$  the set of all typing assertions, while we will denote with  $\mathcal{A}^P$  the set of all relational assertions.

Types and relations provide a representation of instances at different granularities. Types provide a more coarse grained representation: an instance is characterized as an instance of one or more types. For example, in the toy KG depicted in Figure 2.2 the instance `lphone6` is characterized as an instance of the type `SmartPhone` by means of the assertion

$$\text{SmartPhone}(\text{lphone6}).$$

## 2. DATASPACES AND KNOWLEDGE GRAPHS

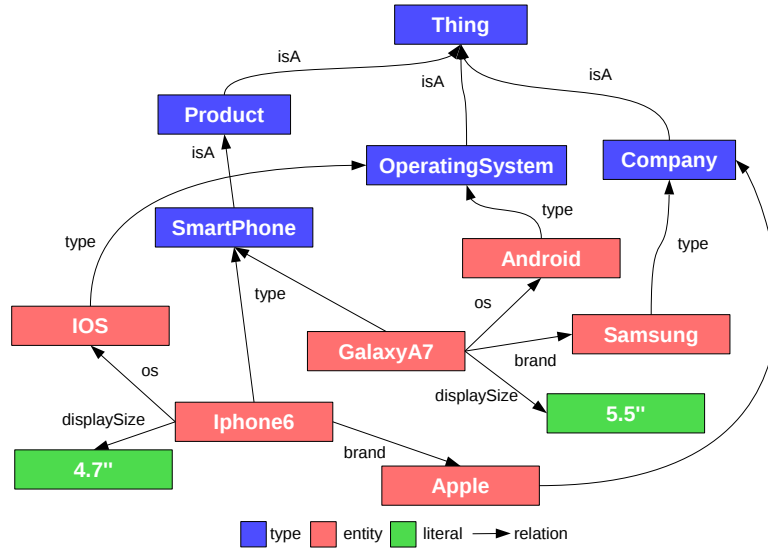


Figure 2.2: A simple KG.

Conversely, relations provide a fine-grained representation: an instance is characterized by means of its reciprocal relations with other instances. For example, the instance Iphone6 may be characterized by its brand (i.e., Apple), its operating system (i.e., ios) etc, by means of assertions such as

$\text{os}(\text{Iphone6}, \text{ios})$ .

### Terminological Axioms

The set  $\mathcal{T}$  of terminological axioms provides a specification of types and relations in the form of FOL statements about them. In practice, such terminological axioms can be specified using particular languages that constitute a fixed subsets of FOL with well defined formal properties and different expressivity, such as RDFS<sup>1</sup> or OWL<sup>2</sup>. Such specifications, along with types and relations, constitute the *schema* of the KG<sup>3</sup>.

One of the most basic, yet common, type of statements are the ones that allow to specify *subtype* relations between types. Statements such as

$$\forall x (D(x) \rightarrow C(x))$$

<sup>1</sup><https://www.w3.org/TR/rdf-schema/>

<sup>2</sup><https://www.w3.org/TR/owl-overview/>

<sup>3</sup>For KGs modeled using RDFS or OWL the schema is usually called *vocabulary* or *ontology*.



specify a subtype relation holding between the type  $D$  and the type  $C$  by stating that, whenever  $x$  is an instance of  $D$ , then  $x$  is an instance of type  $C$ . From terminological axioms about types it is possible to extract a *subtype graph*

$$G = (\mathcal{N}^C, \preceq)$$

where  $\preceq$  is a particular relation over  $\mathcal{N}^C$  introduced to represent the subtype relation between two types. For instance in Figure 2.2, the subtype hierarchy (and hence, the subtype graph  $G$ ) of types in  $\mathcal{N}^C$  can be specified by axioms

$$\begin{aligned} \forall x (\text{SmartPhone}(x) &\rightarrow \text{Product}(x)) \\ \forall x (\text{Product}(x) &\rightarrow \text{Thing}(x)) \\ \forall x (\text{Company}(x) &\rightarrow \text{Thing}(x)) \\ \forall x (\text{OperatingSystem}(x) &\rightarrow \text{Thing}(x)). \end{aligned}$$

Besides types, terminological axioms also provide a specification of KG relations in terms of their *domain* (i.e., subjects of the relation) and *range* (i.e., objects of the relation), also called domain and range *restrictions*. In principle it is possible to specify different kinds of domain and range restrictions (i.e., by stating what instances belong to the domain/range). For example, using RDFS it is possible to specify the domain of a relation  $P$  by means of the axiom

$$\forall x \forall y (P(x, y) \rightarrow C(x)),$$

stating that whenever an instance  $x$  is subject of a relation  $P$ , then  $x$  is an instance of the type  $C$ . Similarly, axioms in the form of statements like

$$\forall x \forall y (P(x, y) \rightarrow D(y))$$

state that whenever an instance  $y$  is object of a relation  $P$ , then  $y$  is an instance of the type  $D$ .

Observe that coarse-grained terminological axioms similar to the ones just described are one of the well established practices to specify a KG relation, that is by making use of the RDFS language. Other more expressive languages, such as OWL2, allow to specify more fine-grained domain or range restrictions. For example, the axiom

$$\forall x \forall y (C(x) \rightarrow (P(x, y) \rightarrow D(y)))$$

## 2. DATASPACES AND KNOWLEDGE GRAPHS

---

states that whenever  $y$  is an object of a relation  $P$  whose subject  $x$  is an instance of the type  $C$ , then  $y$  is an instance of the type  $D$ . Another example of a more fine-grained specification of a relation is the axiom

$$\forall x \forall y (P(x, y) \rightarrow C_1(x) \vee \dots \vee C_n(x))$$

which defines the domain of  $P$  as the set that includes all  $x$  that are instances of type  $C_1$  or  $C_2 \dots$  or  $C_n$ .

A more “basic” language with limited expressive power like RDFS may be easier to understand and master by domain experts. Moreover, despite the limited expressive power, RDFS still supports basic inference that allows to derive new knowledge (in the form of new assertions to be added to the KG) from existing assertions and domain and range restrictions. For example, from the following specification of a relation `os`

$$\forall x \forall y (\text{os}(x, y) \rightarrow \text{SmartPhone}(x))$$

and the relational assertion

$$\text{os}(\text{iphone6}, \text{ios})$$

it follows that the typing assertion `SmartPhone(iphone6)` holds.

The usage of “basic” languages in the specification of the types and relations has some limitations, imposed by their limited expressive power. Suppose, for example, that domain experts in charge of maintaining the KG want to reuse the relation `os` introduced to represent the operating system of smart phones, to characterize also notebooks. Limited by the expressive power of RDFS, they specify the following terminological axiom

$$\forall x \forall y (\text{os}(x, y) \rightarrow \text{Notebook}(x)).$$

From this axiom, together with the axiom and assertions provided in the above paragraph, it can be inferred that `Notebook(iphone6)`. A common practice in order to enable the reuse of relations in similar situations is to *relax* their domain specificity, by defining a new generalist type, say for example `Device`, which is a super-type of `SmartPhone` and `Notebook` and to update terminological axioms accordingly. The effect of this design choice is to make the relation less domain specific. In some cases, where enabling the reuse of relations is the main goal, this design choice ultimately leads to the definition of generalist relations by completely relaxing domain and range restrictions (also known as *underspecification* [1]). The other common practice is to use a more expressive language, such as OWL2. However, this more expressive power comes at the cost of a more difficult understanding and thus error prone maintenance of the terminological axioms by domain experts (e.g, risk of enabling undesired inferences) [47, 1].

### 2.1.2 Data Sources as Source Graphs

Data come from data sources in a variety of different data representation formats. Examples of these formats are depicted in Figure 2.3. Data sources may exchange their data by means of *structured* formats as exemplified by Figure 2.3a (e.g., a database dump or an RDF graph), *semi-structured* formats as exemplified by Figure 2.3b (e.g., CSV, XML, JSON) or *unstructured* formats exemplified by Figure 2.3c (e.g., text). Regardless of how much structured is the format for data exchange supported by a data source, a general approach is to represent the source data as a *source graph* [128, 74, 117, 16]. When not explicitly represented [128, 16] this graph structure is extracted from data using ad-hoc heuristics for semi-structured formats (e.g., quasi-relational structure assumption for tabular data [74]) or NLP techniques for unstructured formats [117].

Hence, a source  $S$  is defined in a similar way as the KG, that is constituted by a *signature*  $\tilde{N}$ , a set of *terminological axioms*  $\tilde{T}$  and a set of *assertions*  $\tilde{A}$

$$S = \langle \tilde{N}, \tilde{T}, \tilde{A} \rangle .$$

Similarly to the KG, a data source signature  $\tilde{N}$  is constituted by a set of *instances*  $\tilde{N}^I$ , *types*  $\tilde{N}^C$ , and *relations*  $\tilde{N}^P$ . Source types and relations are defined as *unary* and *binary* FOL predicates, respectively. In particular, we write  $\tilde{C}(\tilde{a})$  and  $\tilde{P}(\tilde{a}_1, \tilde{a}_2)$  to denote the classification of a source instance  $\tilde{a}$  under the type  $\tilde{C}$  (i.e., a typing assertion) and the relation  $\tilde{P}$  holding between  $\tilde{a}_1$  and  $\tilde{a}_2$  (i.e., a relational assertion), respectively.

Depending on how structured is the format, some information may or may not be explicitly exposed by the data source. Structured data exchange formats such for example RDF allow to explicitly represent and exchange the source signature  $\tilde{N}$ , the assertions in  $\tilde{A}$ , and terminological axioms  $\tilde{T}$  specified using RDFS or OWL2. In particular, RDF provides a graph-based representation of data which is explicitly conceived to support data exchange and integration over the Web [16]. In contrast, semi-structured data formats such as CSV, XML or JSON do not allow to explicitly represent and exchange terminological axioms, but only assertions. XML and JSON represent data with graph-like structures, while CSV represents data as tables. In the latter case, even if not explicitly represented, a graph-like structure can be extracted by leveraging the quasi-relational structure of a table [74]<sup>4</sup>, as exemplified in Figure 2.3b.

Source types and relations provided by a semi-structured data source are not for-

<sup>4</sup>See Section 3.2.2 for a more in-depth discussion.

## 2. DATASPACES AND KNOWLEDGE GRAPHS

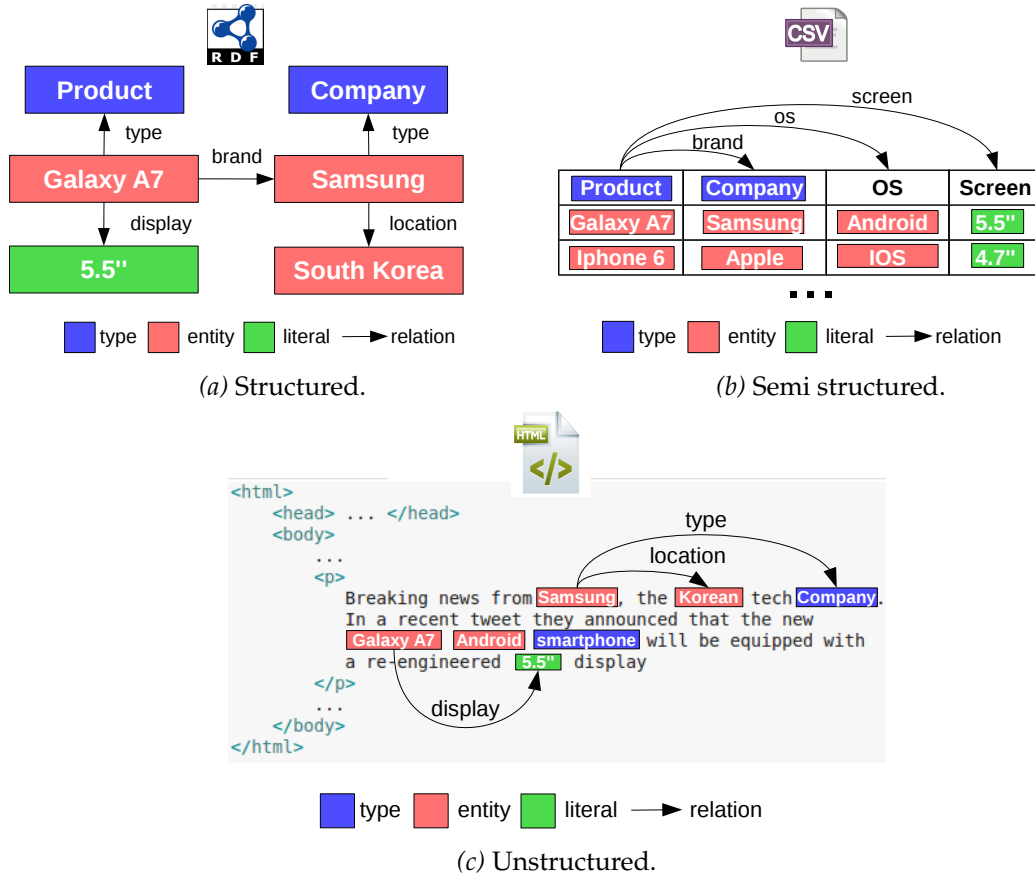


Figure 2.3: Examples of sources that exchange data by means of different formats.

mally specified, resulting in a lack of explicit machine readable semantics attached to them, in contrast with structured data sources. The lack of explicit semantics is more observable in unstructured formats, such as plain or short texts (e.g., microblog posts). In unstructured formats the signature, the terminological axioms and assertions are not explicitly represented. Remarkably, the management of data exchanged in such formats is more challenging and requires the automatic extraction of source instances, types and relations from the data, by applying Named Entity Recognition [87] and Relation Extraction [117] approaches.

In summary, a dataspace may potentially integrate data exchanged in a variety of different formats. In this setting, data *heterogeneity* and *ambiguity* are two of the main challenges that arise. Source data is heterogeneous because comes from independent sources, each one with its possibly own representation. Data is ambiguous because in many cases lacks an explicit semantics attached. Unstructured data is the most

ambiguous, because it lacks not only semantics, but also structure. In contrast, structured data represented and exchanged in RDF or more generally through Semantic Web models, languages and standards is potentially less ambiguous, because its semantics can be explicitly exchanged. Moreover, a common and established practice is to reuse the semantics provided by shared ontologies and vocabularies (e.g., Foaf<sup>5</sup>, Dublin Core<sup>6</sup>, Schema.org<sup>7</sup>), thus potentially reducing heterogeneity of data provided by different sources [16].

### 2.1.3 Mappings

A dataspace stores a set of *mappings*  $\mathcal{M}$  between KG elements (i.e., instances, types and relations) and elements from data sources. We define three types of mappings, and characterize them in terms of the type of elements between which they hold: *type-to-type* (t-to-t), *relation-to-relation* (r-to-r) and *instance-to-instance* (i-to-i). Figure 2.4 provides an example of these mappings. Observe that these three types of mappings allow to specify the integration of the source data into the KG at different granularity levels. Coarse grained integration is performed by establishing t-to-t coarse grained mappings. The integration is refined by establishing r-to-r mappings, so as to provide a richer representation of source instances. Finally, i-to-i mappings provide fine grained integration. As in traditional data integration settings [70], mappings are specified using Horn clauses. This formalism is consistent with the FOL logic-based formalism introduced to represent the KG and the sources, as Horn clauses can be represented by means of FOL formulas.

A t-to-t mapping is encoded as a Horn clause in the form:

$$C(x) \leftarrow \tilde{C}(x)$$

where  $C$  is a KG type,  $\tilde{C}$  is a source type and  $x$  is a source instance. The semantics of a t-to-t mapping is such that all source instances  $\tilde{a}$  that are classified as instances of the source type  $\tilde{C}$ , are also classified as instances of the KG type  $C$ . For example, the specification of the t-to-t mapping depicted in Figure 2.4 between the source type Product and the KG type SmartPhone can be given by means of the Horn Clause

$$\text{SmartPhone}(x) \leftarrow \text{Product}(x).$$

---

<sup>5</sup><http://www.foaf-project.org/>

<sup>6</sup><http://dublincore.org/>

<sup>7</sup><http://schema.org/>

## 2. DATASPACES AND KNOWLEDGE GRAPHS

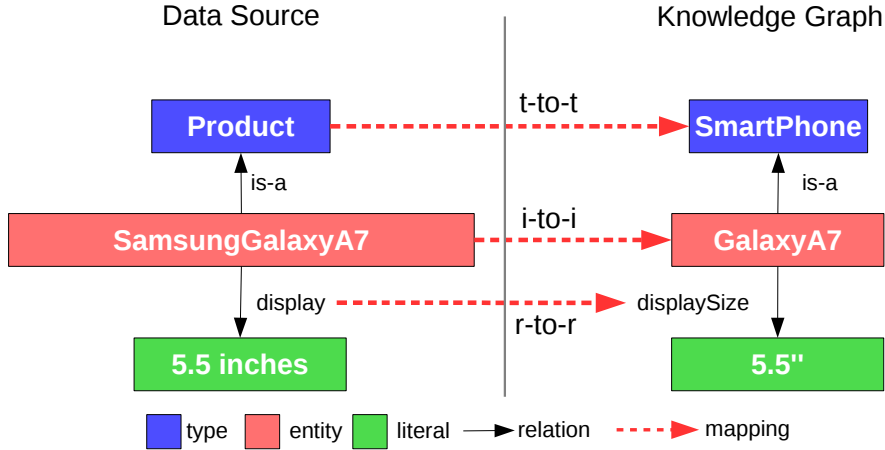


Figure 2.4: Mappings at different granularities.

Similar to t-to-t mappings, r-to-r mappings are encoded as Horn clauses in the form:

$$P(x, y) \leftarrow \tilde{P}(x, y)$$

where,  $P$  is a KG relation,  $\tilde{P}$  is a source relation and  $x$  and  $y$  are two source instances. The semantics of a r-to-r mapping is such that if a source relation  $\tilde{P}$  holds between two source instances  $x$  and  $y$ , then the KG relation  $P$  holds between  $x$  and  $y$ . For example, the specification of the r-to-r mapping depicted in Figure 2.4 between the source relation `display` and the KG relation `displaySize` can be given by means of the Horn Clause

$$\text{displaySize}(x, y) \leftarrow \text{display}(x, y).$$

Instances are mapped by i-to-i mappings, specified Horn clauses with empty head and using special relations that hold between a source and a KG instance:

$$\text{sameAs}(x, y).$$

The (rather trivial) semantics of an i-to-i mapping is such that the source instance  $x$  is equivalent to the KG instance  $y$ . For example, the specification of the i-to-i mapping depicted in Figure 2.4 between `SamsungGalaxyA7` and the KG instance `GalaxyA7` can be given by means of the Horn Clause

$$\text{sameAs}(\text{GalaxyA7}, \text{SamsungGalaxyA7}).$$

## 2.2 Facets: End-user Oriented Representation

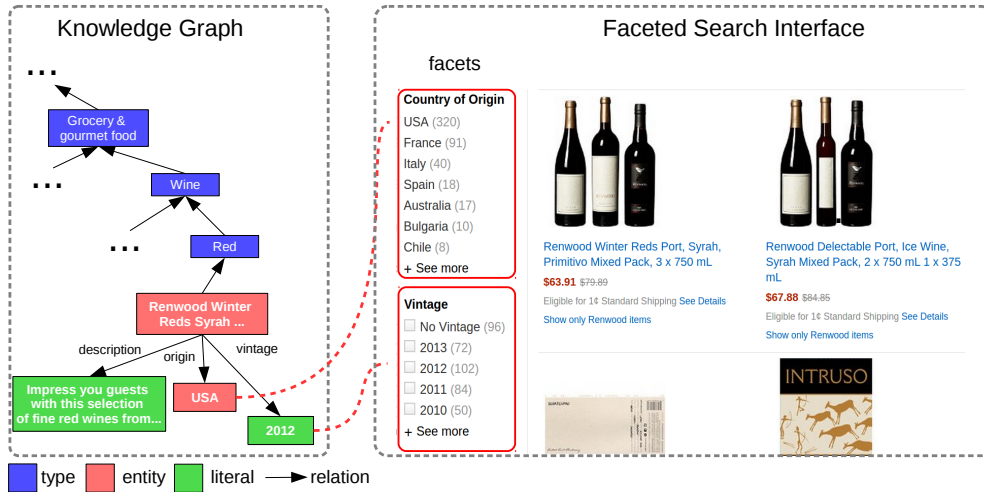


Figure 2.5: A example of facets built on top of a KG.

As a final remark, observe that in principle, the definition given in this section allows to specify also more complex mappings, such for instance the t-to-t mapping

$$\text{AndroidDevice}(x) \leftarrow \text{Product}(x), \text{operatingSystem}(x, \text{Android})$$

without compromising the formal framework provided by this Chapter. However, allowing the specification of arbitrarily complex and expressive mappings come at the cost of sacrificing their ease of maintenance, and is a rare practice in real world dataspace.

## 2.2 Facets: End-user Oriented Representation

Within a dataspace, a KG provides a semantically rich representation of instances through types and relations. This representation is “general purpose”, and supports the spectrum of data management operations performed to maintain the dataspace. Besides such back-end oriented perspective, however, the other important goal of a KG is to enable a DI application to provide end-users with enhanced access to the data of the dataspace, like one depicted in Figure 2.5. One established way to pursue this goal is through the definition of a particular type of relations, that we name *facets* [147, 31]. A facet is a relation that represents a *salient* characteristic of instances of a KG type (i.e., domain) and can be informally defined as:

## 2. DATASPACE AND KNOWLEDGE GRAPHS

---

A (1) *mutually exclusive*, and (2) *collectively exhaustive* aspect, property, or characteristic of a (3) class or *specific subject* [138].

Facets are relations but, in contrast with the general purpose ones introduced in Section 2.1.1, are meant for data presentation to end-users. For example, Figure 2.5 depicts a dataspace in which wine bottles are characterized by facets that include the country of origin and an year of production (i.e., vintage). On top of this representation a DI application can provide a faceted search interface where users can use facets to filter the results of a full text query [105, 151]. Applying the high level definition provided in the beginning of this section, a facet is defined as a relation that:

1. has many-1 cardinality (referred to “mutually exclusive”, in the above definition)<sup>8</sup>;
2. exhaustively covers the set of possible values of the relationship that represents (referred to “collectively exhaustive”);
3. is domain specific (referred to “specific subject”);

Observe that a facet is, by all means, a relation. However, a relation is not necessarily a facet. Consider, for instance, the KG depicted in Figure 2.5, which defines three different relations: description, origin and vintage. Among them, the relation vintage is a facet for representing wines, because it is domain specific, in contrast with the relation description (every product offer may have a description attached). In fact, the range of description potentially contains heterogeneous values: such facet would be composed by all the descriptions of all KG instances, and can hardly be considered a salient characteristic of wine bottles. Through the rest of this dissertation, with the term *facets* we will refer to this particular type of KG relations.

From a formal point of view, a facet holds between a fixed *domain* of instances of a specific KG type  $C$  and a *range* of facet values  $\mathcal{V} = \{v_1, \dots, v_n\}$ :

$$F = \langle C, \mathcal{V} \rangle .$$

---

<sup>8</sup>In real world, for practical reasons, the constraint on mutual exclusivity may be somehow relaxed, as for instance a facet representing actors starring in a movie. In this case, although not of many-1 cardinality, the relation represented by this facet is still a salient aspect for movies.



## 2.2 Facets: End-user Oriented Representation

---

Since  $F$  is a relation, its specification is given by a set of terminological axioms that intuitively model the properties that a relation must satisfy in order to be considered as a facet:

$$\begin{aligned} & \forall x (C(x) \rightarrow \exists y F(x, y)) \\ & \forall x (C(x) \rightarrow \forall z (F(x, y) \wedge F(x, z) \rightarrow y = z)) \\ & \exists x (C(x) \wedge F(x, v_1)) \\ & \dots \\ & \exists x (C(x) \wedge F(x, v_n)) \end{aligned}$$

where  $F(x, v)$  is a *faceted assertion*, meaning that  $v$  is asserted to be the value of the facet  $F$  that characterizes the instance  $x$ . Observe that the first two axioms specify the many-1 cardinality of the facet, while domain specificity with respect to a specific KG type  $C$  is enforced by the inclusion of typing assertions  $C(x)$  in terminological axioms.

Depending on the level of abstraction of the KG type to which they are referred, facets can be of two types: generalist or domain specific. For example, the facet seller is generalist, as it may be referred to product offers in general, while the facet vintage is specific for the domain of wines. Domain specific facets are crucial in order to provide enhanced data access to end-users [105]. By relying on domain specific facets, a DI application is able to present more interesting and meaningful information to end-users, which would be more difficult to provide by relying on generalist facets only. This difference between domain specific and generalist representations is exemplified by the two faceted search interfaces depicted in Figure 2.6. The one in the left side is based on representation given by generalist facets<sup>9</sup>, while the one at the right is based on domain specific facets<sup>10</sup>. Wine bottles on the left side are characterized by generalist facets such as seller and price, which are applicable to all the KG instances (i.e., products in this example). Conversely, in the faceted search interface on the right side, instances are characterized by domain specific facets for wines (modeled as a specific KG type), such as grapeVarietal or countryOfOrigin. It is straightforward to notice that provides a more enhanced user experience to end-users.

---

<sup>9</sup><http://bit.ly/less-characterized-wines>, accessed on February 2016

<sup>10</sup><http://bit.ly/more-characterized-wines>, accessed on February 2016

## 2. DATASPACE AND KNOWLEDGE GRAPHS

### Generalist Facets

Clear all filters

**Show only**

New items

**Price**

Up to \$45

\$45 – \$350

Over \$350

\$  to \$

**Category** - Clear

Wine

**Producer**

Cupcake Vineyards

Korbel

Moët & Chandon

Stella Rosa

Veuve Clicquot

More

**Seller**

grooves-inc.com


Houzz

Overstock.com

winelegacy.com

www.ImprintItems.co


Wine



**Copper Ridge 12-Bottle N**

**\$63.00** from winelegacy.com

Copper Ridge Cabernet Sauvignon




**Ravens Wood Vintners B**

**\$13.04** from 5+ stores

#1 in Ravenswood Wine

Ravenswood


Full, dark, spicy, fruity and - more ti



**Aromabar, Premium Editi**

**\$461.04** from 3 stores


The diversity of wine aromas is fas



**Napa Cellars Carneros S**

**\$399.00** from Houzz ★★★★★

Released: Aug '12. Barrel regimen:



**Ultimate Sparkling Wine I**

**\$149.00** from winelegacy.com

071 Raboso Moscato Sensi Dolce

### Domain Specific Facets

**Grape Varietal**

Blend - Red (137)

Nebbiolo (103)

Sangiovese (57)

Sangiovese Grosso (30)

Blend - White (23)

Pinot Grigio (17)

Barbera (16)

+ See more

**Professional Rating**

95 & Up (68)

94 & Up (122)

92 & Up (174)

90 & Up (197)

86 & Up (211)

**Country of Origin**

< Any Country of Origin

**Italy**

Tuscany (204)

Piedmont (162)

Veneto (55)

Friuli-Venezia-Giulia (28)

Sicily (24)

Campania (20)

Marche (17)

+ See more

**Avg. Customer Review**

★★★★★ & Up (29)

★★★★☆ & Up (31)

★★★☆☆ & Up (32)

★★☆☆☆ & Up (32)


**Price**

Under \$10 (6)

\$10 to \$20 (136)

\$20 to \$30 (108)

★★★★★ 3




**NV Chateau Diana Sparkling Moscato 750 mL**  
by Chateau Diana

**\$10.00**

[Show only Chateau Diana items](#)

★★★★☆ 2



**2010 Castellani Vernaccia di San Gimignano, 1 750 mL**  
by Castellani

**\$15.00**

[Show only Castellani items](#)




Figure 2.6: Examples of generalist and domain specific representation of KG instances.

## 2.3 Lifecycle of a Dataspace

The predominant data management methodology adopted for dataspace, and in particular of the management of KG and the mappings between its elements and source data, follows a *pay-as-you-go* approach based on data co-existence and supervised by domain experts with the use of (semi) automatic tools [41]. In this section we discuss the lifecycle of a dataspace that supports DI applications, with particular focus on the process of *bootstrapping* and *maintaining* the KG and the mappings over time.

### 2.3.1 Example

To better illustrate the lifecycle of a dataspace, we rely on an example from the eCommerce domain and in particular a Comparison Shopping Engine (CSE) similar to the one described in Section 1.1. In the case of a CSE, data sources consist of several eMar-

26

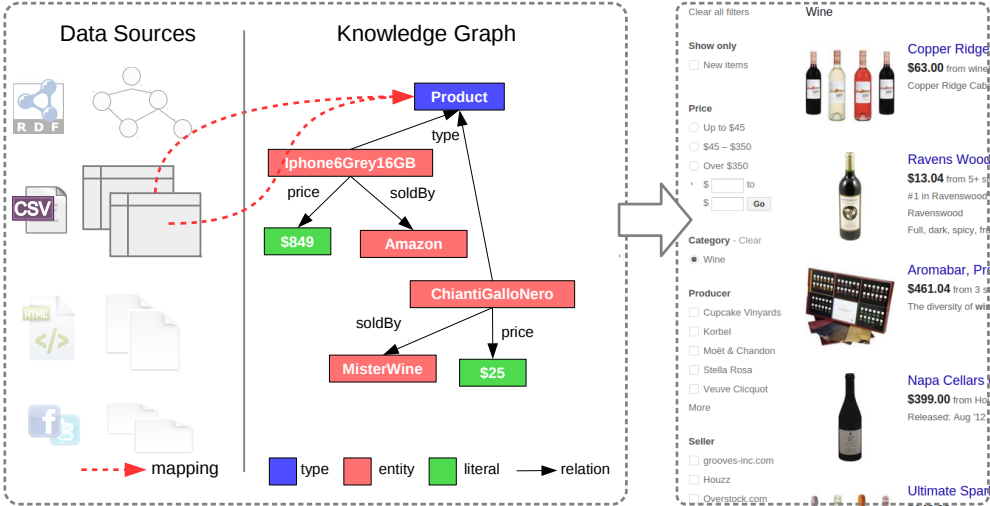


Figure 2.7: Bootstrapping a KG.

ketplaces exchanging data by means of structured or semi-structured formats, such for instance tabular CSV files. The goal of a CSE is to provide users with uniform and advanced access to offers sold by individual eMarketplaces and integrated into the dataspace.

Figure 2.7 depicts the dataspace of our hypothetical CSE resulting from the bootstrapping phase. Observe that the KG specifies only one type (i.e., Product) of instances. This representation is generalist, but sufficient to support the establishment of coarse-grained t-to-t mappings. From an end-user viewpoint, offers are represented by a set of facets such as price and soldBy. Observe that, as the KG specifies Offers as the only one type, such facets are somehow domain specific, given the current status of the KG (i.e., they are specific for offers). However, they capture only few salient aspects of offers, being the level of abstraction of the type Offer high. At this stage, source data is weakly integrated into the dataspace, but the CSE is still able to deliver search and browse features on top of it. However, such features (i.e., the faceted search interface depicted in Figure 2.7) are based on the representation provided by generalist facets and thus cannot go much far beyond full text and/or basic faceted search.

Figure 2.8 depicts the dataspace of our hypothetical CSE after the beginning of the maintenance phase. Now the KG provides a more rich and domain specific representation of instances. The KG has been enriched with new types, relations and facets, and thus supports the establishment of fine-grained mappings. The KG specifies two

## 2. DATASPACES AND KNOWLEDGE GRAPHS

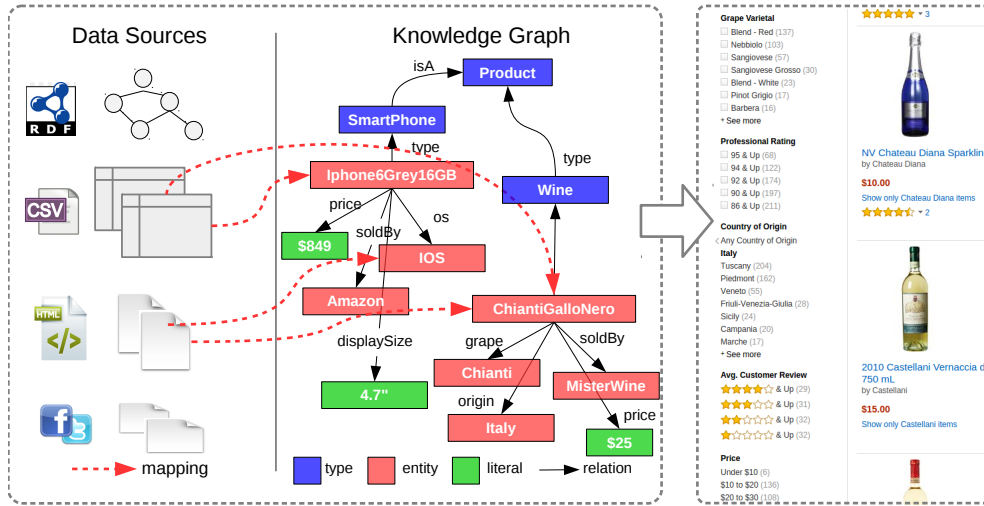


Figure 2.8: Maintaining a KG.

types of products, SmartPhone and Wine, with domain specific facets such as displaySize for smartphones and grape for wines. The presence of this more fine-grained representation supports the establishment of fine-grained mappings, thus tightening the integration between sources up. The CSE now starts to deliver enhanced user-experience on top of the domain specific representation provided by the KG.

### 2.3.2 Model, Map and Materialize

Three different but related data management tasks are continuously performed during the whole lifespan of a dataspace, from bootstrapping to maintenance: the *modeling* of the schema of the KG, the definition of *mappings* between the sources and the KG, and *materialization* of integrated data into the dataspace. Modeling accounts to enrichment of the schema of the KG with types, relations and facets. The goal of the first two activities (i.e., modeling and mapping) is to enable the materialization phase, where mappings from the sources to KG types and relations defined in the modeling phase are automatically leveraged in order to transform source data and import it into the dataspace.

Observe the materialization of source data is not strictly necessary, at least in principle. Mappings are a set of “transformation” rules (e.g, from a source type to a KG type) that can be leveraged to enable the formulation of queries over the integrated data without requiring to materialize it, in the vein of virtual data integration method-

ologies. However, materialization is necessary for the case of dataspace of DI applications that require further bulk analysis or processing over the integrated data (e.g, indexing into a full text search engine). Conversely, DI applications that have specific requirements concerning the freshness of data (e.g., a CSE for flights) may choose to materialize data in response to end-users queries and apply mappings to transform data “on the fly”.

In this methodology, the integration between data sources is initially weak (e.g, few coarse grained t-to-t and/or r-to-r mappings). The representation provided by the KG is incrementally enriched (i.e., modeling) over time with new types, relations, and facets for data presentation, which in turn supports the refinement of the integration of source data via the establishment of new and more fine-grained mappings. As the integration is tightened, and the representation is richer, the KG is incrementally enriched with domain specific facets, so as to support more advanced data access features, from an end-user perspective. This pay-as-you-go approach based on the refinement of the integration over time differentiates this methodology for managing a dataspace from traditional data integration methodologies that require fully fledged integration before any service can be delivered to end-users [70].

Modeling, mapping, and materialization are performed during the whole lifecycle of the dataspace, with one remark concerning modeling. In fact, while it is true that modeling is performed during the entire lifecycle of a dataspace, it has different goals whenever performed in the bootstrapping or maintenance phases, as schematized in Figure 2.9. The main goal of the bootstrapping phase is to provide a representation of data so as to immediately start the integration of data sources. In this phase, the representation provided by the KG is generalist, and aims at *covering* the higher amount of data as possible. For example, in a CSE, data may be represented with a generalist type such as Offer, and characterized by a price (a generalist facet). Conversely, the main focus during the maintenance phase is to provide a richer and *domain specific* representation of data, so as to tighten the integration up via the establishment of new mappings, but also to enable advanced data access features through the definition of domain specific facets.

### 2.3.3 Role of Domain Experts in Data Management Tasks

The contribution of domain experts is fundamental in the management of the dataspace, as they are in charge of incrementally drive, supervise, and validate all the data management tasks presented in the last section. Through an effective supervision, do-

## 2. DATASPACE AND KNOWLEDGE GRAPHS

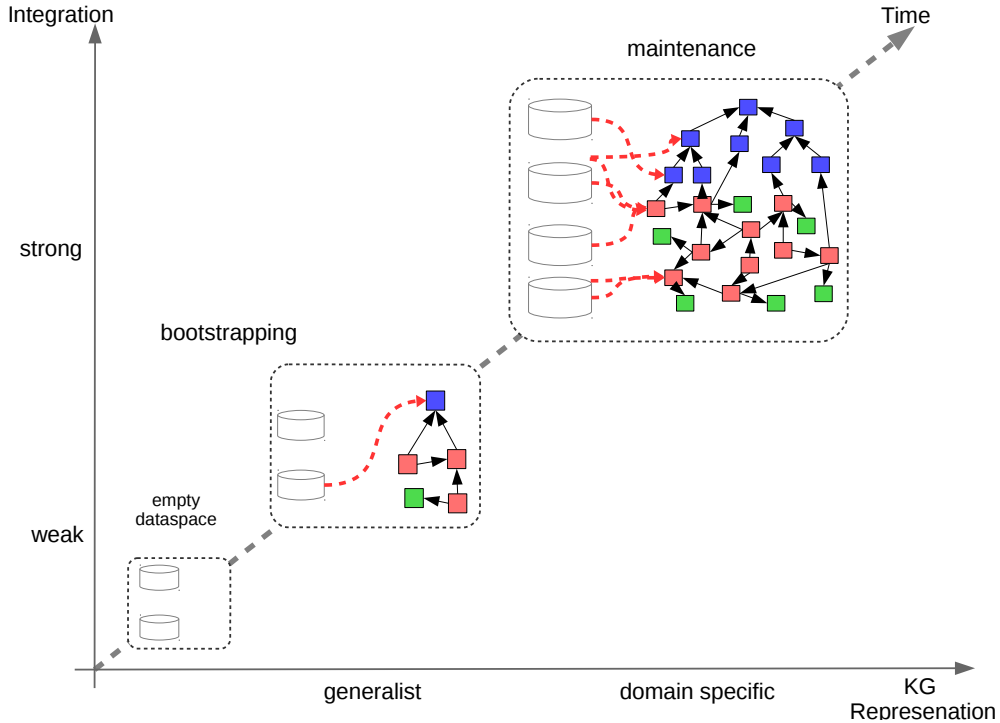


Figure 2.9: Evolution of a dataspace over time.

main experts incrementally enrich the KG with a richer and domain specific representation of instances, and establish mappings between elements of such representation and the sources. Providing automatic support for domain experts in these tasks is crucial for a proper and effective management of the dataspace.

Domain experts may be relatively autonomous in modeling the KG during the bootstrapping phase because it requires the definition of a generalist representation. However, enriching the schema so as to provide a more rich and domain specific representation of instances once in the maintenance phase, requires a deep understanding of the salient characteristics of instances for a large number of diverse domains, and thus is difficult and time consuming at a large scale. Intelligent tools are needed also to support domain experts in the establishment of mappings, to ultimately ensure a high quality of the data in the dataspace. As a final remark, domain experts must be equipped not only with tools that actually enrich the schema of the KG and establish mappings, but also with the necessary information to proper supervise such tools and validate their results, in order to ensure the adequate quality of data.

---

## **2.4 Summary**

---

In this chapter, we described the anatomy of a dataspace, its components and how these components, and in particular domain specific facets, provide a rich representation that can be leveraged by DI applications to deliver advanced data access features to end-users. We described the lifecycle of a dataspace, focusing on the bootstrapping and maintenance of the dataspace and its corresponding KG over time.

Domain experts in charge of supervising the whole data management process of a dataspace face hard challenges. Enriching a KG with a domain specific representation of instances is difficult, but nevertheless crucial. Such process requires, among other things, the extraction of end-user oriented relations that model the *salient* characteristics of instances of a KG type, which are named facets, and their consequent domain specific interpretation. However, this is particularly challenging and time consuming for domain experts to be performed at a large scale, because it requires extensive knowledge of disparate domains. Moreover, domain experts in charge of maintaining the dataspace and the KG must be equipped not only with tools to actually perform a diverse set of data management tasks, but also with the necessary information to properly supervise such tools and validate their results, in order to ensure the adequate quality of data.





# 3

## Literature Review

Chapter 2 provided a detailed discussion of the technical and methodological background in the management of a dataspace. We highlighted the necessity to support domain experts in crucial data management tasks such as the enrichment of the schema of the KG (to support the modeling phase), the establishment of mappings, and to provide domain experts with adequate information for supervision and validation. We now discuss the state-of-the-art in the management of dataspace, focusing on how effectively supports the the enrichment of the schema of the KG (Section 3.1) and the establishment of mappings (Section 3.2). Moreover, we will discuss also approaches that aim at *profiling* the current status of a KG, so as to provide the vital information to domain experts useful for supervision and validation (Section 3.3), before concluding the chapter in Section 3.4.

### 3.1 Knowledge Graph Schema Enrichment

---

State-of-the-art approaches support the enrichment of the KG schema by extracting types (e.g., [91, 148, 149, 126, 92, 93, 75, 72, 10, 130, 32, 82]), relations and facets [69, 132, 151, 33, 60, 100, 73] from source data. In our discussion we will focus our attention mainly on relation and facet extraction approaches, because they share the same goal of this dissertation: to enrich the KG with a rich representation of instances. As sources of a dataspace expose data in different formats (as discussed in Section 2.1.2), we classify and discuss state-of-the-art approaches depending on how structured is data from which they extract types, relations or facets.

### 3. LITERATURE REVIEW

---

#### 3.1.1 Extraction from Structured Sources

A large body of work focus on the extraction types and relations from databases (see, [128] for a recent survey). The general goal of such body of work is to bootstrap the schema of KG, eventually including a set of terminological axioms. The most elementary approach for the extraction of types and relations from a relational database consists of creating a KG type from each table (i.e., table-to-type) and a KG relation for each table column (i.e., column-to-relation) [13, 15]. An example of such approach is depicted in Figure 3.1. Although elementary, this approach inspired most of related work in this area, which aim at automatically discovering the semantics hidden in the database structure for extracted types and relations.

State-of-the-art approaches extract types and relations along with a set of terminological axioms involving them (e.g., [24, 95, 63, 144, 64]). Terminological axioms are usually in the form of domain and range restrictions and emerge from constraints specified in the database (e.g., primary or foreign keys). Consider, for instance, the relational database described in Figure 3.1. Following the elementary approach described earlier, the type `Product` is extracted from the table `Products`, while the type `ProductCategory` is extracted from the table `Product_Categories`. An instance of `Product` is characterized by a set of relations including `brand` and `category`. The semantics of the latter relation is specified by terminological axioms that emerges from the foreign key between tables `Products` and `Product_Categories` specified by the database. Hence, domain and range restrictions can be extracted in the form of terminological axioms like

$$\begin{aligned} \forall x \text{ (category}(x, y) \rightarrow \text{Product}(x)) \\ \forall y \text{ (category}(x, y) \rightarrow \text{ProductCategory}(y)). \end{aligned}$$

Approaches described in this section are mainly applicable during the bootstrapping phase of the KG lifecycle, but not in the maintenance phase, because they have two limitations: (1) they extract generalist relations, and (2) they consider a single data source as input. As a result, they are capable to extract only a generalist, and not domain specific, representation of instances. Moreover, this representation has limited power in representing data exposed by other data sources. These two limitations make the described approaches applicable during the early life of a KG, and are particularly useful when a KG is bootstrapped from a set of “core” data sources (e.g., a preliminary product catalog of the Comparison Shopping Engine described in Section 1.2). However, they provide only a generalist representation, which applies to data from single

### 3.1 Knowledge Graph Schema Enrichment

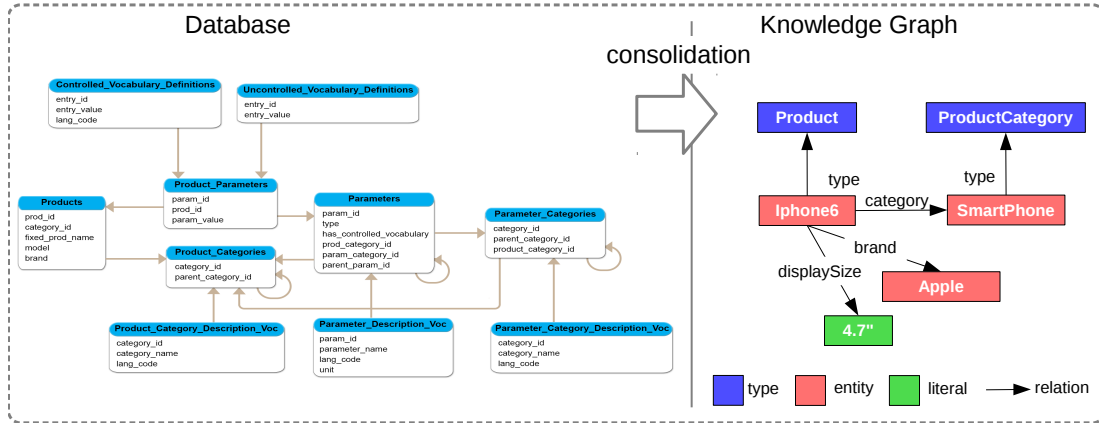


Figure 3.1: An example of extraction of the KG schema from a relational database.

sources in isolation and thus are of little help in the management phase.

#### 3.1.2 Extraction from Semi-Structured Sources

Related work in the extraction of types and relations from databases inspired the construction of the DBpedia KG [69]. The general idea behind the DBpedia project is that knowledge can be extracted from Wikipedia articles and consolidated into a KG which includes relations. In their very first work [5], DBpedia creators extract relations from Wikipedia infoboxes (e.g., the ones depicted in Figure 3.2). Infoboxes are consistently-formatted boxes, which are present in most of Wikipedia articles, and that synthetically characterize instances of a given set of types (e.g., musical artists). Infoboxes are generated through the application of domain specific *templates*, as for example the one depicted in Figure 3.3. Practically, a template consists of a structured set of predefined attributes (e.g., name) and relative values. The DBpedia relation extraction framework turns these attributes into relations and outputs the corresponding set of relational assertions involving them. For example, given the template in Figure 3.3, the DBpedia relation extraction framework extracts the relation `birthPlace` along with the relational assertion `birthPlace(BillGates, Seattle)`.

The approach to relation extraction just described has some evident, yet well known limitations, which stem from the intrinsic heterogeneity that characterizes Wikipedia templates and their attributes [69]. In fact, as the Wikipedia infobox templating system has evolved over time, different communities of Wikipedia editors use different templates to represent the same types of entities (e.g., Japanese cities vs Swiss cities).

### 3. LITERATURE REVIEW



Figure 3.2: Two infoboxes from Wikipedia (as of February 2016).

Templates created by different communities also model the same relations with syntactically different attributes (e.g., birthplace and placeofbirth). Moreover, attribute values are heterogeneous, in the sense that they are expressed using a wide range of different formats. As a result, the DBpedia relation extraction approach produces a set of relations that are: (1) not normalized, and (2) whose ranges potentially contains not normalized, possibly inconsistent values. In summary: while the relations resulting from the extraction cover a wide range of knowledge domains, they provide a low-quality representation of instances.

To address the limitations of the automatic extraction of relations, DBpedia creators proposed a different crowd-sourcing based paradigm, based on the explicit definition of mappings between template attributes and the well defined relations provided by the DBpedia Ontology, which constitutes the schema of the DBpedia KG [17]. These mappings, along with the DBpedia Ontology, are collaboratively curated and maintained by a community of users<sup>1</sup>. This significantly increases the quality and richness of representation provided by KG. Observe that a similar approach has been applied to the YAGO KG [132], which is the other noticeable KG built from knowledge extracted from Wikipedia articles. In fact, as for DBpedia relations, the extrac-

<sup>1</sup><http://mappings.dbpedia.org>

### 3.1 Knowledge Graph Schema Enrichment

```
{{Infobox person
| name           = Bill Gates
| image          = Bill Gates in WEF ,2007.jpg
| caption       = Bill Gates at the [[World Economic Forum]] in [[Davos]], 2007
| birth_name    = William Henry Gates III
| birth_date    = {{birth date and age|1955|10|28}}
| birth_place   = [[Seattle]], Washington, US
| occupation    = {{unbulleted list| Chairman of [[Microsoft]] (non-executive) | Co-chair of [[Bill & Melinda Gates Foundation]] | Director of [[Berkshire Hathaway]] | CEO of [[Cascade Investment]] }}
| networth     = {{increase}}[[United States dollar|US$]]53 [[1,000,000,000 (number)|billion]] (2010)<ref>
[http://www.forbes.com/profile/bill-gates/ Bill Gates profile]. Forbes.com. Retrieved April 2010.</ref>
| spouse       = Melinda Gates (married 1994)
| children     = 3
| residence    = [[Medina, Washington]]
| alma_mater   = Harvard University (dropped out in 1975)
| website      = {{URL|microsoft.com/presspass/exec/billg}}
| signature    = BillGates Signature.svg
| parents     = {{unbulleted list| [[William H. Gates, Sr.]] | [[Mary Maxwell Gates]] }}
}}
```

■ type   ■ entity   ■ literal   ■ relation

Figure 3.3: An example of template for persons.

tion approach implemented by YAGO relies on a predefined set of mappings between template attributes and KG relations.

Observations on the weaknesses of the DBpedia relation extraction approach are particularly relevant to this dissertation as they stem from data heterogeneity. In fact, while it is true that the DBpedia KG is enriched with types and relations extracted from a single data source (i.e., Wikipedia), it is also true that Wikipedia editors are organized in different communities each one with different and independent ways of classifying and representing source instances. Observed from this perspective, Wikipedia poses the same data management challenges faced by DI applications that deal with the integration of multiple data sources. Approaches based on the explicit definition of mappings between template attributes and KG relations introduce a bottleneck in the management of the dataspace, especially when extending the KG towards new domains of knowledge.

The problem of extracting domain specific facets, to represent instances described by Wikipedia articles have been studied also by Li et al. [151]. The approach proposed by the authors takes in input a set of Wikipedia articles, and outputs a set of facets. Observe that, in this case, all the extracted facets share a common domain which includes all the entities described by input articles. Conversely, the ranges of facets include all the instances described by articles that are hyperlinked from input articles. Hyperlinks between two given articles are considered as a clue of the fact that there exists a relationship between instances described by those articles. Given a set of input articles, the proposed approach selects all the instances described by hyperlinked articles and partitions them into facet ranges, based on the classification provided by Wikipedia categories. As Wikipedia categories provide a very rich classification scheme [132]

### 3. LITERATURE REVIEW

---

The image shows a web page interface for book discovery. On the left, there are two filter sections: 'Author' and 'Language'. The 'Author' section lists Mia Sheridan (1), Laurann Dohner (2), Vi Keeland (3), Penelope Ward (2), Roger Priddy (312), Brandon Stanton (1), and Mark R. Levin (1), with a '+ See more' link. The 'Language' section lists English (19,390,047), German (2,679,317), French (2,537,471), Spanish (1,318,487), Italian (937,581), Russian (1,013,914), and Chinese (643,200), also with a '+ See more' link. On the right, two book listings are shown. The first is 'Fates and Furies: A Novel' by Lauren Groff, featuring a blue cover with white text. It has a 4.5-star rating (6 reviews), a hardcover price of \$17.53, and a Kindle Edition price of \$13.59. The second is 'The Story of Elena Ferrante' by Elena Ferrante, featuring a cover with a woman and a child. It has a 5-star rating and a paperback price of \$9.99. The Kindle Edition price is partially obscured.

Figure 3.4: A Web page describing books.

the authors devise an automatic approach to find the partition that best balances the generality and the specificity of facets.

Facets have been extracted from HTML documents provided by semi-structured data sources using unsupervised [33] or supervised [60] machine learning techniques. Similarly to Li et al. [151], those approaches take in input a set of HTML documents describing domain specific source instances (e.g., books from Figure 3.4) and output a set of facets, specified in terms of the same domain, which includes all the instances described by input documents, and with facet ranges extracted from HTML elements embedded in the documents (e.g., the ones highlighted in red in Figure 3.4). Again, the main challenges that such approaches must face stem from the intrinsic heterogeneity of values extracted from HTML elements. In order to address these challenges, QD-Miner [33] propose an unsupervised approach based on the adaptation of the Quality Threshold clustering algorithm [46], which devises a set of facet ranges by jointly maximizing specificity and coverage across instances described by the input documents. The same challenge is tackled by Kong and Allan [60], who propose a Graphical Model based approach that is based on the estimation of the probability of two values to be part of the same facet range, based on training data.

### 3.1.3 Extraction from Unstructured Data Sources

Some state-of-the-art approaches extract facets from unstructured data sources [100, 110]. Large unstructured text document collections are analyzed in order to identify a set of relevant terms, that will form the ranges of extracted facets. Such approaches typically rely on lexical resources such as WordNet [37] to handle the intrinsic heterogeneity and ambiguity of source data, but also on extra information provided by full text query logs, when available. Other state-of-the-art approaches extract types taxonomies from unstructured data sources [82, 130, 32, 126, 92, 93, 75, 72, 10, 149]. State-of-the-art approaches in this area analyze lexico-graphical patterns such as *x is a y* found in text (also known as Hearst Patterns [44]) and devise a type taxonomy from them. They do not extract relations, nor facets. Observe that approaches discussed in this section cover unstructured data sources, which represent only one of the possible formats used to exchange data in dataspace. For this reason, they are complementary to the contributions of this dissertation, which focus on more structured data (see Section 4.6).

## 3.2 Mapping Discovery

---

State-of-the-art approaches related to mapping discovery automatically establish mappings between source and KG instances, types and relations, provided by structured [38, 122, 27, 71, 131, 83, 78, 104], semi-structured [74, 143, 120, 145, 112, 156, 86, 154, 153, 155] and unstructured [87, 119] data sources. Related work that is most relevant to this dissertation focuses on the establishment of mappings between structured and semi-structured data sources and the KG, and we review them in the following sections. We will not cover the establishment of mappings from unstructured data sources, as state-of-the-art approaches focus on the establishment of i-to-i mappings between source instances described in free text and KG instances. Those approaches, proposed in the literature under the general classification of Named Entity Recognition and Linking, are less relevant to this dissertation, which focuses more on the schema of the KG (i.e., types and relations). However, we point the interested reader to [87, 119], for two surveys of that area.

### 3. LITERATURE REVIEW

---

#### 3.2.1 Mapping Structured Sources

Early approaches on the discovery of mappings in presence of structured sources have been provided by the Data Management community, under the general classification schema matching [128] approaches. They support the management of particular dataspace where sources provide direct access to tables and views of a database. However, these approaches can be hardly applied to most of the dataspace on a Web scale, where sources are distributed over the Web and are accessible via the HTTP protocol [39]. More recently, the discovery of mappings between structured data sources have been studied also in the ontology matching field (see [122] for a survey), with focus on data sources that represent and exchange data by means of the RDF data model.

Ontology matching approaches mainly focus on the establishment of t-to-t mappings and r-to-r mappings, while the establishment of i-to-i mappings has been studied under the general classification of Instance Matching [38]. The widely adopted approach in establishing t-to-t and r-to-r mappings is to analyze terminological axioms (i.e., subtype relations, domain and range restrictions) as well as types' and relations' lexicalizations. State-of-the-art approaches apply a composition of different criteria in order to compute the similarity between two types or relations. Syntactic similarity criteria are used to compare lexicalizations (see, [26] for a survey of the main approaches). Structural similarity criteria apply graph matching techniques to the graph extracted from terminological axioms, which includes the subtype hierarchy but also domain and range relationships between types and relations (e.g., [83, 71, 131]).

State-of-the-art ontology matching approaches perform particularly well on t-to-t mappings as testified by results of the Ontology Alignment Evaluation Initiative [123], a yearly competition held at the Ontology Matching Workshop active since 2006<sup>2</sup>. Unfortunately, current ontology matching approaches do not perform comparably well in the establishment of r-to-r mappings [78, 104, 27]. In this context, approaches based on well chosen string similarity metrics tuned to consider relations outperform more "holistic" ones that jointly establish t-to-t, r-to-r and i-to-i mappings [27] such as [71, 131]. The important remark is that information about the domain and range of relations is crucial for effectively establishing r-to-r mappings. In structured sources that adhere to the RDF data model, this information is explicitly available (see Section 2.1.2).

---

<sup>2</sup><http://oaei.ontologymatching.org/>



However, in the case of generalist relations, domain and range information may not be discriminative enough, or even be absent (i.e., no terminological axioms specified). In fact, generalist relations are often underspecified [1] as a result of precise modeling choices that aim at favoring sharing and re-use, e.g., the relation `dc:date` from the general purpose Dublin Core Elements Vocabulary<sup>3</sup>. Still, underspecified relations may be used by a data source with a more domain specific semantics. This kind of domain specificity (i.e., emerging from usage) cannot be captured by state-of-the-art relation mapping approaches are based on the analysis of terminological axioms.

### 3.2.2 Mapping Semi-Structured Sources

A large body of work discovers mappings between semi-structured data sources and the KG, and in particular of tabular (e.g., CSV) data, under the general classification of table annotation approaches [74, 143, 120, 145, 112, 156, 86, 154, 153, 155]). The task of interpreting a semi-structured table gained attention since a seminal study from 2008 by Cafarella et al. [21], who shown that the Web pages contain a huge amount of high-quality tables containing useful relational data. Table annotation approaches are particularly relevant to this dissertation, because semi-structured tables are one of the most commonly used exchange format in dataspace. Figure 3.5 depicts the input (i.e., a table) and the expected output (i.e., a set of mappings) of a generic table annotation algorithm. Many table annotation approaches assume that a table include a *subject column* that is, a column containing the subject entities described by the table [143, 145, 155], while the remaining columns contain entities or literal values that are in a relationship with subject entities. The goal of a table annotation approach is to interpret a table by establishing mappings between cells, columns and pair of columns, and a KG instances (i-to-i), types (t-to-t), and relations (r-to-r), respectively.

Venetis et al. [143] apply a maximum likelihood inference model to estimate the probability of a relation that holds between values from two columns. The same inference model is applied in order to annotate columns with types. The estimation of the maximum likelihood model is based on computation of the frequencies within the KG of all pairs of values of the same row as subjects and objects of relations, and does not consider any type information. Wang et al. [145] interpret tabular data sources using the Probase KG [149]. Probbase is a probabilistic KG and it provides scores that model the *plausibility* and the *ambiguity* of entities being instance of a certain type and a type being characterized by certain relations. Those scores are computed during the

---

<sup>3</sup><http://dublincore.org/documents/dces/>

### 3. LITERATURE REVIEW

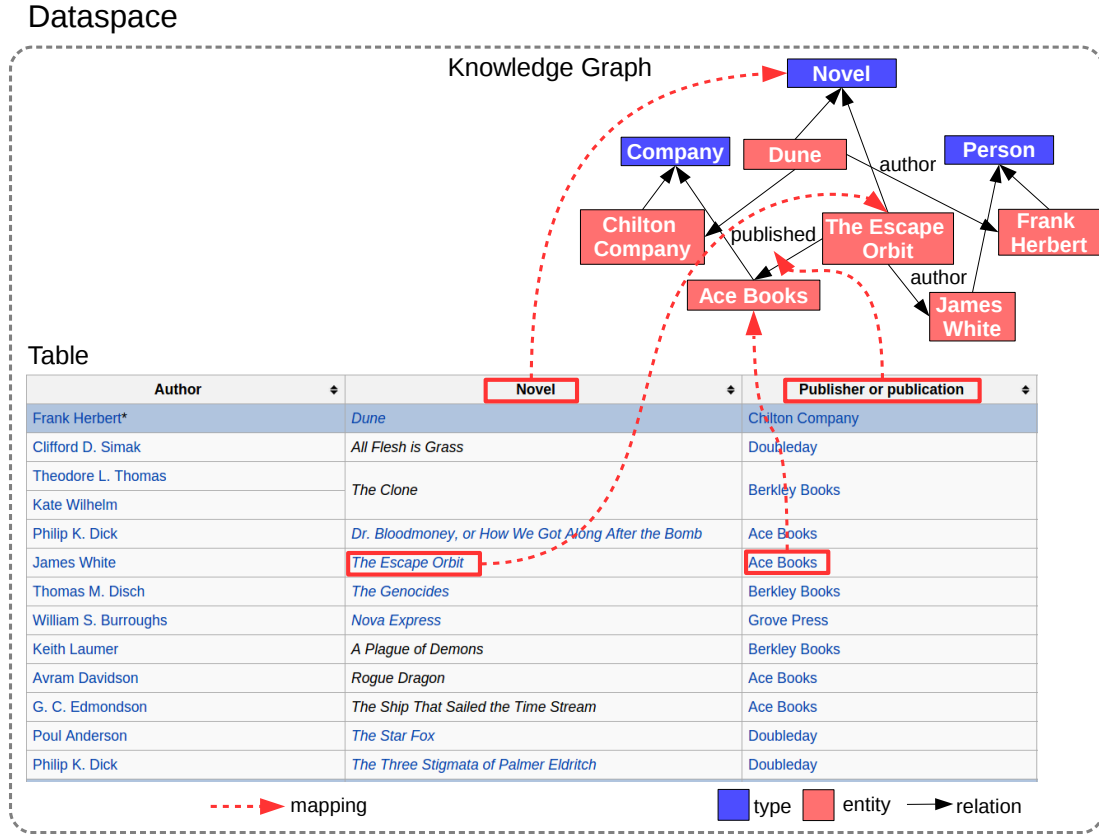


Figure 3.5: Schematization of the input and the output of a table annotation approach.

bootstrapping of the KG. They interpret a table by identifying the subject column, establish a t-to-t mapping between it and a KG type and then establish r-to-r mappings between the remaining columns and KG relations. Their approach is KG dependent, since they rely on plausibility and ambiguity scores provided by Probase in order to compute the overall score of a candidate mapping.

TableMiner [155] interprets tables with entities, types and relations from the generalist KG Freebase. One of the main contributions of TableMiner is the usage of contextual information extracted from the Web page that contains the table (e.g., table caption, surrounding text, RDFa/Microdata annotations). Following the same intuition of Wang et al. [145], TableMiner starts with identifying the subject column and provides a set of preliminary i-to-i and t-to-t mappings. Those mappings are then jointly refined using an iterative learning procedure. r-to-r mappings are then computed based on the result of the refinement phase.

The above mentioned table annotation approaches tackle the establishment of the

various type of mappings independently and propose a principled combination of the result. However, also more holistic approaches have been proposed by Limaye et al. [74] and Mulwad et al. [86]. They model the interdependence between table cells, columns and rows using probabilistic graphical models [59] and perform collective inference in order to jointly compute the optimal mappings. However, more recent approaches that tackle the establishment of different (i.e., i-to-i, t-to-t, r-to-r) mappings separately outperformed the joint inference based ones [143, 155].

Karma [57, 136, 137, 113] is designed for situations where data from different data sources to be interpreted partially overlap and propose a semi-automatic and interactive mapping process that learns how to establish new mappings based on mappings previously defined. The domain expert is asked to initially specify t-to-t mappings between columns and KG types. Then, Karma automatically computes a set of r-to-r mappings between pairs of table columns and KG relations. At each step, the domain expert can revise and refine the mappings automatically computed by the system. As Karma assumes that data from the sources to be interpreted partially overlap, it learns how to suggest t-to-t [57] and r-to-r mappings [137], by using a Conditional Random Fields [65] (CRF) based model trained with previously defined t-to-t mappings that uses features extracted from values in the columns. Given the defined t-to-t mappings, Karma is also able to infer r-to-r mappings by leveraging axioms in the input ontology in terms of relation domains and ranges.

### 3.3 Knowledge Graph Profiling

---

Another body of work that is relevant to this dissertation is related to data *summarization* [152, 141, 84, 22, 51, 111, 61, 29, 67, 4], which aims at profiling a KG by providing an abstract representation of its data (i.e., a *summary*) in terms of types, relations and their reciprocal usage. Effective summarization approaches are crucial in order to support both algorithms and domain-experts in the supervision of data management tasks in dataspace, by providing the capability to answer to questions like: (1) what instances or types are described in the KG? (2) What relations are used to characterize the instances? (3) What types of instances are linked by a certain relations and how frequently? We now review state-of-the-art summarization approaches explicitly proposed for large KGs.

A schematization of the summarization process for a KG is provided in Figure 3.6. A first body of work devise summarization models aimed at identifying portions of

### 3. LITERATURE REVIEW

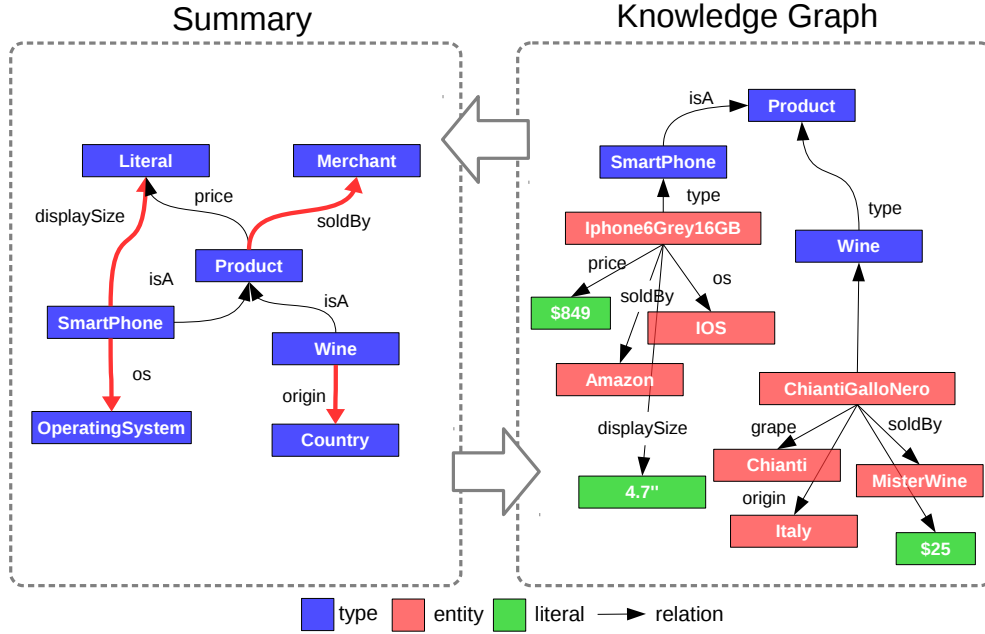


Figure 3.6: Overview of a generic structured data summarization approach.

data that are estimated to be more relevant in order to understand and explore the structure and the representation of KG instances. Early approaches rank the axioms of an ontology based on their salience so as to present a view over the ontology to the users [152], while more recent approaches such as RDF Digest [141] identify the most salient subset of data using a more rich set of different criteria, including the distribution of instances.

A second body of work focus on the summarization of a KG by reporting statistics about the usage of types and relations in the data. Among these approaches Loupe [84], the most noticeable one, extracts types and relations along with a rich set of statistics on their usage for the representation of instances. Loupe provides a summary of a KG by extracting a rich set of abstract relation patterns found in the data and their corresponding frequency. Those patterns are extracted in the form of  $(C, P, D)$ , where  $C$  and  $D$  are types, and  $P$  is a relation. Patterns provide an abstract overview of how relations are used to represent instances of particular types. In [22], authors consider types and relation usage in the summarization process of an RDF graph and use information similar to relation patterns. A similar approach is also used in MashQL [51], a system proposed to query graph-based source data (e.g., RDF) without prior knowledge about the structure and schema of a data source. Pattern extraction from RDF data is also discussed in [111], but in the context of domain

specific experiments and not with the purpose of defining a general structured data summarization framework.

Other approaches that tackle the KG summarization problem do not extract relationships between source types but instead provide several other statistics. SchemeEx extracts interesting theoretic measures for large KGs, by considering the co-occurrence of types and relations [61]. A data analysis approach on RDF data based on a warehouse-style analytic is proposed in [29]. 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 KGs, and long runtime. In the same line of work, Linked Open Vocabularies<sup>4</sup>, RDFStats [67] and LODStats [4] provide several statistics about the usage of types and relations, but without representing connections between types.

### 3.4 Summary

---

In this chapter we reviewed the state-of-the-art in supporting domain experts in the main data management tasks for the maintenance of dataspace. We started from by reviewing approaches that provide automatic support to domain experts in the modeling of the KG schema, and in particular approaches whose goal is to enrich the schema of the KG (Section 3.1). We saw that a large body of work have been proposed to support the enrichment of the schema by extracting types, relations and facets from diverse types of sources, from structured sources to unstructured sources. Then, we turned our attention on the discovery of mappings between source data and elements of the KG schema, focusing in particular on structured and semi-structured data sources (Section 3.2). Finally, we reviewed approaches that aim at profiling a KG, so as to provide domain experts with the necessary information needed to supervise and validate approaches that automatize the above data management tasks (Section 3.3).

As we saw in this chapter, there are a variety of different approaches that can be potentially applied to support domain experts in the management of the dataspace. However, such approaches have several limitations, which we summarize in the following.

---

<sup>4</sup><http://lov.okfn.org/>

### 3. LITERATURE REVIEW

---

#### **Lack of Support for the Extraction of Facets**

Most of the approaches described in Section 3.1, focus on the extraction of types and relations, while few focused on the extraction of facets [100, 110, 151, 33, 60]. Although facets are by all means relations, they serve different purposes, being relations back-end oriented and facets end-user oriented. Moreover, to the best of our knowledge, no work focused on extracting domain specific facets. As a result, the current state-of-the-art supports domain experts in the enrichment of the schema of the KG with types, relations, and generalist facets, but not domain specific ones.

#### **Limited Insights About the Status of the KG**

Related work in the profiling of KGs is more limited, compared to KG schema enrichment and mapping discovery. From the review given in Section 3.3, emerges that state-of-the-art approaches currently provide a profile of the KG that is either incomplete, because includes only a portion of data estimated to be more relevant in order to understand and explore the structure of the KG, or a set of statistics without being able to capture patterns in the data, which can in principle provide useful insights to domain experts during the execution of data management tasks for the maintenance of the dataspace.

In summary, current state-of-the-art provides limited support for the extraction of facets, and in particular domain specific ones, which are currently left to nearly completely manual work by domain experts. To overcome this limitation, in this dissertation we propose an approach to the enrichment of the KG with domain specific facets composed by two different steps, discussed in Chapters 4 and 5. In particular, we first extract domain specific facets from structured data sources based on existing t-to-t mappings already established by domain experts, to enforce domain specificity. Then, we provide a domain specific interpretation of extracted facets, by reusing relations specified by the KG of the dataspace or eventually by other external KGs. While the first approach specifically enriches the KG with domain specific facets, the second approach reconciles the end-user oriented representation of instances provided by facets to the back-end oriented representation provided by relations with the benefit of tightening the integration between the sources up and thus empowering the capability to provide advanced data access features to end-users of the dataspace. As supervision and validation of domain experts is crucial in this picture, in Chapter 6 we propose a KG profiling approach that provides domain experts with insights about the current

status of the KG. In particular, our approach is capable to provide an overview of the specificity of relations of a KG, which constitutes vital information for domain experts in charge of supervising the enrichment of the KG schema with domain specific facets and their domain specific interpretation with KG relations.





# 4

## Domain Specific Facet Extraction

### 4.1 Overview

---

*Type-based* and *facet-based* browsing are two examples of data access features that many DI intensive applications, such as the Comparison Shopping Engine described in Section 1.1, aim to deliver to their users. These features require the creation and maintenance of a domain specific representation respectively based on KG types, and *facets* [138, 147]. As introduced in Sections 2.1.1 and 2.2, KG types provide a coarse-grained representation of all the instances in the dataspace, which helps users to rapidly recall the “family” of instances they are interested in. Conversely, facets provide a fine-grained representation of KG instances, which helps users to rapidly recall instances with specific characteristics (e.g., “Grape: Barolo”, “Type: Red Wine”).

Facet creation and maintenance is an extremely time and effort consuming task in the context of DI applications and specifically in the context of CSEs. This task is left to manual work of domain experts and requires a deep understanding of the salient characteristics of KG instances (e.g., wines are characterized by grape, type, provenance, and so on) for a large number of diverse product types or domains. As a result, many CSEs provide only few generalist facets (e.g., price and merchant) and others provide a richer set of domain specific facets but only for a limited amount of popular product types.

This chapter introduces an automatic, domain specific facet extraction approach, which supports domain experts in the creation of significant domain specific facets. Facets are specialized relations aimed at model the salient characteristics of entities from specific domains (e.g., news, actors, or wine bottles), and thus the technical

## 4. DOMAIN SPECIFIC FACET EXTRACTION

---

problem tackled in this chapter accounts to the extraction of salient domain specific relations. Our approach leverages the information already present in the dataspace, namely (i) taxonomies used to classify instances from structured the data sources and (ii) t-to-t mappings established between source and KG types, to suggest meaningful facets specific for a given KG type. In fact, unlike the KG types, which have to cover instances from very diverse domains, source taxonomies are often domain specific. Domain experts map domain specific types from source taxonomies (e.g., Barolo) to generalist types in the KG (e.g., Wines). The intuition behind the proposed approach is to reuse the fine-grained source types that occur in several source taxonomies mapped to KG types (e.g., Barolo, Cabernet), to extract a set of relevant facets for a given KG type. In addition, since our approach extracts facets from source types, the generation of the corresponding faceted assertions for extracted facets over dataspace instances is straightforward, supporting facet-based browsing.

The proposed approach incorporates an automatic facet extraction algorithm that consists of two steps: *extraction* of potential facet values (e.g., Cabernet) and *clustering* of facet values into sets of mutually exclusive facet ranges (e.g., Bordeaux, Cabernet, Chianti). The algorithm is based on structural analysis of source taxonomies and on *Taxonomy Layer Distance*, a novel metric introduced to evaluate the distance between source types in different heterogeneous taxonomies. Experiments conducted to evaluate the approach show that our algorithm is able to extract meaningful facets that can then be refined by domain experts. The remaining of the chapter discusses in details this contribution, starting from Section 4.2, in which we provide a definition of the problem of extracting domain specific facets. In Section 4.3 we describe our facet extraction approach, and we discuss how to populate the KG with faceted assertions for the extracted facets in Section 4.4. We evaluate our approach in Section 4.5. Comparison with related work (Section 4.6) and a final summary of the contribution here described (Section 4.7) end the chapter.

### 4.2 Problem Definition

---

Our goal is to extract a set of facets so as to provide a domain specific representation of instances of a specific KG type. As input of the problem, we assume that there exists a KG subtype graph and a set of mappings between types from a set of source taxonomies and KG types. We assume that the mappings have many-to-one cardinality, i.e., many source types in each source taxonomy are mapped one KG type. In the rest of this section we recall the formal definitions introduced in Chapter 2 and define the

Domain Specific Facet Extraction problem.

### KG Subtype Graph and Types

The KG subtype graph  $G = (\mathcal{N}^C, \preceq)$  consists of a set of  $\mathcal{N}^C$  of KG types and a subtype relation  $\preceq$ . We recall from Section 2.1.1 that such subtype graph is constructed from terminological axioms about KG types.

### Source Taxonomy and Types

A source taxonomy  $\tilde{T} = (\tilde{\mathcal{N}}^C, \tilde{\preceq})$  consists of a set  $\tilde{\mathcal{N}}^C$  of source types and a subtype relation  $\tilde{\preceq}$  that imposes a partial order over  $\tilde{\mathcal{N}}^C$ . We also assume that a source type  $\tilde{C}$  is associated to one or more lexicalizations. Consistently with Section 2.2, we assume that the correspondence between a source type  $\tilde{C}$  and its lexicalizations is provided by the values taken by the function  $lex(\tilde{C})$ .

### T-to-t Mapping

A t-to-t mapping  $C(x) \leftarrow \tilde{C}(x)$  is a correspondence between a *leaf* type  $\tilde{C}$  of some source taxonomy  $\tilde{\mathcal{N}}^C$  and a KG type  $C$ . The semantics of a t-to-t mapping between  $\tilde{C}$  and  $C$  is that instances that are classified under  $\tilde{C}$  at the source are classified as instances of  $C$  once they are integrated into the dataspace.

### Facet

Recall from Section 2.2 that a *facet* can be defined as “a clearly defined, mutually exclusive, and collectively exhaustive aspect, property, or characteristic of a class or specific subject” [138]. A facet  $F = \langle C, \mathcal{V} \rangle$  is a relation that holds between a *domain* of instances of the KG type  $C$  and a *range* of facet values  $\mathcal{V} = \{v_1, \dots, v_n\}$ . For example, a facet such as  $F = \langle \text{Wines}, \{\text{Italy}, \text{France}, \dots, \text{Chile}\} \rangle$  may conceptualize the relation that holds between instances of the KG type Wines and the respective country of origin.

## 4. DOMAIN SPECIFIC FACET EXTRACTION

---

### Facet Extraction

Given a KG type  $C$  and a set of t-to-t mappings between source types  $\tilde{C}_1, \dots, \tilde{C}_i$  and  $C$ , extract a set of facets  $\mathcal{F}^C$  that provide a domain specific representation of instances of  $C$ . More formally, given  $C$  and mappings between source types  $\tilde{C}_1, \dots, \tilde{C}_i$  and  $C$ , we aim at extracting a set of facets

$$\mathcal{F}^C = \{F_1 = \langle C, \mathcal{V}_1 \rangle, \dots, F_n = \langle C, \mathcal{V}_n \rangle\}$$

where the domain of all the facets is fixed and includes all instances of the KG type  $C$ . Observe that, as the facet domain is fixed, solving the Domain Specific Facet Extraction Problem basically accounts to the extraction of facet ranges  $\mathcal{V}^C = \{\mathcal{V}_1, \dots, \mathcal{V}_n\}$  for domain specific facets, given a KG type  $C$ .

### 4.3 Approach

---

The approach to facet extraction proposed in this dissertation is sketched in Figure 4.1 and is aimed to support domain experts who are in charge of maintaining domain specific KGs and corresponding mappings. Domain experts trigger the extraction process for a specified KG type. An automatic facet extraction algorithm suggests a set of domain specific facets to domain experts, who inspect, validate and refine them, deciding which facets will be consolidated into the KG.

The automatic facet extraction algorithm at the core of the proposed approach is inspired by the following principle: specialized taxonomies used in data sources contain information that can be analyzed to extract a set of significant domain specific facets that characterize instances of a specific KG type. The facet extraction algorithm extracts the set of facets  $\mathcal{F}^C$  for a KG type  $C$  using a two-phase process:

1. **Value Extraction:** a set of normalized facet values is produced by case lowerization, special characters removal and stemming of all the source types mapped to  $C$ .
2. **Value Clustering:** facet values are clustered together into facet ranges according to source taxonomies structural analysis. Since we look for facets of mutual exclusive values, we admit facets containing at least two values. Thus, values that cannot be added to any facet (i.e., clusters of one element) are discarded.

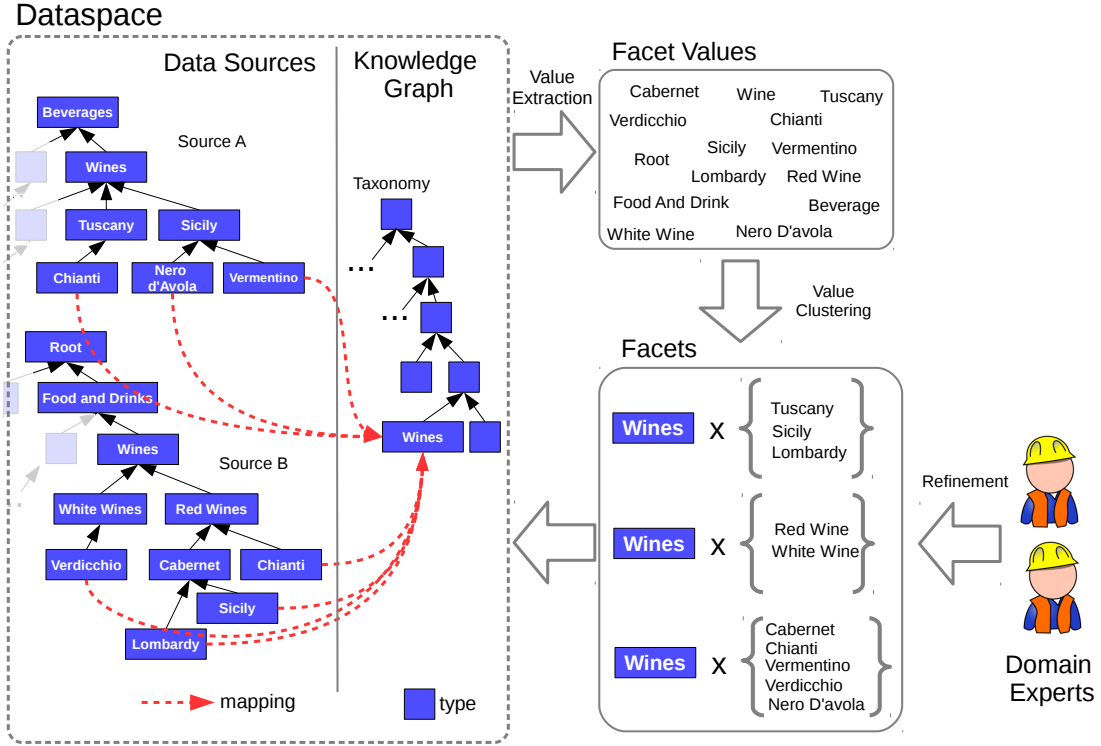


Figure 4.1: Overview of the proposed approach to domain specific facet extraction.

### 4.3.1 Value Extraction

During this phase we identify the set of facet values that are frequently used for the representation of source instances. In order to identify such values for a KG type  $C$  we rely on existing mappings to  $C$ . For each source taxonomy  $\tilde{T}$  we form the set  $N_{\tilde{T}}^C$  of values occurring as lexicalization of source types mapped to  $C$  and all their ancestors in the respective source taxonomy. The level of detail of source taxonomies can be different in each source taxonomy, thus ancestors' lexicalizations are included in  $N_{\tilde{T}}^C$  to consider every possible significant value. The set  $N_{\tilde{T}}^C$  for a KG type  $C$  and a source taxonomy  $\tilde{T}$  is defined as

$$N_{\tilde{T}}^C = \{lex(\tilde{C}) \mid C(x) \leftarrow \tilde{C}(x) \vee C(x) \leftarrow \tilde{C}'(x), \text{ with } \tilde{C}' \preceq \tilde{C}\}.$$

The set  $V_{\tilde{T}}^C$  of normalized values is obtained by applying case lowerization, special characters removal and stemming to  $N_{\tilde{T}}^C$ . As far as stemming is concerned, we use Hunspell Stemmer<sup>1</sup> to normalize values' terms with respect to their singular form.

<sup>1</sup><http://hunspell.sourceforge.net/>

## 4. DOMAIN SPECIFIC FACET EXTRACTION

---

Hunspell stemmer is based on language dependent stemming rules that are available for most of languages. Normalized values are then unioned together to form the set  $V^C$  of facet values for a for a KG type  $C$ . In this phase, duplicated values are removed. The set  $V^C$  of unique values for a KG type  $C$  over all the  $n$  source taxonomies is defined as

$$V^C = \bigcup_{i=1}^n V_{T_i}^C.$$

After normalization and unioning, a simple ranking function is applied to  $V^C$ . Unique values are ranked according to their frequency over the all sets  $V_{T_i}^C$ . Intuitively, the more a value occurs as source type mapped to the KG type  $C$ , the higher rank it will get. Based on this ranking we reduce  $V^C$  to the set  $V_k^C$  of the top  $k$  frequent values. The rationale behind this choice is to keep only those values that are more commonly used across many independent and heterogeneous sources and thus are likely to be more relevant for the fine-grained representation of dataspace instances. The set  $V_k^C$  of the top frequent values produced by this phase represents the input for the next phase. In addition, we keep track of the (possibly) many source types to which each value  $v \in V_k^C$  correspond. In this way, the annotation of the dataspace instances with facet values extracted by the algorithm is straightforward.

**Example.** Given the two taxonomies  $A$  and  $B$  in Figure 4.1, for the KG type Wines

$$N_A^{\text{Wines}} = \{\text{Beverages, Wines, Tuscany, Chianti, Sicily, Nero d'Avola, Vermentino}\}$$

and

$$N_B^{\text{Wines}} = \{\text{Root, Food And Drinks, Wines, White Wines, Verdicchio, Red Wines, Cabernet, Lombardy, Sicily, Chianti}\}.$$

After normalization, values from  $N_A^{\text{Wines}}$  and  $N_B^{\text{Wines}}$  form the set of unique facet values

$$V^{\text{Wines}} = \{\text{Root, Beverage, Food And Drink, Wine, Lombardy, Cabernet, Tuscany, Chianti, Sicily, Vermentino, Nero d'Avola, White Wine, Verdicchio, Red Wine}\}.$$

### 4.3.2 Value Clustering

Values in  $V_k^C$  are clustered to form the set of facet ranges  $\mathcal{V}^C$ . The goal is to cluster together all values that are more likely to be coordinate values identifying objects of the

same relation. As an example, suppose that the set  $V_k^{\text{Wines}} = \{\text{Cabernet, Chianti, Lombardy, Sicily}\}$  is produced in the Value Extraction phase. An ideal clustering should be  $V_1 = \{\text{Cabernet, Chianti}\}$  and  $V_2 = \{\text{Lombardy, Sicily}\}$  because each facet range refers to a same relation (i.e., the wine’s grape variety and the origin Italian region).

So as to discover the set  $\mathcal{V}^C$  of facets ranges over  $V_k^C$ , the DBSCAN *density-based* clustering algorithm [35] is used. DBSCAN clusters together values within a maximum distance threshold  $\epsilon$  and satisfying a cluster density criterion and discards as noise values that are distant from any resulting cluster. The DBSCAN algorithm requires in input the minimum cardinality of expected clusters and the maximum distance threshold  $\epsilon$ . In the proposed approach, the minimum cardinality is set to 2, while the best value for  $\epsilon$  is found empirically (see Section 4.5). Finally, DBSCAN does not require a number of expected clusters as input. DBSCAN is used for several reasons. The proposed approach must deal with heterogeneous taxonomies, thus it cannot make any assumption about the shape of clusters (i.e., facet ranges) and it must employ clustering techniques that incorporate the notion of noise. Otherwise, clustering algorithms requiring the expected number of clusters as input are not suitable (e.g., KMeans [43]) since the number of facets to detect is not known in advance.

In order to use the DBSCAN clustering algorithm it is crucial to provide an effective distance metric between the values. We propose a distance metric that considers near those values that refer to a same characteristic of instances, according to a taxonomic structural criterion. We now formally define the proposed distance metric, starting from the principle that it aims at capturing: source type mutual exclusivity.

### Source Type Mutual Exclusivity Principle

Recall from Section 2.2 that a facet is “a clearly defined, mutually exclusive, and collectively exhaustive aspect, property, or characteristic of a class or specific subject” [138]. The Source Type Mutual Exclusivity principle (STME) states that the more two values refer to mutually exclusive types, the more they should be grouped together into the same facet range. Given two source types  $\tilde{C}_1$  and  $\tilde{C}_2$ , their occurrence as siblings may indicate that  $\tilde{C}_1$  and  $\tilde{C}_2$  are mutually exclusive (e.g., Cabernet and Chianti in Figure 4.2). STME is a *structural* principle: it takes source taxonomies structure into account by considering reciprocal relationships among types.

## 4. DOMAIN SPECIFIC FACET EXTRACTION

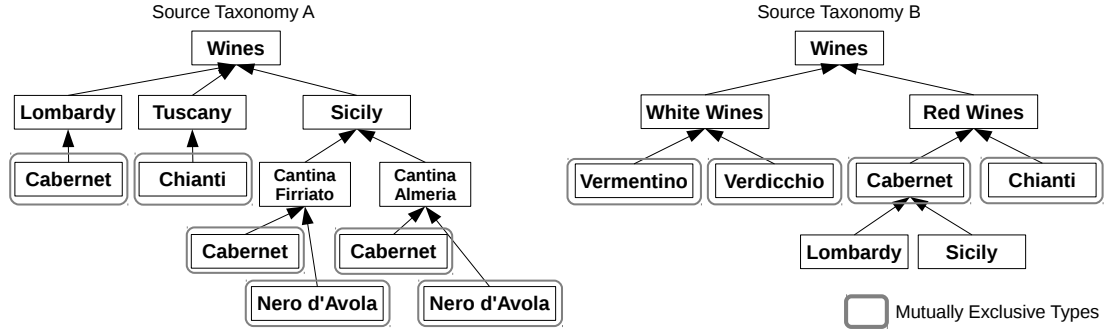


Figure 4.2: An example of mutually exclusive source types.

### Taxonomy Layer Distance

We propose a distance metric that captures the STME principle by considering sibling relationships between source types, or more generally, the co-occurrence of types on a same taxonomy layer. Given a taxonomy  $\tilde{T}$ , a taxonomy layer  $l_{\tilde{T}}$  of  $\tilde{T}$  is the set of all types that are at the same distance from the taxonomy root. For example, the set  $\{\text{Lombardy}, \text{Tuscany}, \text{Sicily}\}$  is a layer for taxonomy  $A$  in Figure 4.2. At large scale, types occurring on same taxonomy layers are likely to be mutually exclusive since they usually represent partitions of the set of source instances categorized under the considered taxonomy. Considering co-occurrences on the same taxonomy layer represents a principled way to capture the STME principle: the more two values  $v_1$  and  $v_2$  co-occur at the same layer across all source taxonomies, the more they should be clustered together and thus the less they are distant from each other.

We compute the Taxonomy Layer Distance (TLD) between two values  $v_1$  and  $v_2$  by counting their co-occurrences on the same taxonomy layer and scaling it by their nominal occurrences across all source taxonomies. Computing TLD is equivalent to computing the Jaccard Distance between the two sets of taxonomy layers where two values  $v_1$  and  $v_2$  occur, respectively. Given a value  $v$  and a source taxonomy  $\tilde{T}$  we define the set  $L_{\tilde{T}}^v$  of layers containing  $v$  in  $\tilde{T}$  as  $L_{\tilde{T}}^v = \{l_{\tilde{T}} \mid v \in l_{\tilde{T}}\}$  (a type can occur in more than one layer). The overall set  $L^v$  of layers containing  $v$  is computed by unioning all layers across all  $n$  source taxonomies as

$$L^v = \bigcup_{i=1}^n L_{\tilde{T}_i}^v.$$



The Jaccard Distance between  $L^{v_1}$  and  $L^{v_2}$  is then computed as:

$$\text{TLD}(v_1, v_2) = 1 - \frac{|L^{v_1} \cap L^{v_2}|}{|L^{v_1} \cup L^{v_2}|}.$$

**Example.** Given the two source taxonomies from Figure 4.2 and values Cabernet and Chianti, we first compute layers containing Cabernet and Chianti

$$\begin{aligned} l_A^1 &= \{\text{Cabernet, Chianti, CantinaFirriato, CantinaAlmeria}\} \\ l_A^2 &= \{\text{Cabernet, Nero d'Avola}\} \\ l_B^1 &= \{\text{Vermentino, Cabernet, Verdicchio, Chianti}\}. \end{aligned}$$

The set of layers containing Cabernet and Chianti are

$$\begin{aligned} L_A^{\text{Cabernet}} &= \{l_A^1, l_A^2\} \\ L_B^{\text{Cabernet}} &= \{l_B^1\} \\ L^{\text{Cabernet}} &= L_A^{\text{Cabernet}} \cup L_B^{\text{Cabernet}} = \{l_A^1, l_A^2, l_B^1\} \\ L_A^{\text{Chianti}} &= \{l_A^1\} \\ L_B^{\text{Chianti}} &= \{l_B^1\} \\ L^{\text{Chianti}} &= L_A^{\text{Chianti}} \cup L_B^{\text{Chianti}} = \{l_A^1, l_B^1\}. \end{aligned}$$

Hence, the distance between Chianti and Cabernet is computed as

$$\text{TLD}(\text{Cabernet, Chianti}) = 1 - \frac{|L^{\text{Cabernet}} \cap L^{\text{Chianti}}|}{|L^{\text{Cabernet}} \cup L^{\text{Chianti}}|} = 1 - \frac{2}{3} = \frac{1}{3}.$$

---

## 4.4 Faceted Assertions Generation

---

The output of the facet extraction algorithm described in the last section is the set of extracted facets  $\mathcal{F}^C = \{F_1 = \langle C, \mathcal{V}_1 \rangle, \dots, F_n = \langle C, \mathcal{V}_n \rangle\}$  that, after the validation of domain experts, enrich the schema of the KG. However, to enable the proper publication of such facets in the front-end of the dataspace, additional information must be stored. In particular, to enable query answering after the materialization of source data into the dataspace (see, Section 2.3), a set of mappings from the sources to the newly extracted facets has to be generated.

Recall from Section 4.3.1 that values in the facet ranges are extracted from the lexicalizations of source types and normalized by applying lowerization, special characters removal and stemming. During such phase, we keep track of the (possibly) many

## 4. DOMAIN SPECIFIC FACET EXTRACTION

---

source types to which each normalized facet value correspond. To this end, we define a convenient function *origin* that takes a normalized facet value in input and returns the set of source types from which the value have been extracted. This function conceptually “reverses” the value normalization by accessing the correspondences between normalized facet values and source types stored during the value extraction phase.

Given an extracted facet  $F = \langle C, \mathcal{V} \rangle$  and the correspondences between normalized facet values and source types from which they have been extracted, mappings are generated in the following form:

$$F(x, v) \leftarrow \tilde{C}(x), \tilde{C} \in \text{origin}(v), v \in \mathcal{V}.$$

The semantics of such mappings is that a value  $v$  is asserted to be the value of the facet  $F$  for the instance  $x$ , if  $x$  is an instance of the source type  $\tilde{C}$ ,  $v$  is included in the extracted facet range and was extracted from  $\tilde{C}$ . Such mapping acts as a transformation rule used to populate the KG with a set of faceted assertions during the materialization of source data into the dataspace. Such faceted assertions will then be leveraged in the front-end of the dataspace to support, for example, a faceted search interface (see Section 2.2 for more details).

### 4.5 Evaluation

---

The core idea of our proposed approach to domain specific facet extraction is that we group facet values according to a structural criterion (i.e., TLD). Hence, we focus on evaluating the facet value clustering phase. Our goal is to show that TLD effectively captures the STME principle and supports domain experts in facets definition. To the best of our knowledge there are no distance metrics for taxonomies that explicitly aim at capturing the STME principle. However, structural similarity metrics that consider path distance between types of a taxonomy are good candidates to compare our work to. Intuitively, the more two source types co-occur in the same source taxonomy path (i.e., they are similar to some degree according to structural similarity metrics) from the root to a leaf, the less they are mutually exclusive and the more they should be clustered into different facet ranges (i.e., the clustering algorithm should consider them distant from each other).

In the experiments described in this section, TLD is compared with two known structural type similarity metrics, namely Leacock and Chodorow [68] (LC) and Wu and Palmer [150] (WP) metrics. Both LC and WP achieve high effectiveness results in

determining the similarity of concepts within the WordNet taxonomy [118]. LC measures the similarity between two taxonomy types by considering the shortest path between them and scaling it by the depth of the taxonomy. Similarly, WP measures the similarity between two types by considering the distance from their nearest common ancestor and the distance of the nearest common ancestor from the taxonomy root. We adapted LC and WP to the case of multiple taxonomies. More specifically, given two source types  $\tilde{C}$  and  $\tilde{D}$  we evaluate their LC and WP similarities for each source taxonomy where  $\tilde{C}$  and  $\tilde{D}$  co-occur and we take the mean similarity as the final distance value.

#### 4.5.1 Gold Standard

We created a gold standard from the real world TrovaPrezzi Italian PCE dataspace. We chose ten TrovaPrezzi global categories and ran the Values Extraction phase over them. We presented the set of top  $k$  frequent values to TrovaPrezzi domain experts, who found that relevant facet values generally appear among the top 100 ranked values. Thus we choose  $k = 100$  as cardinality of the set of extracted facet values. Facet values were manually grouped together by a domain expert from TrovaPrezzi mapping team and facets were then validated by other domain experts in order to ensure their correctness. As we expected, some of the values were discarded by domain experts as they could be added to any existing facet range.

Table 4.1 shows some statistics about the Gold Standard. Gold Standards' KG types cover different domains and a relevant portion of the overall KG of the CSE, that is 688 source taxonomies and 22594 leaf mappings. For each source taxonomy an average of about 33 mappings have been specified. Moreover, for 322 source taxonomies mappings to more than one KG type have been specified. Notice that all the data upon which we created the gold standard are lexicalized in Italian. Our approach to domain specific facet extraction is language independent, thus results that we present in following sections are comparable to others obtained considering different languages. For sake of clarity, we provide examples translated to English.

#### 4.5.2 Evaluation Metrics

We evaluate our facet extraction approach from two different perspectives: facet value effectiveness and value clustering effectiveness. This kind of evaluation campaign

## 4. DOMAIN SPECIFIC FACET EXTRACTION

Table 4.1: Gold Standard statistics.

	#Source Taxonomies	#Mappings	Mean	Std.Dev.
Dogs and Cats Food	80	5592	69.90	266.95
Grappe, Liquors, Aperitives	115	1254	10.90	24.49
Wines	184	8967	48.73	324.63
Beers	58	156	2.69	3.84
DVD Movies	164	2042	12.45	45.48
Rings	138	936	6.78	13.61
Blu-Ray Movies	55	395	7.18	16.67
Musical Instruments	128	1306	10.20	27.49
Ski and Snowboards	55	790	14.36	23.45
Necklaces	148	1156	7.81	17.88
<b>Overall</b>	<b>688</b>	<b>22594</b>	<b>32.84</b>	<b>195.79</b>

has been previously used to evaluate several facet extraction approaches [60, 33]. We introduce the notation we will use in the rest of the section. Given a KG type  $C$ , we denote with  $V^C$  the set of discovered facet values (i.e., values that have not been classified as noise by the algorithm). We denote with  $V_*^C$  the set of gold standard facet values (i.e., values not classified as noise by domain experts). Lastly, we denote with  $\mathcal{V}^C$  the set of manually discovered facet ranges (i.e., the gold standard for the KG type  $C$ ), which is compared to the set  $\mathcal{V}_*^C$  of automatically discovered facet ranges.

### Value Effectiveness

In our proposed approach noisy values are discarded. In order to evaluate the ability of our technique to filter noisy values out we compare sets  $V^C$  and  $V_*^C$ , using Precision ( $P$ ), Recall ( $R$ ) and F-Measure ( $F_1$ ). All these metrics do not take clustering effectiveness into account.

### Value Clustering Effectiveness

We evaluate clustering effectiveness using several standard clustering quality metrics, that are Purity ( $P^*$ ), Normalized Mutual Information ( $NMI^*$ ), Entropy ( $E^*$ ), and F-Measure for clustering ( $F^*$ ). One remark about the usage of these evaluation metrics is that the set of facet values clustered by our approach is different from the set of facet values grouped by humans (i.e.,  $\mathcal{V}^C \neq \mathcal{V}_*^C$ ). We may fail in including meaningful values into some clusters, or we may mistakenly include noisy values into some facets. Clustering quality metrics cannot handle these cases. Thus, we modify facet ranges

in  $\mathcal{V}^C$  by (1) removing all noisy values and by (2) adding to  $\mathcal{V}^C$  as single value facet ranges all gold standard values that have been automatically classified as noise. These adjustment ensures that  $\mathcal{V}^C = \mathcal{V}_*^C$  and thus clustering quality metrics can be used properly. With this adjustment, facet value effectiveness is not considered.

### Overall Quality

In order to evaluate the overall effectiveness of our approach, we aggregate facet value precision  $P$ , facet value recall  $R$  and clustering F-measure  $F^*$  into an overall quality measure. The  $PRF^*$  measure combines  $P$ ,  $R$  and  $F^*$  by means of an harmonic mean:

$$PRF^* = \frac{3 * P * R * F^*}{R * P + P * F + P * R}.$$

#### 4.5.3 Experimental Results

We conducted several experiments, comparing clustering performance of TLD, LC and WP metrics. We recall from Section 4.3.2 that the DBSCAN algorithm used for clustering is configured with a maximum distance threshold  $\epsilon$ . Optimal values of  $\epsilon$  depend on the used distance metric, and influence clustering performance. The tuning of  $\epsilon$  can be driven by two orthogonal factors: overall quality (i.e.,  $PRF^*$ ) and the number of discovered clusters. High values of  $\epsilon$  (i.e., quality oriented configuration) can lead to better overall quality, but fewer discovered clusters (i.e., the clustering algorithm will tend to group values into one single cluster). Lower values of  $\epsilon$  (i.e., cluster number oriented configuration) can lead to lower quality, but more discovered clusters. We found that quality oriented and cluster number oriented configurations generally coincide except for LC. In the following section we refer to quality oriented configuration of LC as  $LC_q$  while we indicate with  $LC_n$  the corresponding cluster number oriented configuration. Since optimal configurations for WP and TLD coincide we omit pedices for them.

Table 4.2 presents results of our experiments. TLD is more effective in finding relevant facet values and discarding noisy ones, as indicated by an higher  $F_1$ . The ability of effectively discarding noisy values substantially reduces domain experts' effort in validating discovered facets.  $LC_q$  and WP obtain almost perfect value recall, but substantially lower precision. Thus, they do not effectively support domain experts. Moreover, TLD achieves best performance according to quite all clustering effectiveness metrics, with the exceptions of purity and entropy for  $LC_n$ . Clusters discovered

#### 4. DOMAIN SPECIFIC FACET EXTRACTION

Table 4.2: Effectiveness of TLD, LC and WP metrics.

	Value Effectiveness			Clustering Effectiveness			Quality	
	$P$	$R$	$F_1$	$F^*$	$NMI^*$	Purity		$E^*$
$LC_n$	0.359	0.447	0.370	0.403	0.603	<b>0.308</b>	<b>0.243</b>	0.359
$LC_q$	0.394	0.953	0.537	0.666	0.709	0.220	0.685	0.531
WP	0.377	<b>0.984</b>	0.525	0.682	0.714	0.210	0.744	0.520
TLD	<b>0.416</b>	0.901	<b>0.541</b>	<b>0.719</b>	<b>0.746</b>	0.286	0.416	<b>0.558</b>

Table 4.3: Number of groups discovered by TLD, LC, and WP for each source type, compared to the gold standard.

	$ \mathcal{V}_*^C $	$LC_q$	$LC_n$	WP	TLD
Dogs and Cats Food	3	1	5	1	7
Grappe, Liquors, Aperitives	1	1	5	1	6
Wines	3	1	1	1	6
Beers	2	6	4	3	14
DVD Movies	2	2	3	1	3
Rings	4	1	6	2	7
Blu-Ray Movies	2	2	3	2	5
Musical Instruments	6	1	3	1	5
Ski and Snowboards	1	1	3	1	7
Necklaces	8	2	6	3	11

by  $LC_n$  contain more homogeneous values, in the sense that they have been manually classified as belonging to the same gold standard group. However,  $LC_n$  achieves better purity and entropy at the cost of discarding most of the values as noise, thus sacrificing overall quality.

The difference between TLD and state-of-the-art metrics is even more evident if we consider the number of detected clusters for each gold standard type  $C$  (Table 4.3). WP and  $LC_q$  fail in properly partitioning the overall set  $V_k^C$  of facet values, thus failing in detecting groups (i.e., they detect only one or two clusters). They are too inclusive and thus they group facet values at a granularity level that is too high to be suitable for effectively supporting domain experts in bootstrapping a faceted classification system within the dataspace. From the other side,  $LC_n$  discards too much values to be effective.

In addition to standard evaluation metrics, we provide a more intuitive insight of results of the facet extraction process, using TLD for facet value clustering compare to state-of-the-art metrics. Table 4.4 depicts an example of facets ranges discovered for the KG type Wines by TLD, WP,  $LC_q$ , and  $LC_n$  compared to manually defined ones.

Table 4.4: Discovered ranges of domain specific facet ranges for TLD, WP and LC compared to manually discovered ones. Numbers after facet ranges indicate their cardinality.

$LC_q$	$\mathcal{V}_1 = \{\text{Wine, Red Wine, White Wine, } \dots, \text{ Piedmont, Lombardy, } \dots, \text{ Sicily, Donnafugata, Cusumano, } \dots, \text{ Alessandro di Camporeale, } \dots, \text{ France}\}$ (98)
$LC_n$	$\mathcal{V}_1 = \{\text{Wine, Red Wine, White Wine, } \dots, \text{ France, } \dots, \text{ Chianti}\}$ (36)
WP	$\mathcal{V}_1 = \{\text{Wine, Red Wine, White Wine, } \dots, \text{ Piedmont, Lombardy, } \dots, \text{ Sicily, Donnafugata, Cusumano, } \dots, \text{ France}\}$ (100)
TLD	$\mathcal{V}_1 = \{\text{ Piedmont, Tuscany, Sicily, } \dots, \text{ France}\}$ (14) $\mathcal{V}_2 = \{\text{ Red, White, Rosé}\}$ (3) $\mathcal{V}_3 = \{\text{ Red Wine, White Wine, Rosé Wine}\}$ (3) $\mathcal{V}_4 = \{\text{ Moscato, Chardonnay, } \dots, \text{ Merlot}\}$ (13) $\mathcal{V}_5 = \{\text{ Tuscany Wine, Sicily Wine}\}$ (2) $\mathcal{V}_6 = \{\text{ Donnafugata, Cusumano, } \dots, \text{ Principi di Butera}\}$ (27)
Gold Standard	$\mathcal{V}_1 = \{\text{ Piedmont, Lombardy, } \dots, \text{ Sicily}\}$ (21) $\mathcal{V}_2 = \{\text{ Red Wine, White Wine, } \dots, \text{ Rosé Wine}\}$ (14) $\mathcal{V}_3 = \{\text{ Donnafugata, Cusumano, } \dots, \text{ Alessandro di Camporeale}\}$ (12)

Validating and refining ranges discovered by TLD requires much less domain experts' effort than  $LC_q$ ,  $LC_n$  and WP.

Table 4.4 highlights a difficulty of TLD in grouping together different lexicalizations of same values (e.g., Red Wine and Red). One naive approach to overcome this difficulty is to normalize source type names by removing terms belonging to the KG types for which the facets are extracted (e.g., the term Wine when extracting facets to characterize instances of the KG type Wines). However, this naive solution cannot be generalized to every KG type. For example, if we remove from the source type Dog Food all the terms belonging to the gold standard KG type Dogs and Cats Food we end up with an empty, inconsistent facet value. Moreover, also the more conservative approach of removing KG type terms only if they *all* occur in the source type cannot be generalized. For example, if we consider the gold standard type Musical Instruments, using the more conservative approach we will not normalize source types Wind Instruments and Winds.

We implemented and evaluated both the previously described naive solutions, and found that they both decrease the effectiveness of our approach. We believe that effectively solving the problem of different lexicalizations requires Natural Language Processing language specific techniques. NLP techniques can be used to discriminate between KG types terms that refer to nouns, verbs, etc. and thus can be safely removed from source types without creating inconsistencies or change types lexicalizations' semantics. Introducing this kind of NLP language specific techniques comes at the cost

## 4. DOMAIN SPECIFIC FACET EXTRACTION

---

of sacrificing the language independence of our approach. However, this represents an interesting extension of our approach.

### 4.6 Comparison with Prior Art

---

Different approaches to the problem of extracting a domain specific representation of instances (in terms of relations and facets) from heterogeneous data sources have been proposed (see Section 3.1). Facets are usually extracted from a *document collection* (e.g., [32, 130, 82, 146]), from search engine *query results* (e.g., [151, 60, 33, 55]) or from the combination of documents and search engine *query logs* (e.g., [110, 100, 73]). Document collection based approaches tackle the problem of extracting facets from a document collection. External resources such as WordNet [32, 130], Wikipedia [32, 146] or its Linked Data version DBpedia [82] are exploited to enrich the extracted set of facets values. Our approach is different from document collection based ones because we analyze source taxonomies structure in order to provide sets of mutually exclusive facet values.

The focus of query result based approaches is on classification of HTML documents returned by a data source in response to a keyword query search. Facets are extracted from Wikipedia documents [151] analyzing, among other things, Wikipedia categories and reciprocal links between documents. In more general approaches facets are extracted by analyzing raw HTML pages in using unsupervised [33] or supervised [60] machine learning techniques. State of art query result based approaches deal with the specific problem of integrating and ranking heterogeneous facets that are already present in documents. Our approach takes in input source taxonomies and mappings between them and KG types.

The focus of query logs approaches is on the usage of user query statistics to identify facet values. Query logs are analyzed in order to select relevant facet values with respect to closed, fixed [110] or open, not defined a-priori [100, 73] set of facets. Query log based approaches are strictly dependent on end-user queries: they do not consider currently available dataspace instances. Our approach analyzes source types and taxonomies, and thus provide a more comprehensive and domain specific dataspace instances representation. All the previously described approaches are complementary to ours. We extract facets from a different source than previous approaches: taxonomies used to characterize source instances. We expect the facet extraction process to benefit from the integration of state-of-the-art facet extraction techniques.



Taxonomy structure analysis has been employed in the field of Ontology Matching [122]. In this field, several similarity metrics between ontology (and also taxonomy) concepts have been proposed and/or adapted from other domains [26, 68, 150, 81]. However, experimental results provided in this chapter show that our proposed distance metric (i.e., TLD) is more effective in capturing the mutual exclusiveness of concepts across multiple heterogeneous taxonomies.

## 4.7 Summary

---

This chapter discussed an automatic, language independent approach to the problem of facets extraction from heterogeneous taxonomies. We proposed a novel metric designed ad-hoc to capture source type mutual exclusivity across taxonomies. We used the proposed metric as a clustering distance metric for grouping together mutual exclusive facet values. Experimental results show that our approach outperforms state-of-the-art taxonomy concepts similarity metrics in capturing types mutual exclusiveness. Our approach provides valuable aid and reduces domain experts' effort in enriching the KG schema with domain specific facets. In addition, since our approach extracts facets from source types, the population of the KG with the corresponding faceted assertions for extracted facets over dataspace instances is straightforward, supporting facet-based browsing.



# 5

## Specificity-based Interpretation of Facets

### 5.1 Overview

---

In the last chapter we discussed an automatic facet extraction approach that supports domain experts in the extraction of facets. As for most of state-of-the-art facet extraction approaches (e.g., the ones discussed in Section 3.1) the approach presented in the last chapter does not take into account the interpretation of a extracted facets, in the sense that it is not able to understand *which* relation is modeled by the facet and provide a machine-readable semantics associated with it. In order to fully exploit the powerful domain specific representation potentially provided by facets, a more complete interpretation of their semantics is needed.

This chapter presents an automatic approach to interpret a facet by *annotating* it with relations holding between KG instances of given *domain* and facet values. The key insight behind the approach discussed in this chapter is to re-use relations defined by in a KG to interpret the meaning of a facet. Given a facet comprised of a set of facet values and a facet domain, the proposed approach derives a set of candidate relations from KG and ranks them considering how much their semantics is similar to the semantics of the facet. The core of the approach is the notion of semantic similarity. Our hypothesis is that we can assess the similarity between a facet and a relation  $P$  by considering the extensional semantics of  $P$ , that is the usage of  $P$  in KG with respect to the different domains of knowledge conceptualized by KG types.

Solving the facet annotation problem is challenging for several reasons. Facet values are ambiguous and highly domain dependant, and thus there may be more than one relation suitable for the annotation of a given facet. For example, facet values such

## 5. SPECIFICITY-BASED INTERPRETATION OF FACETS

---

as English, German, French, Spanish may refer to the language of a specific book or to the nationality of a basketball player. So as to leverage the extensional semantics of the relations provided by KG, both facet values and domain must be matched with relational assertions and types that are used to classify entities from KG. We address this challenge by providing a convenient representation of KG that supports the matching of facet values with relational assertions from KG, and the matching of the facet domain with types. Moreover, we design a ranking function that is able to effectively quantify semantic similarity.

The rest of this chapter is organized as follows. Section 5.2 introduces the facet annotation problem, while Section 5.3 defines the semantic similarity between a facet and a KG relation. In Section 5.4, we describe a filter and rank facet annotation approach that, given a facet, selects relations from a KG and ranks them according to their similarity. Section 5.5 describes an extensive evaluation using KGs with different characteristics, along with the comparison of the proposed facet interpretation approach with state-of-the-art ones, which are described in Section 5.6 before concluding the chapter in Section 5.8<sup>1</sup>.

### 5.2 Problem Definition

---

Our approach considers facets and a KG. In the following paragraph we briefly recall the definition of these two kind of artifacts from Chapter 2, and the facet annotation problem. We then highlight the basic intuition behind our proposed approach.

#### Knowledge Graph

A Knowledge Graph (KG) consists of a set of named *relations* holding between named *instances* classified using *types*. A relation  $P$  is specified in terms of *domain* and *range* restrictions expressed by means of First Order Logic (FOL) terminological axioms. Such domain and range axioms typically impose a restriction on the type of subject or object instances of a relation, respectively. We refer to this specification as the *intensional* semantics of a relation (see Section 2.1.1). The KG includes also a set of relational assertions  $\mathcal{A}^P$  in the form of  $P(s, o)$ . A relational assertion explicitly states that the  $P$  holds

---

<sup>1</sup>The approach described in this chapter, which is the result of the most recent work carried out as part of PhD activities of the author, is planned to be submitted within May or June 2016 at most. A version of the working paper is available at <http://bit.ly/facet-annotation-paper-draft>

between a subject instance  $s$  and an object instance  $o$ . Relational assertions provide an alternative specification of the semantics of  $P$  emerging from its usage within KG. We refer to this specification as *extensional* semantics, as opposed to the *intensional* semantics specified by the KG schema.

Types are organized in subtype graph  $G = (\mathcal{N}^C, \preceq)$ , where  $\mathcal{N}^C$  is the set of all KG types, and  $\preceq$  is a subtype relation that holds between them. Observe that in principle a KG may specify more than one subtype graphs (e.g., the DBpedia KG). An instance  $e$  is an instance of one or more types. We define a function  $types(e)$  that returns the set of types of  $e$ , including all the supertypes through the subtype graph (i.e., the transitive closure of the types' set). Both types, and entities are associated to *lexicalizations*. A lexicalization is a sequence of natural language tokens that are associated to a type, entity, or a literal value. We denote with  $lex(C)$ ,  $lex(e)$  the set of lexicalizations of a type  $C$ , an entity  $e$ , respectively. The lexicalizations of a literal value is the set that contains the value itself. For instance, considering Figure 5.1:  $lex(J\_R\_R\_Tolkien) = \{“John Ronald Reuel Tolkien”, “J. R. R. Tolkien”\}$ .

### Facet

Recall from Section 2.2 that a facet is a relation that models a salient characteristic of a specific type of entities and a set of values. A facet  $F = \langle C, \mathcal{V} \rangle$  holds between a *domain* of instances of the type  $C$  and a *range* of facet values  $\mathcal{V} = \{v_1, \dots, v_n\}$ . For instance, the facet  $F = \langle \text{Books}, \{\text{Agatha Christie}, \text{Arthur C. Clarke}, \dots\} \rangle$  is depicted in Figure 5.1. Observe that in this setting, the facet domain  $C$ , is expressed in terms of a source type of entities. We assume this type (e.g., Books) to be available at least in the form of its lexicalization. We notice also that the specification of facet is not “balanced”, in the sense that the facet domain is defined differently from the facet range, being the first a lexicalized type and the second a set of values, respectively.

### Facet Annotation

Annotating a facet  $F$  means finding a relation  $P$  or more generally a set  $\mathcal{P} = \{P_1, \dots, P_n\}$  of relations from KG, such that the semantics of  $P$  is *similar* to the semantics of  $F$ . Figure 5.1 shows a graphical depiction of the facet annotation problem. In essence, solving the facet annotation problem consists of assessing the degree of similarity between the semantics of a relation  $P$  from KG and a facet  $F$ .

## 5. SPECIFICITY-BASED INTERPRETATION OF FACETS

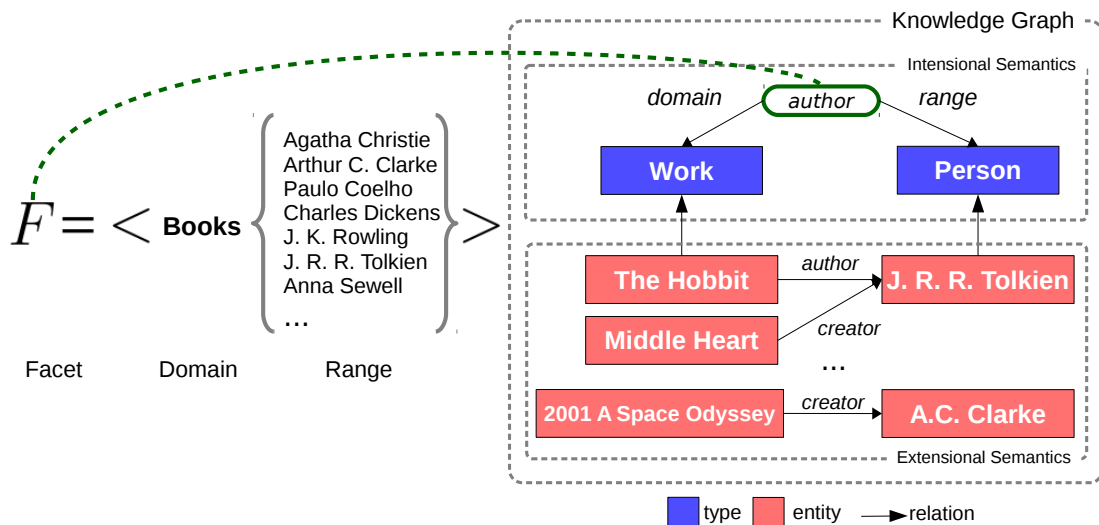


Figure 5.1: The facet annotation problem.

Solving the facet annotation problem basically accounts to solving a relation annotation (i.e., interpretation) problem [74, 143, 120, 145, 112, 156, 86, 154, 153, 155, 27]. The goal of relation annotation approaches (see Section 3.2), is to annotate an input source relation with target relations from a defined KG. However, there is one crucial difference between a relation annotation problem and a facet annotation problem. In fact, the specification of a facet is *unbalanced*, in the sense that the facet domain is specified differently from the facet range. The facet domain is specified by means of a lexicalization of an entity type, while the facet range is specified by means of a set of values. This setting is different from typical relation annotation settings, which deal with relations whose specification is *balanced*. Relation domains and ranges are either both specified in terms of sets of values (i.e., for relation annotation in web table interpretation tasks [74]) or in terms of entity types (i.e., for relation annotation in ontology matching tasks [27]).

### 5.3 Specificity Driven Semantic Similarity

For seek of simplicity, we write that a relation  $P$  is *similar* to a facet  $F$ , meaning that their semantics are similar. In a nutshell, we consider a relation  $P$  similar to a facet  $F$  if  $P$  is: (1) *specific* with respect to the facet domain, (2) it *covers* all the facet values, and (3) it *frequently* connects entities from the facet domain to facet values within KG. Given a facet  $F$ , our intuition is to quantify *specificity*, *coverage*, and *frequency* of relations, by

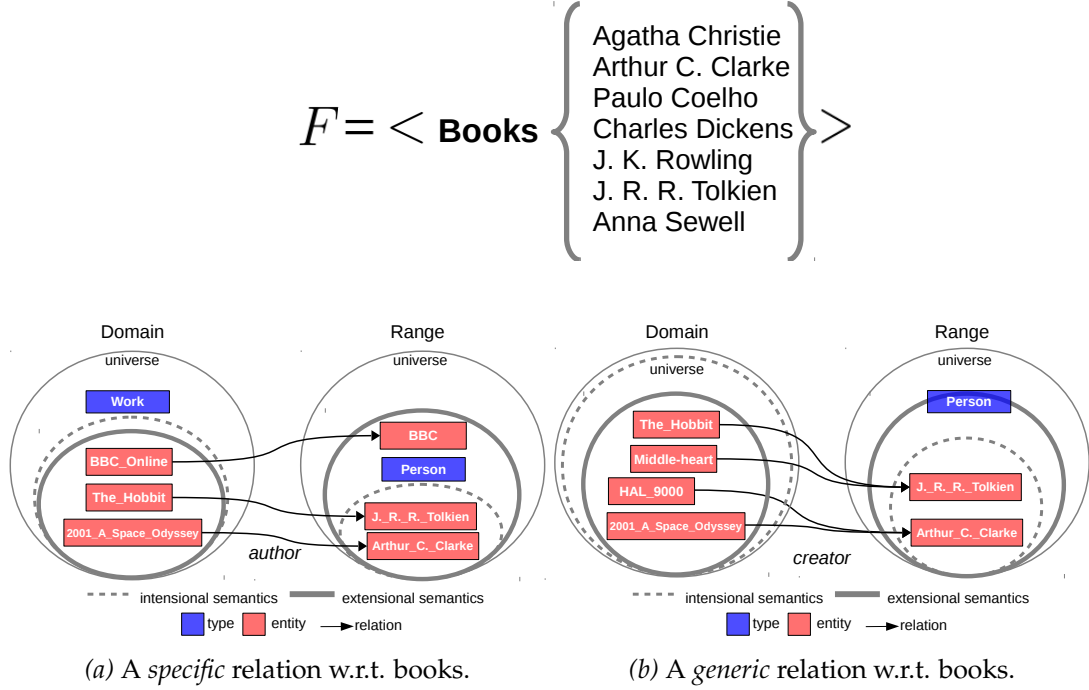


Figure 5.2: An example that illustrates the difference between specific and generic relations.

looking at their extensional semantics, that is the usage in KG.

The core concept on which we base our definition of semantic similarity is *specificity*. We argue that the more a relation  $P$  is specific with respect to the domain and the range of a facet  $F$ , the more  $P$  is similar to  $F$ . We rely on an example to better explain why we consider it crucial in determining the similarity between a facet and a relation<sup>2</sup>. Consider the facet from Figure 5.2. The facet range includes, for instance, Arthur\_C.\_Clarke, and J.\_R.\_R.\_Tolkien, while the facet domain is Books. Intuitively, the facet holds between books and their authors. Now, consider the two relations author (Figure 5.2a) and creator (Figure 5.2b). Both relations hold between The\_Hobbit and J.\_R.\_R.\_Tolkien, and between 2001\_A\_SpaceOdyssey and Arthur\_C.\_Clarke. From this view point they both look similar to  $F$  as they hold between the same entities.

However, if we take a look at the intensional semantics of author and creator, we realize that the domain of the latter is completely unspecified: any entity can potentially be part of such a domain. Moreover, looking at the extensional semantics of creator provided by KG, we realize that creator holds not only between books and writers, but also between fictional characters and places and their respective creators.

<sup>2</sup>The presented example is drawn from the DBpedia KG (version 3.9), with some simplifications.

## 5. SPECIFICITY-BASED INTERPRETATION OF FACETS

---

Observed from this viewpoint, author looks much more similar than creator to  $F$ . In fact, the domain of author is much more specific with respect to the facet domain (i.e., Books), compared to the domain of creator.

Our definition of semantic similarity includes also *coverage*. Ideally, given  $F$ , a relation  $P$  that holds between entities from the domain  $C$  and *all* facet values should be considered highly similar to  $F$ . For instance, given the facet from Figure 5.2, suppose that we find that the relation author holds between instances of type Book and all the books denoted by facet values. In this situation, we consider author similar to the facet, because all the facet values appear within instances of author (i.e., the “coverage” of author over the facet values is maximum).

*Frequency* is orthogonal to coverage. In principle, given  $F$ , the more frequently facet values are connected by a relation  $P$  to instances from the domain  $C$ , the higher the similarity between  $P$  and  $F$ . The rationale behind frequency is to favour relation that are more frequently used within KG with respect to the facet domain, for instance favoring relations like language over origLanguage<sup>3</sup>.

### 5.4 Facet Annotation

---

The intuition underlying our facet annotation approach is to consider the extensional semantics of relations as a source of *evidence* on the similarity between  $F$  and a relation  $P$ . As a principle, we may consider a relational assertion  $P(s, o)$  as positive evidence on the similarity between  $F$  and  $P$  if (1)  $s$  is an instance of the same type described by the facet domain  $C$ , and (2)  $o$  is equal to some facet value  $v \in \mathcal{V}$ . However, the concrete application of this principles is challenging because (1) both the facet domain and facet values must be matched with types and relation objects from KG, and (2) we cannot assume KG to completely cover the domain of knowledge conceptualized by  $F$ . In fact in this situation, it may be that KG only partially represents the domain of knowledge conceptualized by a facet (i.e., some book authors may not correspond to any entity in KG). Thus, although KG may be a source for positive evidence, we cannot consider it as a source of negative evidence whenever the facet domain cannot be matched to any KG type, or some facet values cannot be matched to any object. For this reason, our approach relies on positive evidence only in order to quantify the similarity between  $F$  and  $P$ . We assume the world conceptualized by KG to be *open* in the sense that we admit the annotation of  $F$  with  $P$ , even when the facet domain partially matches

---

<sup>3</sup>Both of them are used in DBpedia to denote the original language of publication of books.



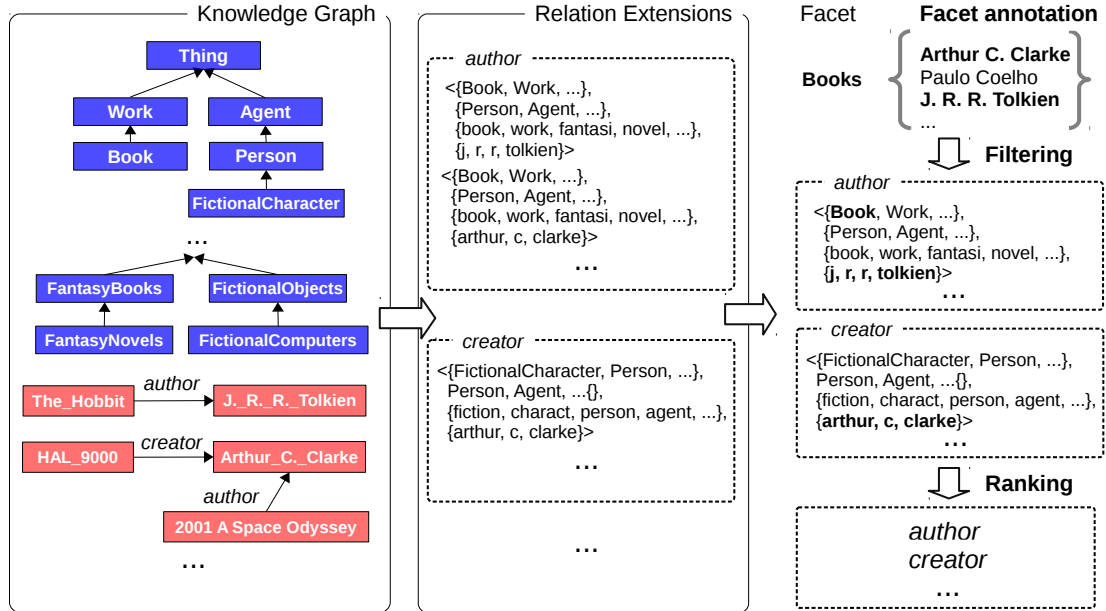


Figure 5.3: The Facet Annotation process.

KG types (i.e., players vs. basketball players) facet values only overlap with relation objects.

Our facet annotation approach is depicted in Figure 5.3 and its composed by on two distinct phases: relation *filtering* and relation *ranking*. Given a facet  $F$ , we first match the facet domain and the facet values to relational assertions  $P(s, o)$  from KG, so as to gather evidence from KG on the similarity between  $F$  and KG relations. In this phase, the facet domain and values are matched against a convenient representation of relational assertions, that we name *relational assertion signature*. Signatures support the matching between the facet domain and the types of  $s$ , and between facet values and objects  $o$ . The result of the filtering phase is a set of matching signatures for each KG relation  $P$ , to which we refer as the *extensions of  $P$  induced by  $F$* . Then, in the ranking phase, we analyze the non empty extensions of relations induced by  $F$  in order to quantify the degree of similarity between each relation and  $F$ , adhering to principles described in Section 5.3: *specificity*, *coverage*, and *frequency*.

### 5.4.1 Relation Filtering

So as to support the filtering phase, we represent each relational assertion in KG by means of a *relational assertion signature* (*signature* for brevity). A signature  $\delta_{(s,o)}$  is an

## 5. SPECIFICITY-BASED INTERPRETATION OF FACETS

---

*abstract* and *lexicalized* representation of a relational assertion  $P(s, o)$ . It is *abstract* as it includes all the types of the subject  $s$  as well as the types of the object  $o$ . It is *lexicalized* as it includes the lexicalization of  $o$  as well as the lexicalization of *all* the types of  $s$ . More formally, a signature  $\delta_{(s,o)}$  is defined as:

$$\delta_{(s,o)} = \langle \text{types}(s), \text{types}(o), \overline{\text{lex}}(s), \text{lex}(o) \rangle \quad (5.1)$$

where  $\text{types}(s)$ ,  $\text{types}(o)$  are the set of types of the subject and the object, respectively and  $\text{lex}(o)$  is the lexicalization of the object. Special attention must be posed to the set of subject types lexicalizations  $\overline{\text{lex}}(s)$ , which is defined as the union of the lexicalizations of *all* the types of the subject. Formally:

$$\overline{\text{lex}}(s) = \bigcup_{t \in \text{types}(s)} \text{lex}(t). \quad (5.2)$$

Through abstraction we explicitly record that *there exists* a relationship that holds between instances of certain types. In other words, we adopt an existential semantics for signatures. Such semantics will be leveraged to quantify the *specificity* of a relation with respect to the facet domain. Through lexicalization we describe subject types and objects of relational assertions with a set of natural language terms, that are matched to facet domain  $C$  and facet values from  $\mathcal{V}$ . This allow us to capture the *coverage* and the *frequency* principles. Signatures are indexed and stored in a convenient inverted index, so as to support the filtering phase, where relation extensions are selected by matching the facet with signatures.

We represent the extensional semantics of a relation  $P$  in terms of the previously defined signatures. In particular, we represent the *extension of a relation*  $P$  as the set  $\Delta_P$  of signatures  $\delta_{(s,o)}$  of all relational assertions from the assertion set  $\mathcal{A}$  of KG involving  $P$ . Formally:

$$\Delta_P = \{\delta_{(s,o)} \mid P(s, o) \in \mathcal{A}\}. \quad (5.3)$$

Given  $F$  and a matching function *match* between a facet and a signature, in the *relation filtering* phase, we form the *extension of a relation*  $P$  induced by  $F$  by selecting the set of signatures of  $P$  that match  $F$ . Formally:

$$\Delta_P^F = \{\delta_{(s,o)} \in \Delta_P \mid \text{match}(F, \delta_{(s,o)})\}. \quad (5.4)$$

$\Delta_P^F$  stores the amount of *positive* evidence that can possibly be found in KG on the similarity between  $F$  and  $P$ . Evidence is collected by matching the facet to signatures.

The criteria according to which signatures are considered to be positive evidence are encoded in the *match* boolean matching function. Our approach is based on the combination of two different matching functions, focusing on matching the facet domain and the facet values, respectively. The *d-match* function selects the signatures whose subject types match the facet domain. In order to accomplish this task we consider the lexicalizations of the subject types  $\overline{lex}(s)$  and compare it with the lexicalization of the facet domain  $lex(C)$ :

$$d\text{-match}(F, \delta_{(s,o)}) \text{ is true iff } lex(C) \cap \overline{lex}(s) \neq \emptyset. \quad (5.5)$$

Orthogonally, *r-match* function focuses on selecting those signatures whose object matches *at least one* facet value:

$$r\text{-match}(F, \delta_{(s,o)}) \text{ is true iff } \exists v \in \mathcal{V} \text{ s.t. } lex(v) \subseteq lex(o). \quad (5.6)$$

We then combine *d-match* and *r-match* together so as to select the signatures whose subject types match the facet domain *and* the object matches at least one facet value. More formally:

$$\Delta_P^F = \{\delta_{(s,o)} \in \Delta_P \mid d\text{-match}(F, \delta_{(s,o)}) \wedge r\text{-match}(F, \delta_{(s,o)})\}. \quad (5.7)$$

In general, the higher the cardinality of  $\Delta_P^F$ , the more positive evidence can be found in KG on the similarity between  $F$  and  $P$ . Conversely, when  $\Delta_P^F = \emptyset$ , we can conclude that  $F$  and  $P$  are not similar, because there is no evidence in KG on their similarity. This relation allows us to prune a relation  $P'$  when  $\Delta_{P'}^F = \emptyset$ . Given  $F$ , we query the indexed signatures and form the set  $\mathcal{P} = \{P_1, \dots, P_n\}$  of relations with a non-empty relation extension  $\Delta_{P_1}^F, \dots, \Delta_{P_n}^F$ .

## 5.4.2 Relation Ranking

In the *relation ranking* phase, extensions selected in the filtering phase are analyzed so as to assess the similarity between  $F$  and relations. We encode all the principles described in Section 5.3 (i.e., *specificity*, *coverage*, *frequency*) into a set of scores that will be aggregated into an overall semantic similarity score. Finally, we rank the relations in  $\mathcal{P}$  accordingly.

### Specificity

We capture the *specificity* of a relation  $P$  with respect to  $F$  by looking at the extension  $\Delta_P$  and compare them with the extension  $\Delta_P^F$  induced by  $F$ . If  $\Delta_P^F$  is very “close” to

## 5. SPECIFICITY-BASED INTERPRETATION OF FACETS

---

$\Delta_P$ , we may consider the semantics of  $P$  to be somehow preserved when annotating  $F$  with  $P$ . Of course, it is crucial to provide a definition of what we mean by “close”. In principle one may apply a majority voting inspired approach, that is the more  $\Delta_P^F$  and  $\Delta_P$  overlap (i.e., more “votes” for  $P$ ), the more we can consider  $P$  similar to  $F$ . However, this majority-based approach is likely to fail in capturing the specificity of  $P$  with respect to  $F$ .

The cardinality of set of facet values  $\mathcal{V}$  is usually several orders of magnitude smaller compared to the cardinality of the set of possible objects of a given relation  $P$ . As a result,  $|\Delta_P^F| \ll |\Delta_P|$ , thus making the direct comparison of the two extensions not discriminative enough. Moreover,  $\Delta_P^F$  is the result of a matching algorithm and it may include signatures resulting from false positive matches. To account for these issues we consider types instead of signatures. We compare the sets of subject and object types extracted from the signatures in  $\Delta_P$  with the ones extracted from signatures in  $\Delta_P^F$ . We follow the intuition that the more these sets are similar, the more  $\Delta_P^F$  is “close” to  $\Delta_P$  and thus the semantics of  $P$  is preserved. We consider this a strong positive clue of the similarity between  $P$  and  $F$ .

Given a generic extension  $\Delta$ , we form the subject type set  $Dom[\Delta]$  by selecting all the types of all the subjects from signatures in  $\Delta$ . More formally, the subject type set  $Dom[\Delta]$  of an extension  $\Delta$  is defined as

$$Dom[\Delta] = \{C \mid \exists \delta_{(s,o)} \in \Delta, C \in types(s) \wedge \nexists C' \in types(s), C' \preceq C\}. \quad (5.8)$$

Observe that  $Dom[\Delta]$  does not contain *all* the subject types. Instead, it contains all the types that are *minimal*. In general, a type  $C$  is minimal for an instance  $e$  if there are no other types in  $types(e)$  that are subtypes of  $C$ .

By considering the minimal types only, we enforce  $Dom[\Delta]$  to provide a view of the specificity of the domains of  $\Delta$ , as  $Dom[\Delta]$  contains the most *specific* types that are used to categorize the subjects of signatures from  $\Delta$ . We then capture the *domain*-specificity of a relation  $P$  with respect to  $F$  by comparing the subject type sets extracted from the two extensions  $\Delta_P$  and  $\Delta_P^F$ . As a principle, we consider  $P$  specific with respect to  $F$  if their extensions contain the same set of specific subject types. More formally, we compare  $Dom[\Delta_P]$  and  $Dom[\Delta_P^F]$  using the well known weighted Jaccard similarity for sets:

$$d-spec(F, P) = \frac{w-card(Dom[\Delta_P] \cap Dom[\Delta_P^F])}{w-card(Dom[\Delta_P] \cup Dom[\Delta_P^F])} \quad (5.9)$$

where the *w-card* function provides a weighted cardinality of the subject type set  $Dom[\Delta]$ . Each type  $C \in Dom[\Delta]$  is weighted considering the cardinality of the set

of its supertypes (i.e., depth) according to the subtype graph. We furthermore scale the weight of each type  $C$  by the maximum depth of all the subtypes of  $C$  to ensure weights within the  $[0, 1]$  interval. More formally:

$$w\text{-card}(\text{Dom}[\Delta]) = \sum_{C \in \text{Dom}[\Delta]} \frac{|\text{supertypes}(C)|}{\max_{C' \in \text{subtypes}(C)} |\text{supertypes}(C')|}. \quad (5.10)$$

The rationale of using this scaled depth is to further bias the function  $d\text{-spec}$  towards the most specific subject types.

We compute also the *range*-specificity of  $P$  with respect to  $F$ , by adapting the  $r\text{-spec}$  function to consider the object types instead of the subject types. Similarly to Equation 5.8, we formally define the set  $\text{Ran}[\Delta]$  of object types extracted from a generic extension  $\Delta$  as:

$$\text{Ran}[\Delta] = \{C \mid \exists \delta_{(s,o)} \in \Delta, C \in \text{types}(o) \wedge \nexists C' \in \text{types}(o), C' \preceq C\}. \quad (5.11)$$

We thus adapt Equation 5.9 accordingly:

$$r\text{-spec}(F, P) = \frac{w\text{-card}(\text{Ran}[\Delta_P] \cap \text{Ran}[\Delta_P^F])}{w\text{-card}(\text{Ran}[\Delta_P] \cup \text{Ran}[\Delta_P^F])}. \quad (5.12)$$

### Coverage

We capture the *coverage* of a relation  $P$  over a facet  $F$  by computing the rate of facet values for which there exists at least one matched signature  $\delta_{(s,o)} \in \Delta_P^F$ . Intuitively, we aim at capturing percentage of facet values for which there is evidence of them being object of the relation. More formally:

$$\text{cov}(F, P) = \frac{|\{v \in \mathcal{V} \mid \exists \delta_{(s,o)} \in \Delta_P^F \text{ s.t. } r\text{-match}(\{v\}, \delta_{(s,o)})\}|}{|\mathcal{V}|}. \quad (5.13)$$

The rationale behind the coverage is to boost those relations whose objects are equally distributed (i.e., they cover) over the facet value set  $\mathcal{V}$ .

### Weighted Frequency

Observe that both specificity and coverage do not take into account the cardinality of the extension  $\Delta_P^F$ , which can in principle give a clue on the similarity between  $P$

## 5. SPECIFICITY-BASED INTERPRETATION OF FACETS

---

and  $F$ : the bigger  $\Delta_P^F$ , the more evidence we got from KG that the semantics of  $P$  is preserved when annotating  $F$ . However, recall that  $\Delta_P^F$  may include false positives resulting from the our matching phase. As a result, not all the evidence gathered from  $\Delta_P^F$  may equally contribute in assessing the similarity between  $P$  and  $F$ . In principle, the highest quality evidence that we may get from KG are signatures whose subject type is *exactly* the facet domain  $C$ .

So as to capture this principle, we introduce the *weighted frequency*. Intuitively, we count signatures in  $\Delta_P^F$ , weighting them considering the similarity between the lexicalization of facet domain  $lex(C)$  and the lexicalization of all subject types  $\overline{lex}(s)$ . Our intuition is that the more similar  $lex(C)$  and  $\overline{lex}(s)$ , the higher quality of the evidence provided by the signature. More formally:

$$freq(F, P) = \frac{1}{|\mathcal{V}|} \sum_{\delta_{(s,o)} \in \Delta_P^F} \frac{|\overline{lex}(s) \cap lex(C)|}{|\overline{lex}(s) \cup lex(C)|} \quad (5.14)$$

where the quality of each signature  $\Delta_P^F$  is computed as the Jaccard similarity between the two lexicalizations. The factor  $1/|\mathcal{V}|$  is introduced to scale down the resulting weighted frequency by the number of the facet values (i.e., the cardinality of  $\mathcal{V}$ ).

### Computing the Final Rank

We aggregate all the scores that capture the specificity, coverage and frequency principles into a single, global similarity score for a relation  $P$  and a  $F$ . Observe that all the scores provide values within the  $[0, 1]$  interval, with the noticeable exception of the *freq* score (i.e., weighted frequency). The *freq* score may in principle have a huge impact on the overall score because provides positive evidence captured in an unbounded way. As a result, it can possibly skew the overall score.

Thus, to make all the scores more comparable we smooth them by a logarithmic factor. Moreover, we enforce 1 as the lower bound of each score and then multiply all the smoothed scores together. This aggregation results in a similarity score which is a non-decreasing monotone function with no upper bound and lower bound equal to 1.

More formally:

$$drc(F, P) = \left(1 + \ln(d\text{-spec}(F, P) + 1)\right) \cdot \left(1 + \ln(r\text{-spec}(F, P) + 1)\right) \cdot \left(1 + \ln(cov(F, P) + 1)\right) \cdot \left(1 + \ln(freq(F, P) + 1)\right) \quad (5.15)$$

where the +1 inside the logarithm ensures each score  $\geq 0$  while the +1 outside the logarithm enforces 1 as lower bound. Intuitively, we thus let each score to contribute by a positive factor, as we consider positive evidence only.

## 5.5 Evaluation

In our experiments we annotate facets with relations from two different KGs: DBpedia and YAGO. We compare our ranking approach with the *majority voting* model and the *maximum likelihood* model proposed by Venetis et al. [143] adapted to the facet annotation problem. We rely on two different gold standards as ground truth for the annotation: a manually created one for DBpedia and a state-of-the-art one for YAGO, which has been introduced for evaluation of table annotation approaches [74]. Experimental results indicate that our approach outperforms the state-of-the-art ones in annotating facets.

We implemented our approach along with the baseline algorithms using the Java programming language and the Lucene library<sup>4</sup>. The code repository, along with all the gold standards we compared our approach against and experimental results provided in this section are publicly accessible<sup>5</sup>. Experiments were conducted on a Dual Core 2.54GHz processor machine, 4GB RAM, 128GB SSD HD, under the Ubuntu Linux Desktop 64bit operating system.

### 5.5.1 Knowledge Graphs

We consider two different KGs with different characteristics, highlighted in Table 5.1. DBpedia (version 3.9) contains 753958 types, 53195 relations and more than 96M relational assertions involving them. In our experiments, we include two subtype graphs:

<sup>4</sup><http://lucene.apache.org/>

<sup>5</sup><https://bitbucket.org/rporrini/facet-annotation/>

## 5. SPECIFICITY-BASED INTERPRETATION OF FACETS

Table 5.1: Knowledge Graphs’ statistics.

	Types	Relations	Relational Assertions
DBpedia	753958	53195	~ 96.8M
YAGO	184512	89	~ 5.5M

Table 5.2: Gold Standards’ statistics.

	#	Facets	Relations		
		Domains	Vital	Correct	
dbpedia-numbers	8	7	~4	~19	
dbpedia-entities	31	13	~7	~7	
dbpedia	39	13	~3	~9	
yago-explicit	83	17	1	-	
yago-ambiguous	83	10	1	-	

the first one is extracted from the DBpedia ontology, and the second one is extracted from DBpedia categories. DBpedia categories are extracted from Wikipedia categories and are known to be noisy and low quality [62]. The second KG we consider is YAGO (version: 2008-w40-2)<sup>6</sup>, which includes 184512 types, 89 relations and more than 5M relational assertions involving them.

### 5.5.2 Gold Standards

The data sets used for evaluation consist of two disjoint sets of facets manually annotated with relations from DBpedia and YAGO, respectively. In the case of DBpedia we found that there exist more than one relation that is similar to the facet, while in case of YAGO there is only one similar relation for each facet. Statistics about these two gold standards are summarized by Table 5.2 , while Table 5.3 presents few examples of facets from the gold standards.

#### DBpedia Gold Standard

Following the methodology proposed for the creation of gold standards for table annotation tasks [143, 86, 155], we manually collected 39 facets from the Web, from different

<sup>6</sup>The reason for using an old version of YAGO is that the state-of-the-art annotated gold standard that we use to evaluate our approach is built considering this particular version of YAGO [74].



Table 5.3: Example of facets from the gold standards.

	Domain	Values	Relations
<b>dbpedia-numbers</b>	music albums	1967, 1969, 1971, . . . , 2000	relYear, year, releaseDate . . .
<b>dbpedia-entities</b>	wines	USA, Italy, . . . , Slovenia	wineRegion, origin, . . .
<b>yago-explicit</b>	wordnet-actor-109765278	1776, California, . . . , Wilson	actedIn
<b>yago-ambiguous</b>	countries	Abu Dhabi, Accra, . . . , Zagreb	hasCapital

domains. 32 facets were collected from Wikipedia (non infobox) table columns, while 7 were collected from IMDB, eCommerce and sport statistics websites. For each facet, we selected from DBpedia KG all the relations whose object matches *at least one* facet values, without considering facet domain matching. We presented the gold standard to a total of 10 human annotators, who rated the similarity of each relation using a Likert scale from 0 up to 2 (0 = *incorrect*, 1 = *correct*, 2 = *vital*). We presented only a portion of the gold standard to each annotator in order to reduce the cognitive workload, ensuring that each relation was been rated by at least 3 annotators. We aggregated all the judgments, manually resolving inconsistencies. We report an average correlation of 0.42 (Spearman Correlation Coefficient) between annotators. We then partitioned the gold standard into two disjoint sets of facets: **dbpedia-numbers** (i.e., facets whose values are digits) and **dbpedia-entities**.

### YAGO Gold Standard

The gold standard for the YAGO dataset has been used to evaluate several table annotation algorithms [74] and consists of tables where cells have been annotated with entities, columns with types and column pairs with relations (on a total number of 90). We derived 90 facets from all table columns involved in at least one relation annotation, considering the lexicalization of annotated type for the corresponding subject columns as facet domain, when present. Among these 90 facets, 7 had no corresponding type annotation for the corresponding relation’s subject column and thus were discarded. We ended up with 83 annotated facets. Observe that, there are no numeric facets. We then created two versions of the YAGO gold standard. Within the **yago-explicit** gold standard we used the local name of types (extracted from the type URI) as facet domains (e.g., wikicategory-American-science-fiction-novels). By considering the local name of types, we minimize ambiguity in facets domains. Conversely, in the **yago-ambiguous** gold standard, we used more general (and thus, ambiguous),

## 5. SPECIFICITY-BASED INTERPRETATION OF FACETS

---

lexicalizations for facet domains, such as novels.

### 5.5.3 Algorithms

Most of the state-of-the-art table annotation approaches have been compared to the *majority voting* based model [112, 143, 74] (MAJ). Given  $F$ , the majority based approach selects all the relations whose object lexicalizations match at least one facet values and ranks them by frequency. The *maximum likelihood* based model [143] (MAX) is based on the application of the maximum likelihood hypothesis. Given a facet  $F$ , the most similar relation is the one that maximizes the conditional probability of occurrence of the relation given the facet values. More formally:

$$ml(P, F) = K_p \times \prod_{v \in \mathcal{V}} \frac{\Pr[P | v]}{\Pr[P]}, \text{ s.t. } \sum_p ml(P, F) = 1$$

where  $\Pr[P | v]$  and  $\Pr[P]$  are computed over the considered KG.

Observe that these two approaches leverage the quasi-relational structure of a table. When annotating a pair of columns, they match cells of the first column to relation subjects and the cells of the other column to relation objects. In a nutshell, they apply different matching criteria in selecting an appropriate relation extension  $\Delta$  so as to compute the semantic similarity. These matching criteria are tailored to *balanced* input, that is relations whose domain and ranges are specified by means of two sets of values. However, we recall from Section 5.2 that the input of the facet annotation problem is unbalanced, being the facet domain the lexicalization of a type. In order to provide a fair comparison, we compare our relation ranking function (i.e., DRC) with the baselines applied to the extensions  $\Delta_p^F$  induced by facets, computed by matching both the facet domain and values to relational assertions.

### 5.5.4 Evaluation Metrics

In our experiments we compare the effectiveness of DRC, MAJ, and MAX using three different metrics: Normalized Discounted Cumulative Gain (nDCG), Mean Average Precision (MAP), and Mean Reciprocal Rank (MRR). nDCG metric compares the ranking of relations established by one algorithm with the ideal ranking derived by ratings specified by human annotators. Intuitively, a higher nDCG value corresponds to a better agreement between the results proposed by the facet annotation approach and an

ideal ordering obtained from human annotators. The nDCG provides a fine-grained measure of the effectiveness of DRC and the baselines, at different ranks. We use it for comparison on both the DBpedia and YAGO datasets.

The MAP metric is computed by averaging the precision calculated at the rank of each *correct* or *vital* relation in  $P$  (as judged by human annotators) for each facet and finally averaged over all the facets. MAP takes into account the fact that there exists more than one similar relation for each facet, and provides a single valued, coarse-grained measure of effectiveness. We use it to for comparison against the DBpedia datasets. The MRR metric is an adaptation of the MAP measure to consider facets for which there exists only one relation. MRR is defined as the multiplicative inverse of the rank of the *vital* relation (if present, 0 otherwise), averaged over all the facets. Intuitively, MRR captures to which extent an high rank is given to the similar relation by the algorithm, being maximum when the similar relation is always ranked first. We use it for comparison against the YAGO gold standard.

### 5.5.5 Experimental Results

#### DBpedia

We compare DRC with the baselines with respect to MAP computed over the top 20 similar relations. We assumed all relations judged as either *correct* or *vital* by human annotators to be equally similar, because we want to ensure that DRC is able to discriminate between similar and not similar relations in a boolean fashion. Experimental results, depicted in Figure 5.4a. DRC consistently achieves better effectiveness in terms of MAP over all the DBpedia datasets. From a more granular point of view, Figure 5.4b gives an overview of the nDCG obtained at different ranks. DRC is more effective than MAJ and MAX to capture the similarity at all the ranks from 1 to 20, achieving a maximum nDCG of 0.81 (at rank 1), compared to 0.77 of MAJ and 0.76 of MAX. The interesting observation is that the more “naive” MAJ baseline performs better than the MAX baseline, at all ranks. These results are compliant the ones provided by the authors of MAX [143]. In they work, they ultimately composed their approach with the majority based one (MAJ) to achieve better results. However, both MAJ and MAX are less effective that DRC in capturing the semantic similarity between facets and DBpedia relations.

Our interpretation for this behavior is that the baselines are less robust to ambigu-

## 5. SPECIFICITY-BASED INTERPRETATION OF FACETS

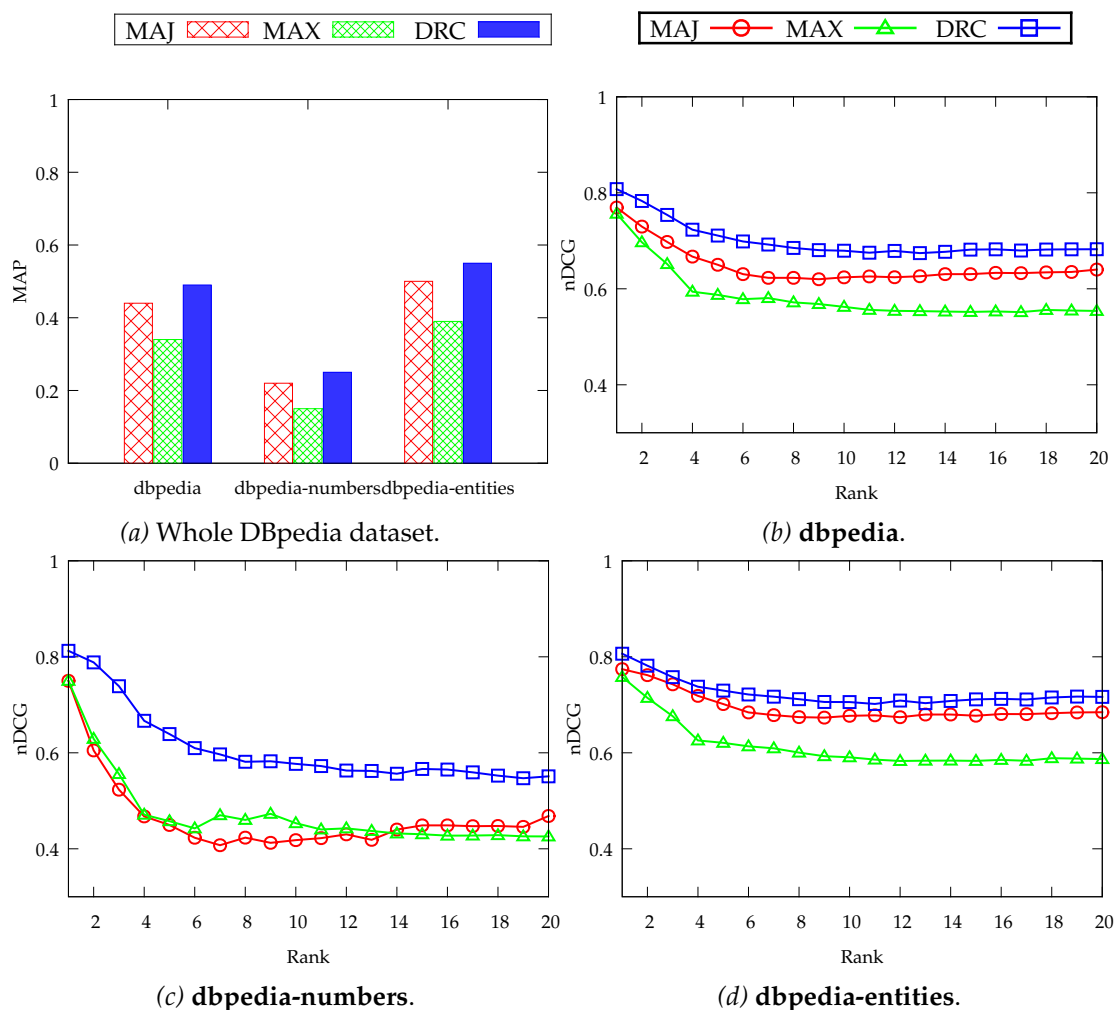


Figure 5.4: Evaluation of DRC over the DBpedia dataset.

ity introduced by the unbalanced specification of facets, which potentially results into an high number of false positive matches between the facet and relational assertions. The baselines, which are not explicitly designed to cope with this distinctive characteristic of facets, are biased towards relations whose extension induced by the input facet has an higher cardinality, even if it contains false positive matches. In contrast, DRC is less sensitive to false positive matches as frequency is weighted by the similarity between the facet domain and the subject type. For example, the top ranked relation computed by DRC for the facet  $F = \langle \text{Airports}, \{\text{Milan}, \text{Rome}, \dots, \text{Turin}\} \rangle$  is *cityServed*, which is ranked only at the fourth place by both MAJ and MAX.

The observation about the robustness of DRC with respect ambiguity of facets is

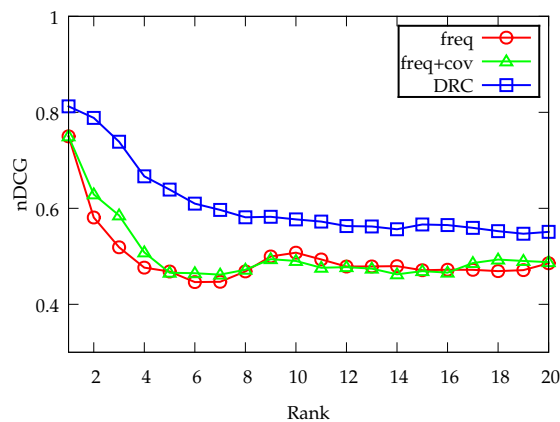


Figure 5.5: The effect of the *specificity* score on the **dbpedia-numbers** dataset.

confirmed by the nDCG achieved by our approach on the **dbpedia-numbers** dataset, depicted in Figure 5.4c. The semantics of digits is ambiguous and is context dependant, as exemplified by the facet  $F = \langle \text{Basketball Players}, \{1948, 1949, \dots, 2012\} \rangle$ . This facet has been extracted from a Web page that lists NBA basketball players by their draft year. The semantics of this facet is ambiguous because, besides the draft year (i.e., the year where a player starts playing for an NBA team as a professional), it may represent for example the birth or retirement year. However, the draft year is one of the most noticeable relations of basketball players, while the birth or retirement year characterize people or athletes in general. Thus, the draft year is *specific* with respect to the domain of basketball players. In this setting, the specificity scores implemented by DRC and in particular the domain specificity is able to boost relations that model the draft year (e.g., the relation `draftYear`) over more generic ones (e.g., `birthDate`). Instead, MAX and MAJ, which do not capture the specificity of relations, assign an higher rank to relations such as `birthDate` or `deathDate`.

The effect of the *specificity* score on DRC on the **dbpedia-numbers** dataset is shown in Figure 5.5. We compare three different versions of DRC in terms of nDCG: the first one uses only the frequency score (i.e., *freq*), the second one uses both the frequency and the coverage score (i.e., *freq+cov*), while the third one is the complete DRC ranking function. Experimental results clearly highlight the importance of capturing the specificity of relations for facet annotation. The specificity score is crucial to provide high quality annotations for numeric facets. We also study the effect of the specificity score over the **dbpedia-entities** dataset and found that it provides an observable, though slight, improvement of effectiveness. This was therefore expected, as entities are less ambiguous compared to numbers. We report an higher nDCG at all ranks for

## 5. SPECIFICITY-BASED INTERPRETATION OF FACETS

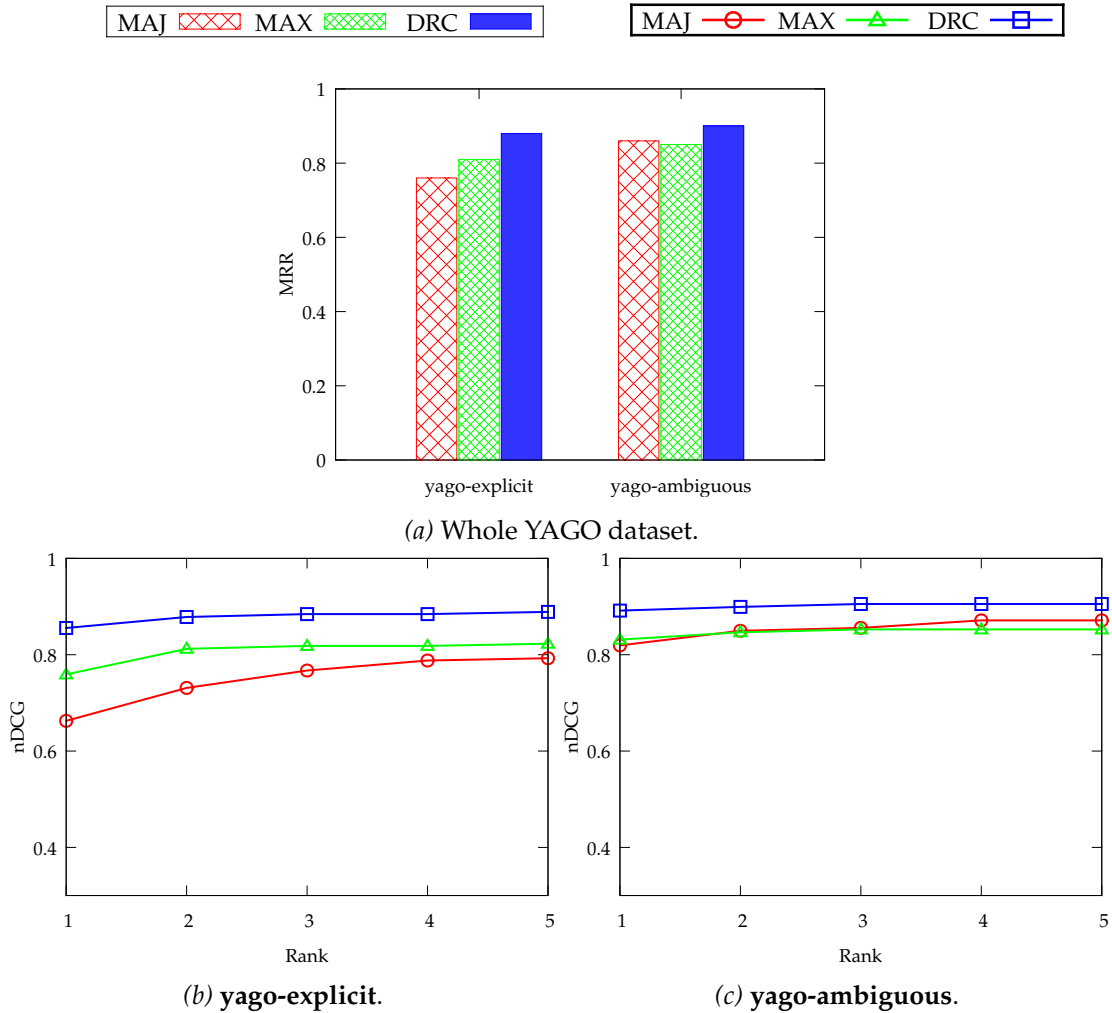


Figure 5.6: Evaluation of DRC over the YAGO dataset.

DRC compared to freq and freq+cov scores, with an average improvement of around 4.5% and 5%, respectively.

### YAGO

We compare DRC with the baselines with respect to MRR computed over the top 5 similar relations for the YAGO dataset. Insights from the previous experiments are further confirmed: the most similar relation for facets is generally ranked higher by DRC compared to baselines, as shown in Figure 5.6a.

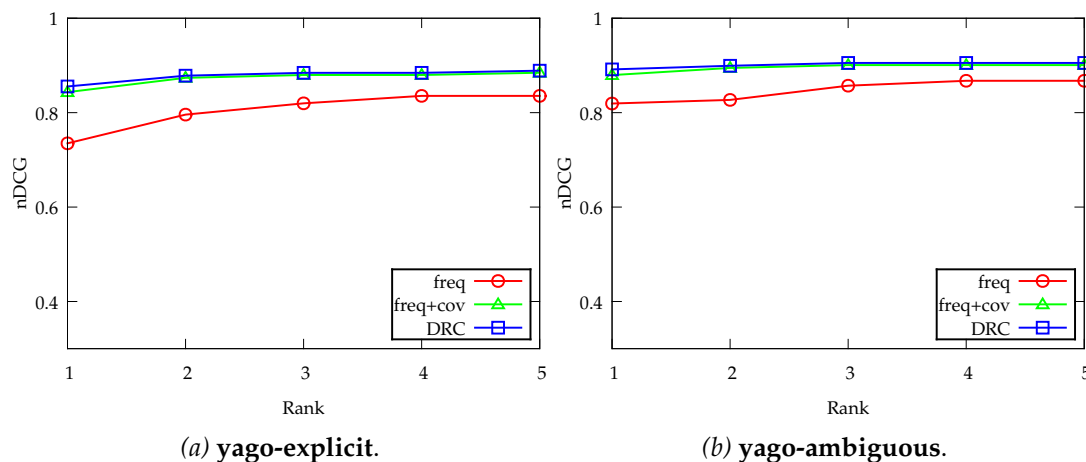


Figure 5.7: The effect of the *specificity* score over the YAGO dataset.

From a more granular point of view, we compare DRC with baselines in terms nDGC (Figures 5.6b and 5.6c). Our approach is able to provide a better characterization of the similarity between a facet and relations also in presence of a single, well defined type hierarchy and a much smaller number of possibly similar relations (89 vs 53195 of the DBpedia dataset), making it suitable also for the annotation of facets with precision oriented KGs. DRC achieves better results at each rank. Observe that the most noticeable improvement over the baselines is obtained at rank 1. This is crucial in order to support a fully automatic (and not validated by humans) facet annotation process. An example of such capability is provided by the facet  $F = \langle \text{Actors}, \{\text{Blondie Hits the Jackpot}, \text{Charlie Chaplin Cavalcade}, \dots\} \rangle$ . For this facet, the top ranked relation computed by DRC for is `actedIn`, which is ranked at the second position (after `created`) by MAJ and ranked at the third position by MAX (after `livesIn` and `created`) by MAX.

As for the DBpedia datasets, in our last experiment we study the effect of the *specificity* score on the quality of the annotation for facets of the YAGO datasets. Again, we compare the complete version of DRC with `freq` and `freq+cov` scores. Experimental results depicted in Figure 5.7 show that the *specificity* score have a little impact on the effectiveness of DRC on the YAGO dataset. However, we recall that the YAGO dataset includes *no* numeric facets, and thus we expected the *specificity* score to have little impact on quality in this particular dataset. Moreover, the subtype graph provided by the YAGO KG is deeper and more specialized compared to DBpedia subtype graphs. As a result, the subject and object type sets extracted from the extension of relations are too sparse compared to ones extracted from extensions induced by facets, thus

## 5. SPECIFICITY-BASED INTERPRETATION OF FACETS

---

making the specificity score less discriminative. A more deep investigation on how to better capture the specificity of relations for KGs with similar characteristics to YAGO represents an interesting extension that we leave for future work.

### 5.6 Comparison with Prior Art

---

The facet annotation problem described in this chapter is similar to the problem of Web table annotation [74, 143, 120, 145, 112, 156, 86, 154, 153, 155]. The general goal of table annotation approaches is to interpret a table by annotating cells with KG entities, columns with KG types, and pairs of columns with KG relations. The approaches that more relevant to ours tackle the problem of relation annotation [74, 143, 145, 86, 155]. Other state-of-the-art table annotation approaches that are less relevant to the contribution described in this chapter are described in detail in Section 3.2.

Table annotation approaches annotate pairs of columns with relations from a given KG [74, 143, 145, 86, 155]. As stated in Section 5.2, the main difference between the specification of a facet and a relation between to table columns is that the first is unbalanced (i.e., the facet domain is specified differently from the range) while the latter is balanced. That is, there is a relation holding between values in different columns whenever they are part of the same row. In contrast, in the facet annotation problem the facet range is a single set of values and there is no other collection of corresponding values on which we can rely. Approaches to relation annotation in web tables heavily rely on the analysis of the reciprocal distributions of values in different columns, but as part of the same row [143, 155]. Due to the unbalanced specification of facets, this kind of information is not available, when dealing with the facet annotation problem. However, approaches to relation annotation in web tables leverage the extensional semantics of relations specified by a KG.

Venetis et al. [143] build KG mined from Web pages using the TextRunner relation extraction framework [9]. They apply a maximum likelihood inference model to estimate the probability of a KG relation that holds between values from two columns. The same inference model is leveraged in order to annotate columns with KG types. The estimation of the maximum likelihood model for relation annotation is based on computation of the frequencies of relational assertions within the KG of all pairs of values of the same row, and does not consider any type-based representation of KG instances. As a result, the approach from Venetis et al. does not allow to quantify the specificity of a relation with respect to instances of a KG type. We claim that this



---

## 5.7 Application to Table Annotation Problems

---

is a crucial feature for a facet annotation approach that deals with the unbalanced specification of a facet. This claim is substantiated by results of our experiments from Section A.6, where we compared our proposed approach with the maximum likelihood based approach adapted to consider the unbalanced specification of a facet, and shown that is less effective in annotating a facet.

Wang et al. [145] annotate web tables using the Probase KG [149]. Probase is a probabilistic KG and it provides scores that model the *plausibility* and the *ambiguity* of entities being instance of a certain type and characterized by certain relations. Those scores are computed during the creation of the KG. The approach to table annotation by Wang et al. [145] is based on the hypothesis that tables often describe real world entities and thus they consist of column that denotes the main entity (i.e., the *subject column*) and multiple relation columns. They interpret a table by identifying the subject column, annotate it with the most suitable Probase type and then annotate the remaining columns with the most suitable relations. Their approach is KG dependent, since they heavily rely on the plausibility and ambiguity scores provided by Probase in order to compute the overall score of a candidate annotation. In contrast, our approach is KG agnostic, since we rely solely on types, which are present in any KG.

TableMiner [155] annotates tables with entities, types and relations from the Freebase KG. One of the main contributions of TableMiner is the usage of contextual information extracted from the Web page that contains the table (e.g., table caption, surrounding text, RDFa/Microdata annotations). Following the same intuition of Wang et al. [145], TableMiner starts with identifying the subject column and provides a set of preliminary entity and type annotations. Those annotations are then jointly refined using an iterative learning procedure. Relation annotations are then computed based on the result of the refinement phase. This approach can hardly be adapted to our problem, mainly because in our setting we lack contextual information for facets being them typically the result of automatic facet extraction approaches (as the one described in Chapter 4) and thus they do not have any caption nor text directly surrounding them and lack any kind of RDFa/Microformat markup.

## 5.7 Application to Table Annotation Problems

---

The approach presented in this chapter tackles the problem of interpreting a facet by annotating it with relations from a KG. As we highlighted in Section 5.2, solving the facet annotation problem basically accounts to solving a particular relation annotation

## 5. SPECIFICITY-BASED INTERPRETATION OF FACETS

ID	Case Number	Date	Block	IUCR	Primary Type	
1	10491950	HZ232851	04/19/2016 11:50:00 PM	026XX W PERSHING RD	1811	NARCOTICS
2	10491982	HZ232852	04/19/2016 11:45:00 PM	079XX S HALSTED ST	1330	CRIMINAL TRESPASS
3	10493028	HZ233790	04/19/2016 11:45:00 PM	0000X S STATE ST	0820	THEFT
4	10491995	HZ232864	04/19/2016 11:41:00 PM	0000X W MADISON ST	0320	ROBBERY
5	10493357	HZ232855	04/19/2016 11:40:00 PM	027XX N SPAULDING AVE	0610	BURGLARY
6	10493694	HZ234375	04/19/2016 11:40:00 PM	012XX S PRAIRIE AVE	0460	BATTERY
7	10492023	HZ232880	04/19/2016 11:37:00 PM	016XX W 79TH ST	0554	ASSAULT
8	10491997	HZ232873	04/19/2016 11:36:00 PM	048XX W POLK ST	0620	BURGLARY
9	10492599	HZ233287	04/19/2016 11:30:00 PM	015XX E 62ND ST	0820	THEFT
10	10493041	HZ233761	04/19/2016 11:30:00 PM	051XX S WENTWORTH AVE	0486	BATTERY
11	10491993	HZ232867	04/19/2016 11:30:00 PM	048XX W MONROE ST	143C	WEAPONS VIOLATION

Figure 5.8: A table with an uninformative subject column.

(i.e., interpretation) problem. Although our approach is tailored to facets, we believe that it can be applied to solve other types of problems which account to the establishment of a r-to-r mapping (i.e., relation-to-relation, see Section 2.1.3).

As highlighted in Section 5.6, table annotation approaches establish r-to-r mappings when annotating a table column with the relation that hold between values in such column and values in a previously identified subject column. Table annotation approaches analyze the reciprocal distributions of values in the subject column and the column to be annotated, when part of the same row, under the assumption that values in the subject columns can be linked to entities in the KG, for example by matching their lexicalizations. However, there are tables for which this assumption does not hold, as for example the table depicted in Figure 5.8. This table includes data about crimes reported in the city of Chicago, and is publicly accessible through the Chicago Open Data portal<sup>7</sup>. The subject column of this table is the **ID** column that contain a progressive numerical identifier and, remarkably, says nothing about the entities described by the table.

This setting is common, especially when dealing with highly domain specific tabular data released by public institution in the Open Government context. In this sit-

<sup>7</sup><https://data.cityofchicago.org/Public-Safety/Crimes-2001-to-present/ijzp-q8t2> - Accessed on April 2016

## 5.7 Application to Table Annotation Problems

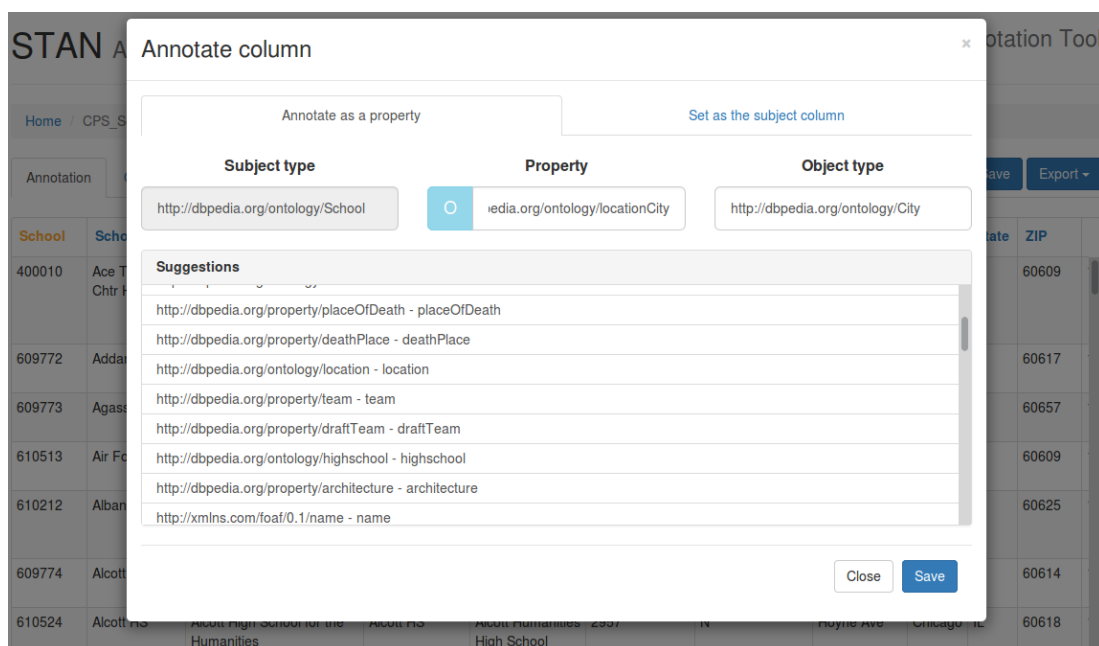


Figure 5.9: A screenshot of the column annotation feature of STAN.

uation, state-of-the-art table annotation approaches are likely to fail in annotating the non-subject columns with relations from a KG, because they are unlikely to be able to match values in this column with the lexicalization of KG entities. We believe that our facet annotation approach can, at least in principle, help to solve this problem. In fact, our approach does not assume the presence of two co-ordinate sets of values, but only the lexicalization of the facet domain and values of the facet range. In the case of the table depicted in Figure 5.8, a similar representation can be provided for non-subject columns. For example, the column **Primary Type** can be represented as a facet:

< Crime, {Narcotics, Criminal Trespass, Theft, Robbery . . .} >

where the facet domain can be specified by hand by domain experts, or even automatically extracted by leveraging data profiling techniques that identify the general topic of a table (e.g., [19]).

As a proof of concept, we adapted and included our facet annotation approach in STAN<sup>8</sup>: a semantic table annotation tool that supports the annotation of tabular data with types and relations from a defined KG. Figure 5.9 depicts a screenshot of the column annotation feature provided by STAN. At the time of writing, STAN supports

<sup>8</sup><http://stan.disco.unimib.it> - <https://github.com/brando91/STAN>

## 5. SPECIFICITY-BASED INTERPRETATION OF FACETS

---

the annotation of a table with the DBpedia KG. Our facet annotation approach is leveraged to suggest relations to annotate table columns. Although the application of our approach to table annotation problems is out of the scope of this dissertation, we believe that the state-of-the-art can benefit from the insights we got by tackling the facet annotation problem. This represents an interesting research direction that we plan to investigate in the near future.

### 5.8 Summary

---

In this chapter we presented an automatic facet interpretation approach that allows to annotate a facet with KG relations. We proposed an approach that is capable to handle the intrinsic ambiguity of facets, by annotating them with relations that are more specific with respect to their facet domain. Experiments conducted by annotating facets with two large KGs (DBpedia and YAGO) confirm the effectiveness of the proposed approach.

# 6

## Knowledge Graph Profiling

### 6.1 Overview

---

In this chapter we study how to extract, represent and convey information needed by domain experts in order to supervise and validate the extraction and interpretation of facets, discussed in the last chapters. In particular, we argue that such information can be effectively provided by means of a *summary* of the KG, which allows domain experts to answer to questions like: what types of instances are described in the KG? What relations are used to characterize the instances? What types of instances are linked by a certain relation and how frequently? How many instances have a certain type and how frequent is the use of a given relation? With such knowledge at hand, domain experts can properly supervise the extraction and annotation phases and validate their result, deciding for example to annotate a facet with a relation that they consider more domain specific with respect to automatically suggested ones.

Here we introduce ABSTAT: a framework that is capable to provide a complete and compact abstract summary of the content of a KG, focusing on KG that are modeled using the RDF data model. With completeness 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 to adopt a minimalization mechanism based on *minimal type patterns*. A minimal type pattern is a triple  $(C, P, D)$  that represents the occurrences of assertions  $P(a, b)$  in RDF data, such that  $C$  is a minimal type of the subject  $a$  and  $D$  is a minimal type of the object  $b$ . By considering patterns that are based on minimal types we are able to exclude several redundant patterns from the summary. The ABSTAT<sup>1</sup> frame-

---

<sup>1</sup><http://abstat.disco.unimib.it>

## 6. KNOWLEDGE GRAPH PROFILING

---

work supports users to query (via SPARQL), to search and to navigate the summaries through web interfaces. The remainder of this chapter is organized as follows. The summarization model is presented in section 6.2. The implementation of the model in ABSTAT is given in Section 6.3. Experiments conducted in order to validate our proposed approach to KG summarization are presented in Section 6.4 while we discuss the related work in Section 6.5. A final summary of the chapter is presented in Section 6.6.

### 6.2 Summarization Model

---

Recall from Section 2.1.1 that a KG is constituted by a *terminology* (or signature)  $\mathcal{N}$ , a set of *terminological axioms*  $\mathcal{T}$  and a set of *assertions*  $\mathcal{A}$

$$\mathcal{K} = \langle \mathcal{N}, \mathcal{T}, \mathcal{A} \rangle .$$

The terminology  $\mathcal{N}$  of the KG contains the set  $\mathcal{N}^C$  of *types*, the set  $\mathcal{N}^P$  of named relations and the set of instances  $\mathcal{N}^I$  composed by entities and literals. Consistently with the used in the rest of the dissertation, we use symbols like  $C, C', \dots$ , and  $D, D', \dots$ , to denote types, symbols  $P, Q$  to denote relations, and symbols  $a, b$  to denote instances.

Assertions in  $\mathcal{A}$  are of two kinds: *type assertions* of form  $C(a)$ , and *relational assertions* of form  $P(a, b)$ , where  $a$  is an entity and  $b$  is either an entity or a literal. We denote the sets of type 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 signature. *Type assertions* occur in a KG as RDF triples  $\langle x, \text{rdf:type}, C \rangle$  when  $x$  and  $C$  are URIs, or can be derived from triples  $\langle x, P, y \hat{C} \rangle$  where  $y$  is a literal (in this case  $y$  is a typed literal), with  $C$  being its datatype. Consistently with definitions from Chapter 2, we say that  $x$  is an instance of a type  $C$ , denoted by  $C(x)$ , either  $x$  is an entity 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. Conversely, an instance can be the subject of several type assertions. A *relation assertion*  $P(x, y)$  is any triple  $\langle x, P, y \rangle$  such that  $P \neq Q$ , where  $Q$  is either `rdf:type` or one of the relations used to model a terminology (e.g. `rdfs:subClassOf`).

### Subtype Graph

A *subtype graph* is a graph

$$G = (\mathcal{N}^C, \preceq)$$

where  $\mathcal{N}^C$  is the set of types (either concept or datatype) and  $\preceq$  is a relation over  $\mathcal{N}^C$ . We always include two type names in  $\mathcal{N}^C$ , namely `owl:Thing` and `rdfs:Literal`, such that every concept 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.

### Abstract Knowledge Pattern

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 [129]. For sake of clarity, we will use the term *pattern* to refer to an AKP in the rest of the Chapter. A pattern is a triple

$$(C, P, D)$$

such that  $C$  and  $D$  are types and  $P$  is a relation. Intuitively, an AKP states that there are instances of type  $C$  that are linked to instances of a type  $D$  by a relation  $P$ . In ABSTAT we represent a set of AKP occurring in the KG, which profiles the usage of the terminology. However, instead of representing every AKP occurring in the KG, 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 the subtype graph.

### 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

$$\{C(x), P(y), P(x, y)\} \subseteq \mathcal{A}.$$

Patterns are also denoted by the symbol  $\pi$ . For KGs that include 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 the subtype graph that represents types and their subtypes.

## 6. KNOWLEDGE GRAPH PROFILING

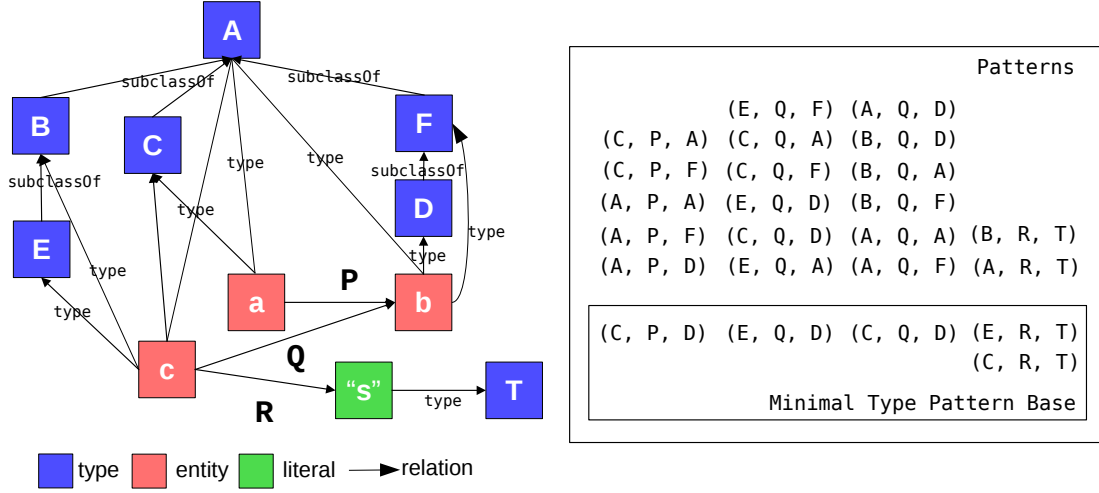


Figure 6.1: A small KG and corresponding patterns.

### 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 subtype 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  $C' \preceq C$  or a type  $D'$  such that  $D'(b) \in \mathcal{A}$  and  $D' \prec^G D$ .

### 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  $\hat{\Pi}^{\mathcal{A},G}$  such that  $\pi \in \hat{\Pi}^{\mathcal{A},G}$  iff  $\pi$  is a minimal type pattern for some relational 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. However, 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 6.1, 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  $\mathcal{A}$ :

$$(C, P, D), (C, P, F), (C, P, A), (A, P, D), (A, P, F), (A, P, A).$$



The minimal type pattern base for the KG includes the patterns:

$$(E, Q, D), (E, R, T), (C, Q, D), (C, R, T), (C, P, D)$$

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.

### Data Summary

A *summary* of a KG  $\mathcal{K} = \langle \mathcal{N}, \mathcal{T}, \mathcal{A} \rangle$  is a triple

$$\Sigma^{\mathcal{A}, \mathcal{T}} = \langle G, \hat{\Pi}^{\mathcal{A}, G}, Stat \rangle$$

such that:  $G$  is *Subtype Graph*,  $\hat{\Pi}^{\mathcal{A}, G}$  is a *Minimal Type Pattern Base* for  $\mathcal{A}$  under  $G$ , and  $Stat$  is a set of *statistics* about the elements of  $G$  and  $\Pi$ . Statistics describe the occurrences of types, relations and patterns. They show how many instances have  $C$  as minimal type, how many relational assertions use a relation  $P$  and how many instances that have  $C$  as minimal type are linked to instances that have  $D$  as minimal type by a relation  $P$ .

## 6.3 Summary Extraction

---

Our summarization process, depicted in Figure 6.2, 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 entity is computed. Finally,  $\mathcal{A}^P$  is processed in order to compute the minimal type patterns that will form the minimal pattern base  $\hat{\Pi}^{\mathcal{A}, G}$ . During each phase we keep track of the occurrence of types, relations and patterns, which will be included as statistics in the summary.

### Summary Extraction

The subtype graph  $G$  is extracted by traversing all the subrelation and 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 entities (i.e., *external types*).

## 6. KNOWLEDGE GRAPH PROFILING

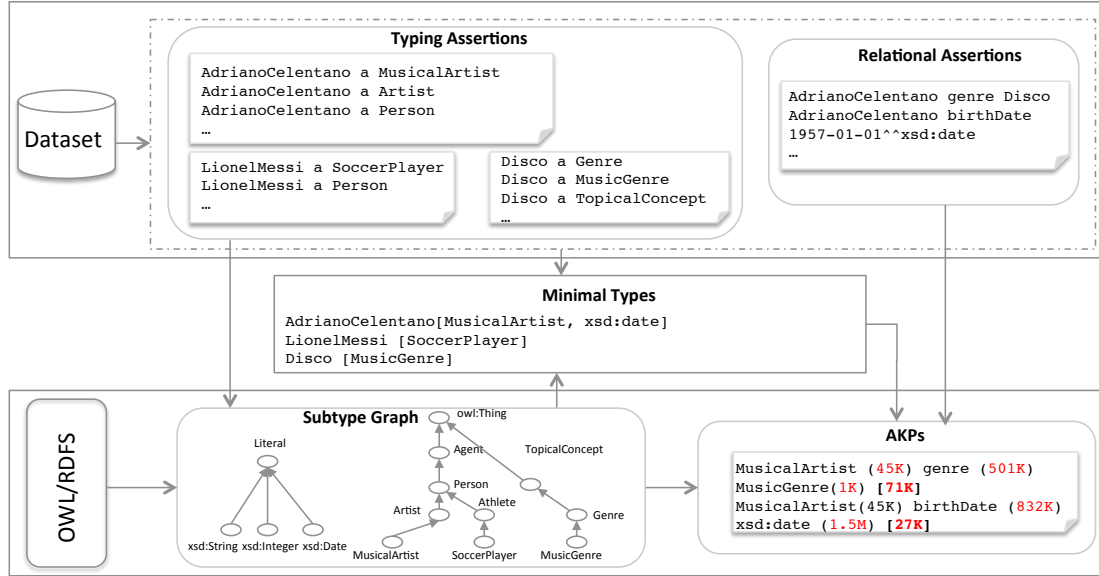


Figure 6.2: The summarization workflow.

Given an entity  $x$ , we compute the set  $M_x$  of minimal types with respect to  $G$ . We first select all the typing assertions  $C(x) \in \mathcal{A}^C$  and form the set  $\mathcal{A}_x^C$  of typing assertions about  $x$ . We then iteratively process  $\mathcal{A}_x^C$ . At each iteration we select a type  $C$  and remove from  $M_x$  all the supertypes of  $C$  according to  $G$ . Then, if  $M_x$  does not contain any  $C'$  such that  $C' \preceq C$ , we add  $C$  to  $M_x$ . Notice that one preliminary step of the algorithm is to include  $C$  in  $G$  if it was not included during the subtype graph extraction phase. If a type  $C$  is not defined in the input terminology, is automatically considered as a minimal type for the entity  $x$ . This approach allows us to handle the types of entities that are not included in the original terminology.

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 type pattern base. If  $y$  is a literal value we consider its explicit type if present, `rdfs:Literal` otherwise.

### Summary Storage and Presentation

Every summary is stored, indexed and made accessible through two user interfaces, i.e., `ABSTATBrowse`<sup>2</sup> and `ABSTATSearch`<sup>3</sup>, and a `SPARQL` endpoint<sup>4</sup>. In particular,

<sup>2</sup><http://abstat.disco.unimib.it>

<sup>3</sup><http://abstat.disco.unimib.it/search>

<sup>4</sup><http://abstat.disco.unimib.it/sparql>

Table 6.1: KGs and summaries statistics.

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

ABSTATSearch<sup>5</sup> implements a full-text search functionality over a set of summaries. Types, relations and patterns are represented by means of their local names (e.g., “Person”, “birthPlace” or “Person birthPlace Country”), conveniently tokenized, stemmed and indexed, and retrieved using Lucene Score as ranking model.

## 6.4 Evaluation

We evaluate our summaries from different, orthogonal perspectives. We measure the *compactness* of the summaries obtained by ABSTAT with the one obtained by Loupe [84], an approach similar to ours that does not use minimalization. We show that our summaries provide useful insights on the semantics of relations, based on their usage within a KG. In this evaluation we use the summaries extracted from three KG: DBpedia Core 2014 (**db2014-core**)<sup>6</sup>, DBpedia 3.9 (**db3.9-infobox**)<sup>7</sup> and Linked Brainz (**lb**). **db2014-core** and **db3.9-infobox** KGs are based on the DBpedia ontology while the **lb** KG is based on the Music Ontology. DBpedia and LinkedBrainz have “complementary relations and requirements”; and “contain real, large scale data, being challenging enough to assess the abilities of QA systems” [76]. Finally, a user study has been carried out to evaluate if our summaries are informative enough to help users to understand a large KG.

### 6.4.1 Compactness

Table 6.1 provides a quantitative overview of KGs and their summaries. To evaluate 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

<sup>5</sup><http://abstat.disco.unimib.it/search>

<sup>6</sup>The DBpedia 2014 version with mapping based relations only

<sup>7</sup>The DBpedia Core 3.9 version plus automatically extracted relations

## 6. KNOWLEDGE GRAPH PROFILING

---

summary has been extracted.

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

### 6.4.2 Informativeness

Although data exploration is a complex task [66], we want to evaluate the informativeness of ABSTAT on specific ones; to provide insights about the semantics of the relations and to help users formulating SPARQL queries.

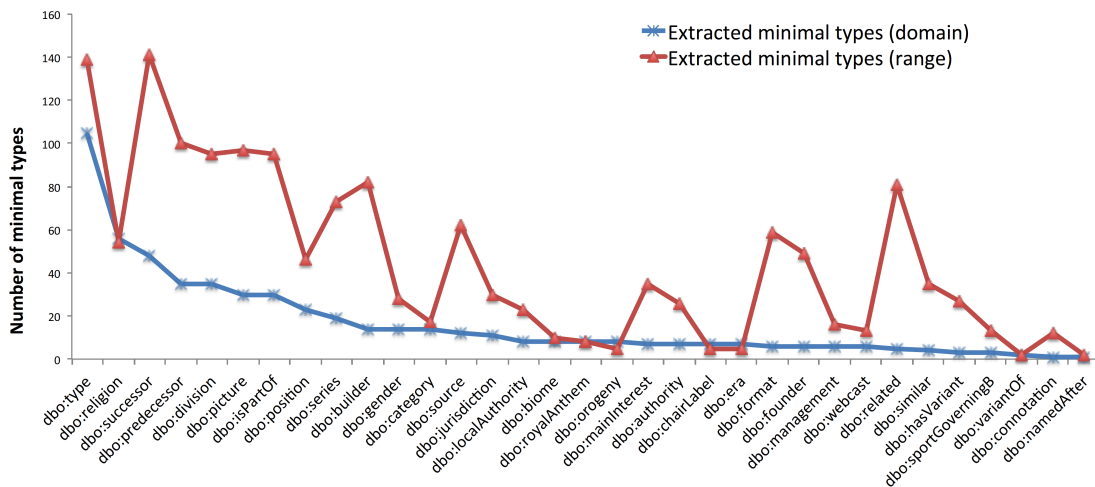
#### Insights About the Semantics of the Relations

Our summaries convey valuable information on the semantics of relations for which the terminology does not provide any domain and/or range restrictions. Table 6.2 provides an overview of the total number of unspecified relations from the KGs. For example, around 18% of relations from **db2014-core** KG have no domain restrictions while 13% have no range restrictions. Observe that this KG 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** KG which includes also triples generated by Information Extraction approaches, most of the relations (i.e., the ones from the `dbpedia.org/property` namespace) are not specified within the terminology.

In general, underspecification may be the result of precise modeling choices, e.g., the relation `dc:date` from the **lb** KG. This relation 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` relation from the **db2014-core** KG, whose domain is not specified in the corresponding terminology. However, this relation is

Table 6.2: Total number of relations with unspecified domain and range in each KG.

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

Figure 6.3: Distribution of the number of minimal types from the domain and range extracted for not specified relations of the **db2014-core** KG.

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` relation by looking only at the terminology axioms is not possible.

We can push our analysis further to a more fine-grained level. Figure 6.3 provides an overview of the number of different minimal types that constitute the domain and range of unspecified relations extracted from the summary of the **db2014-core** KG. The left part of the plot shows those relations 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` relation. Surprisingly, the `dbo:religion` relation is among them: its semantics is not as clear as one might think, as its range covers 54 disparate minimal types, such as `dbo:Organization`, `dbo:Sport` or `dbo:EthnicGroup`. Conversely, the relation `dbo:variantOf`, whose semantics is intuitively harder to guess, is used within the KG with a very specific meaning, as its domain and range covers only 2 minimal types: `dbo:Colour` and `dbo:Automobile`.

## 6. KNOWLEDGE GRAPH PROFILING

---

### Preliminary User Study

Formulating SPARQL queries requires prior knowledge about the KG, a knowledge that we want to support with ABSTAT. We designed a user study where we asked participants to formulate SPARQL queries selected from the *Questions and Answering in Linked Open Data* benchmark<sup>8</sup> [142] to the **db3.9-infobox** KG. The selected queries were taken from logs of the PowerAqua QA system and are believed to be representative of realistic information needs [76], although we cannot guarantee that they cover every possible information need. We provided the participants the query in natural language and a “template” of the corresponding SPARQL query, with spaces intentionally left blank for relations and/or concepts. 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:

```
SELECT DISTINCT ?uri
WHERE { ?uri ... <Vienna> . ?uri ... <Berlin> . }
```

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 [102]. Overall 20 participants with no prior knowledge about the ABSTAT framework were selected and split into 2 groups: **abstat** and **control**. We profiled all the participants in terms of knowledge about SPARQL, data modeling, DBpedia dataset and ontology, so as to create two homogeneous groups. We trained for about 20 minutes on how to use ABSTAT only the participants from the first group. Both groups execute SPARQL queries against the **db3.9-infobox** KG through the same interface and were asked to submit the results they considered correct for each query. We measured the time spent to complete each query and the correctness of the answers. The correctness of the answers is calculated as the ratio between the number of correct answers to the given query against the total number of answers. Table 6.3 provides the results of the performance of the users on the query completion task<sup>9</sup>. The time needed to perform the 5 tasks from all participants in average is 38.6m, while the minimum and the maximum time is

<sup>8</sup><http://qald.sebastianwalter.org/>

<sup>9</sup>The raw data from the user study can be found at <http://abstat.disco.unimib.it/downloads/user-study>

Table 6.3: Results of the user study.

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

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. Using 5 tasks to perform the user study is coherent with other related work which suggest that the user study would have 20-60 participants, who are given 10-30 minutes of training, followed by all participants doing the same 2-20 tasks [102].

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 entities mentioned in the query and very few of them submitted exploratory 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 KG, which is why this exploratory approach was successful for relatively simple queries. However, this exploratory approach does not work with other non-indexed KGs (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 tasks at increasing difficulty, still performing the tasks faster.

## 6. KNOWLEDGE GRAPH PROFILING

---

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 due to the fact that the entity Surfing (which is used as object of the relation `dbo:occupation`) is classified with no type other than `owl:Thing`. As a consequence, participants from the **abstat** group went through a more time consuming trial and error process in order to guess the right type and relation. 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 [101] in our model to cope with untyped instances and suggest possible improvements of ABSTAT for example including values for concepts that are defined by closed and relatively small instance sets.

### 6.5 Comparison with Prior Art

---

We compare our work to approaches explicitly proposed to summarize Linked Data and ontologies, and to extract statistics about the KG. A first body of work has focused on summarization models aimed at identifying subsets of KGs that are considered to be more relevant. In [152], terminological axioms are ranked based on their salience to present to the user a view about the terminology of a KG. RDF Digest [141] identifies the most salient subset of a KG including the distribution of instances in order to efficiently create summaries. Differently from these approaches, ours aims at providing a complete summary with respect to the KG.

A second body of work has focused on approaches to describe KGs by reporting statistics about the usage of the terminology in the data. The most similar approach to ABSTAT is Loupe [84], a framework to summarize and inspect Linked Data KGs. Loupe extracts types and relations, along with a rich set of statistics. Similarly to ABSTAT, Loupe offers a triple inspection functionality, which provides information about *triple patterns* that appear in the KG and their frequency. Triple patterns are equivalent to our patterns. However, Loupe does not apply any minimalization technique: as shown in Section 6.4.1, summaries computed by our model are significantly more compact.

In [22], authors consider vocabulary usage in the summarization process of an



RDF KG and use information similar to patterns. A similar approach is also used in MashQL [51], a system proposed to query a KG without prior knowledge about the terminology used to characterize instances. Our model excludes several redundant patterns from the summary through minimalization, thus producing more compact summaries. Knowledge pattern extraction from RDF KGs is also discussed in [111], but in the context of domain specific experiments and not with the purpose of defining a general KG summarization framework. Our summarization model can be applied to any KG that analyzes the usage of the terminology within the KG and focuses on the representation of the summary.

Other approaches proposed to describe KGs do not extract connections between types but provide several statistics. SchemeEx extracts interesting theoretic measures for large KGs, by considering the co-occurrence of types and relations [61] in the representation of instances. A data analysis approach on RDF KGs based on an warehouse-style analytic is proposed in [29]. 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 KGs, and long runtime. However, this approach differently from ours requires the design of a data warehouse especially for a RDF KG. Linked Open Vocabularies<sup>10</sup>, RDFStats [67] and LODStats [4] provide several statistics about the usage types and relations but they do not represent the connections between types.

## 6.6 Summary

---

Getting an understanding of the shape and nature of the data from large KGs so is a complex and a challenging task, yet vital to properly supervise data management tasks for dataspace. In this chapter, we studied how to profile a KG by proposing a minimalization-based summarization model to support domain experts in better understanding the semantics of relations of a KG. Based on the experimentation we show that our summarization framework is able to provide both *compact* and *informative* summaries for a given KG. We showed that using ABSTAT framework, summaries are more compact than the ones generated from other models and they also help humans (or algorithms, as well) to gain insights about the semantics of underspecified relations in the KG.

---

<sup>10</sup><http://lov.okfn.org/>



# 7

## Conclusion

### 7.1 Summary of the Dissertation

---

This dissertation studied how to enrich the schema of Knowledge Graphs (KGs) with domain specific facets extracted from a vast amount of structured sources, providing interactive methods to domain experts in charge of maintaining a dataspace to ensure the adequate quality of the data. We focused on data management settings that imply the incremental integration of independent *data sources* into a *dataspace* [39], where a KG provides a structured machine readable representation of data on top of which advanced data access features are built, and where the integration process is managed and supervised by domain experts. This dissertation focuses on two main aspects:

- **Domain specificity.** This dissertation proposed a facet extraction and interpretation approach that incorporate the notion of domain specificity. We leverage already established mappings between source and KG types to extract domain specific facets, and propose an approach to provide a domain specific interpretation of them, by re-using relations already defined in the KG.
- **Enhanced Domain Expert Supervision.** By mean of the KG summarization approach presented in this dissertation, domain experts can profile and inspect the KG to understand the usage of relations in the data. This acquired understanding is crucial in order to properly supervise the approaches presented in this dissertation.

## 7. CONCLUSION

---

### Domain Specific Facet Extraction

Chapter 4 introduced an automatic, domain specific facet extraction approach, which supports domain experts in the definition of significant domain specific facets. Facets are specialized relations aimed at model the salient characteristics of entities from specific domains (e.g., news, actors, or wine bottles), and thus the technical problem tackled in this contribution accounts to the extraction of salient domain specific relations. The basic intuition behind the proposed approach is to reuse the representation of source instances provided by domain specific data sources. In particular, we leverage mappings established between source and KG types, to suggest meaningful, domain specific facets for a given KG type, based on *Taxonomy Layer Distance* (TLD), a novel metric used as clustering distance metric for grouping together facet values. Through our approach we are not only able to extract meaningful facets, but also to populate the KG with the relative faceted assertions about instances, thus supporting the provision on advanced data access features with an enhanced user-experience.

### Specificity-based Interpretation of Facets

Chapter 5 discussed how to interpreting extracted facets by *annotating* them with relations holding between KG instances of given domain and facet values. Given a facet, the proposed approach derives a set of candidate relations from KG and ranks them considering how much their semantics is similar to the semantics of the facet, focusing on domain specificity. The proposed approach handles the intrinsic ambiguity of facets by annotating them with relations that are more specific with respect to their facet domain. Our approach effectively quantifies the specificity of relation by considering its extensional semantics, that is the usage of the relation in the KG with respect to the different domains of knowledge conceptualized by KG types.

### Knowledge Graph Profiling

Chapter 6 introduced ABSTAT: a KG *summarization* framework that provides an abstract view of KG relations. ABSTAT is capable to provide an overview of the specificity of relations of a KG. Such information is crucial to enable a proper supervision of the extraction and annotation phases by domain experts. Due to the proposed minimalization based approach, ABSTAT is able to provide both compact and informative summaries for a given KG. We showed that using the ABSTAT framework, summaries

are more compact than the ones generated from other related models and they also help humans (or algorithms) to gain insights about the semantics of relations in the KG.

## 7.2 Future Work

---

Several future lines of work originate from the contributions presented in this dissertation. The rest of the section summarizes the main ones.

### “Local” Improvements

The three proposed approaches can be independently extended and improved along different directions. Firstly, advanced NLP techniques can be used to improve the facet extraction approach, by normalizing source types considering different lexicalizations (as discussed in Section 4.5). Secondly, the definition of the specificity of a relation with respect to a facet domain (discussed in Chapter 5.2) can be adapted to consider KGs with deep and specialized subtype graphs (as pointed out in Section 5.5). Thirdly, we plan to conduct the user study described in Section 6.4 in large scale, thus including more users with different background characteristics in order to analyze in details which is the target group of users for which ABSTAT is more useful. Also, we plan to complement our coverage-oriented approach with relevance-oriented summarization approaches based on connectivity analysis as discussed in Section 6.5.

### Table Annotation

As discussed in Section 5.8, the facet annotation approach can in principle be applied to the problem of relation annotation in tabular data. To this end, a preliminary work has been done by including an adapted version of the facet annotation approach into a table annotation tool, namely STAN<sup>1</sup>. Further investigation and additional experiments are needed, but we believe that this represents one of the most interesting directions of future work originating from the contributions of this dissertation.

---

<sup>1</sup><http://stan.disco.unimib.it>

## 7. CONCLUSION

---

### **Bring Explicit Feedback into the Picture**

The approaches discussed in this dissertation are data-driven, and are conceived to be included in a workflow where the validation of domain experts at all the stages is crucial. Explicit feedback gathered from domain experts validation represents an invaluable information that can be used to refine the interpretation phase. In particular, we envision a system that learns how to better capture the domain specificity of KG relations in the annotation of facets, based on explicit feedback from domain experts on its past effectiveness. Towards this end, it would be interesting to investigate to which extent the facet annotation approach can benefit from Schema and Ontology matching literature, where the development of interactive matching systems based on explicit feedback loops recently gained attention with the advent of Crowd Sourcing platforms and paradigms (e.g., [30, 34, 121, 14]).

### **Consider Data Access Patterns from End-users**

The representation of data provided by specialized sources is not the only source of valuable information that can be leveraged in order to extract domain specific facets [110]. As the ultimate purpose of DI application is to make integrated data searchable and accessible to end-users, the understanding of *how* end-users search and browse the data may provide useful insights on how to represent dataspace instances. In other words, a domain specific representation may emerge not only from data, but also from *usage*. For example, the analysis of full text queries submitted by users with exploratory information needs (e.g., the full text query “android mobile phones”), may reveal interesting patterns useful to identify what are the salient characteristics of instances in particular domains (e.g., the operating system for mobile phones). We believe that our facet extraction approach can gain a huge benefit from the study of frequent patterns in the access of integrated data. Towards this end, however, it is crucial to develop effective and efficient methods to analyze big, potentially noisy data access logs.

### **Reuse of Summaries**

We envision a large scale application of our summarization approach. As ABSTAT focuses on RDF KGs, we would like to run our summarization framework on KGs of the Linked Open Data cloud. Such large scale analysis of well known KGs would poten-

tially unveil commonly used data representation patterns from independent sources. This would particularly benefit domain experts, especially in providing a domain specific interpretation of facets. With “global” information about common patterns of use of relations from shared vocabularies, domain experts will potentially get better insights about their most commonly adopted semantics.

### **Support for Multilingual Dataspaces**

In this dissertation we didn’t consider the case where data provided by the sources is lexicalized in different languages. The facet extraction approach presented in this dissertation does not leverage any language-specific feature in the extraction of facets, and can be easily applied to different languages other than English (see 4.5). From the other side, the facet interpretation approach presented in Chapter 5 implicitly assumes consistency between the language in which facet values and KG instances are lexicalized. In order to extend such approach to the multilingual case, one would have to investigate to which extent our approach can benefit from the literature in Cross-Lingual Schema and Ontology Matching (e.g., [45, 97]). We believe that this would particularly benefit the application of the approaches discussed in this dissertation to scenarios that require the integration of multilingual data, as for example integrating Open Data released by institutions from different countries.







# Product Autocomplete

## A.1 Overview

---

As pointed out in Chapter 2, a rich and domain specific representation allows a DI application to provide advanced search and browse features over the KG instances. This chapter describes a case study from the eCommerce domain where KG types and facets are leveraged in order to provide an advanced Product Autocomplete feature for eMarketplaces. The Product Autocompletion system, which is named COMMA, described in this chapter is an example of the impact of the contributions of this dissertation, evaluated on the real world scenario of an Italian eMarketplace.

This chapter is organized as follows: Section A.2 frames the contribution described in the chapter in the context of the eCommerce domain. The autocompletion problem and the is discussed in Section A.3; the matching algorithm that constitutes the core of COMMA and that leverages domain specific representation of instances is described in Section A.4; details about the implementation and the deployment in a real world scenario of the proposed Product Autocomplete system are given in Section A.5; experiments carried out to evaluate the system are discussed in Section A.6, and a summary of the contribution described in this chapter is given in Section A.8.

## A.2 Autocomplete Interfaces in eCommerce

---

An ordinary eMarketplace typically performs a matchmaking activity between

## A. PRODUCT AUTOCOMPLETE

---

search queries submitted by the users and a set of products. Maximizing the *accuracy* and *coverage* of the matchmaking algorithm over search queries is a crucial aim that a successful eMarketplace must achieve. Providing an autocompletion interface is a recent but already established approach to improve product search features. For particular DI applications such as CSE (Comparison Shopping Engines), providing an autocompletion feature *as-a-service* to small/medium client eMarketplaces (i.e., data sources of the dataspace of a CSE, consistently with terminology from Chapter 2), often characterized by a poor retrieval features, represents an interesting business opportunity.

An autocompletion interface can perform two types of autocompletion operations: it can complete the *query* that the user is expressing by proposing a set of queries that are most relevant and reasonable to the query fragment already typed by the user; this type of autocompletion can be called *query-oriented* [23, 11, 40, 36]. The autocompletion interface can instead preview the *results* of the query fragment that is being typed by the user; this type of autocompletion can be called *result-oriented* [12, 25, 52, 125, 80].

Most of the autocompletion interfaces proposed so far in the eCommerce domain (e.g., Amazon, and Bing, Google and Yahoo Shopping) are based on query-oriented approaches. In fact, all the above CSEs integrate product offers coming from several eMarketplaces, achieving an almost complete coverage of the products available on the market. Remarkably, the advanced search features provided by such CSE are based on the structured information provided by their KGs. However, when a query is submitted to a single small/medium-sized eMarketplace (i.e., a data source of the CSE), correctly completing a query that would not lead to any or poorly ranked results, would not be effective. In this cases, result-oriented autocompletion can be highly effective, by previewing only product offers available in the eMarketplace, and by providing an insight on the internal working of the search functionality. We focus on the latter scenario, investigating a novel matching and ranking technique to support result-oriented autocompletion for small/medium eMarketplaces.

One of the main problems to address for an Autocompletion System (AS) is to seamlessly handle the different types of query that can be submitted by a user. Autocompletion techniques that adopt string-based matching algorithms and consider offers' titles and descriptions, can provide accurate results when users submit queries targeted to specific products (e.g., "Samsung i7500"), and they can be made error-tolerant by adopting approximate matching methods [90, 25, 52] over misspelled terms (e.g., "Samsong 7500"). However, user queries are often not really aimed at describing a specific product that is being searched but they are rather aimed at *exploring* the set of

available products or offers, looking for products of a given category (e.g., MobilePhone) or characterized by specific features (e.g., Android as value of the facet operatingSystem). Sometimes, more targeted and more generic keywords are even mixed in a *hybrid* type of query (e.g., “Samsung Android”).

Product types from the KG of the CSE and the most significant product features (i.e., facets) represent an invaluable source of information for handling *exploratory* and *hybrid* queries. Indexing the terms contained in this representation enables an auto-completion interface to provide results also for *exploratory* and *hybrid* queries but obtaining a relatively poor ranking, as shown by experiments discussed later on. For sake of clarity, we will call semantic techniques any matching technique aimed at handling this representation (types and facets) in a different way from pure textual descriptions, in the vein of research carried out in semantic search [79, 139].

### A.3 Problem Definition

---

Consistently with the terminology introduced in Chapter 2, an offer  $o$  (i.e., a named instance of the KG) is associated with one lexicalization  $lex(o)$  that shortly describes it in natural language (e.g., the string “Samsung Galaxy Tab 16 GB Android 2.2”). An offer is instance of exactly one KG type  $C$  (e.g., Tablet) and it is characterized by a set  $\mathcal{F}$  of *faceted assertions* describing the technical features of the offer, which are represented as

$$F(o, v)$$

where  $v$  is the *facet value* (e.g., mp3, which can be for example a value of the faceted assertion supportedAudio( $o$ , mp3)). An offer is thus represented as a triple

$$o = \langle lex(o), C, \mathcal{F} \rangle$$

where  $lex(o)$  is the offer’s lexicalization,  $C$  is a the type whose  $o$  is an instance of, and  $\mathcal{F} = \{F_1(o, v_1), \dots, F_n(o, v_n)\}$  is a set of faceted assertions.

#### The Autocompletion Problem

A result-driven AS presents a set of  $k$  results that are most relevant to keyword-based query typed by the user, by processing the query at each interaction of the user with the system. Observe that every *query fragment* (a query where the last keyword typed

## A. PRODUCT AUTOCOMPLETE

---

in by the user is a string representing a word fragment, e.g., “smartphone nok”) is considered an input query by the AS.

In order to be effective and perceived as instantaneous the autocompletion operation must be completed in a relatively short time span (i.e., maximum 100 ms) [85]: this constraint represents a serious limit to the possibility of exploiting complex techniques to improve the accuracy and coverage of the autocompletion results. Since there is an element of distraction caused by User Interface interrupts, which can overcome the user while typing the query, the high quality of the data retrieved by the AS must justify the distraction caused to the user by the AS [7]. An AS has therefore to fulfill both *efficiency* and *effectiveness* requirements, and since an improvement in the latter usually has a negative impact on the former, finding a good trade-off between efficiency and effectiveness is a major goal for matching algorithms developed in this context.

*Targeted* queries (e.g., “samsung galaxy tab”) are submitted by users that look for a specific product; the keywords used in this case usually point to specific terms used in offer lexicalization (e.g., brand and model). These queries can be handled using well known exact (e.g., prefix matching) and approximate matching techniques [90], which can support provide results also when query terms are misspelled [25, 52]. However, even approximate matching techniques would be unsuccessful when an *exploratory* query is submitted. Consider a query such as “Tablet PC with Office Mobile” which refers to a class of products rather than to a specific product. In this query, neither the keywords nor any string approximately matching these keywords occur in the offers’ lexicalization.

Instead, many keywords used in *exploratory* queries describe product types (e.g., TabletPC) and/or technical features. These pieces of information are often available from types and facet values. Furthermore, the distinction between *targeted* and *exploratory* queries is not sharp; in fact queries such as “samsung tab office mobile” can contain terms referring to specific products as well as to generic facet values; we will refer to these queries as *hybrid* queries. Using matching algorithms that specifically leverage well-structured representation of offers provided by the KG (i.e., semantic matching) can therefore play a crucial role in effectively answering *exploratory* (and *hybrid*) queries. One of the greatest challenges for a result-driven AS seamlessly processing all the above types of queries consists in combining syntactic and semantic matching so that the capability of handling more types of queries (improving the coverage of AS) does not lower the quality of the results returned when *targeted* queries are submitted.

---

## A.4 COMMA

---

The autocompletion approach proposed in this chapter and implemented by COMMA consists of three main phases as sketched in Figure A.1:

- **indexing (offline)**: the structured information from the KG that characterizes the offers is indexed along three different descriptive dimensions: lexicalizations, types and faceted assertions;
- **filtering (online)**: given a query fragment  $q$  submitted by the user, the matching algorithm applies three filters, one for each descriptive dimension; one filter selects offers whose *lexicalization* match with the query (syntactic matching); one filter selects offers whose *types* match with the query (semantic matching); one filter selects offers whose *facet values* match with the query (semantic matching); the results of the three filters are combined by means of a set union returning a final set of matching offers;
- **ranking (online)**: several scoring methods are applied to rank the offers selected by the filters; local scoring methods evaluate the relevance of each offer to the query by considering different criteria such as their popularity, the strength of the syntactic match and the semantic relevance; the local scores are combined in a final relevance score, which is used to rank the offers and define the list of top-k relevant offers.

Ranking is applied after filtering because most of the scores applied in ranking reuse the results of matches discovered during filtering; moreover, this approach allows to limit the number of items to which scoring functions are applied, achieving a better efficiency.

### A.4.1 Indexing

In order to index the lexicalizations, types and faceted assertions associated with each offer, three inverted indexes are built:  $I^{lex}$ ,  $I^T$  and  $I^F$  respectively denote the lexicalization, type and facet indexes. In the facet index  $I^F$  only facet values (i.e., the objects of the the domain specific relation conceptualized by the facet) are indexed under the assumption that the names of the facets are not very significant for searches (e.g., a user is more likely to search for “samsung phone”, instead of “brand samsung phone”).

## A. PRODUCT AUTOCOMPLETE

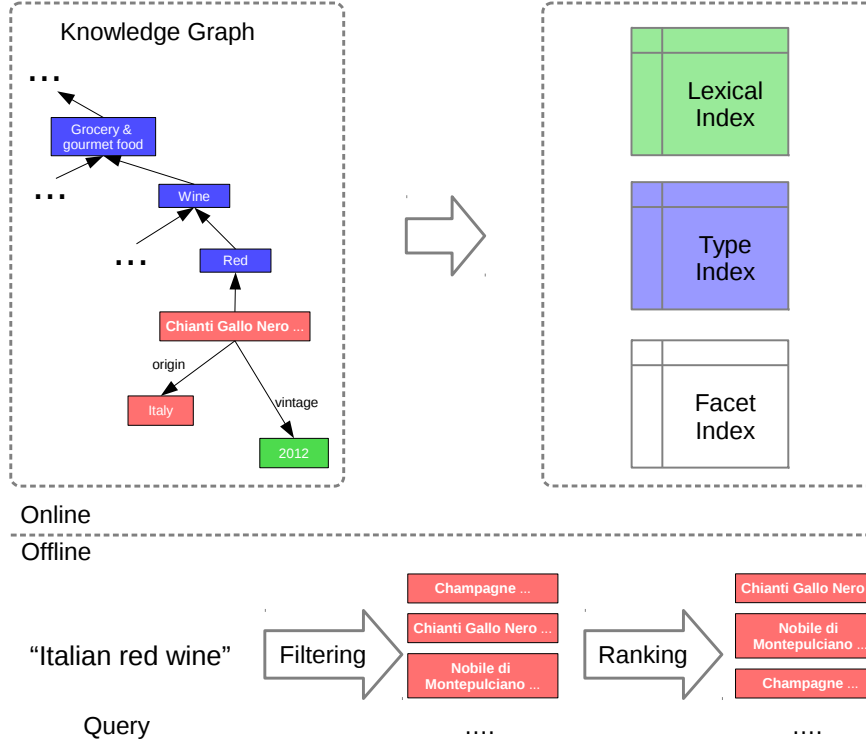


Figure A.1: Overview of the matching algorithm.

All the indexes are built by extracting every term occurring in the indexed string (respectively the lexicalization, the type identifier and the facets' values); we tokenize the strings representing codes (e.g., from "Nokia c5-03" we extract the three terms "nokia", "c5" and "03"); terms are stemmed and stop-words are removed. We call *lexicalization terms*, *type terms* and *facet value terms* the elements belonging respectively to the sets  $I^{lex}$ ,  $I^T$  and  $I^F$ .

### A.4.2 Syntactic and Semantic Filtering

Let  $O_{SYN}^q$ ,  $O_{TYPE}^q$ , and  $O_{FAC}^q$  be the set of offers whose textual descriptions (lexicalizations, in our case), types and facet values respectively match a query  $q$ . The set  $O^q$  of offers matching a query  $q$  can be defined as follows:

$$O^q = O_{SYN}^q \cup O_{TYPE}^q \cup O_{FAC}^q.$$

In other words, all the offers that match the input query along at least one dimension are selected in the filtering phase. The rest of this section explains how each

matching set is obtained.

### Syntactic Filters

The set  $O_{SYN}^q$  of the offers that syntactically match against a query fragment is computed by performing a set union of the results of two filters: a prefix-based (*pref*) string matcher, and an approximate string matcher (*asm*) based on normalized edit distance (*nedit* [90]) and a filtering threshold  $th^{asm}$ . We combine the results of these two filters to handle keyword fragments together with complete keywords with possible misspellings. In fact, using only an *asm* matcher with a reasonably high threshold would lead to poor results when matching short prefixes of long terms.

A lexicalization term  $l \in I^{lex}$  *pref-matches* an input keyword fragment  $k$ , iff  $k$  is a prefix of  $l$ ; a lexicalization term  $l \in I^{lex}$  *asm-matches* an input keyword fragment  $k$ , iff  $nedit(k, l) \leq th^{asm}$ . An offer  $o$  *pref-matches* (*asm-matches*, respectively) an input keyword fragment  $k$ , iff there is at least a lexicalization term  $l \in o$  such that  $l$  *pref-matches* (*asm-matches*)  $k$ . An offer  $o$  *pref-matches* (*asm-matches*, respectively) a query  $q = \{k_1, \dots, k_n\}$  if and only if  $o$  *pref-matches* (*asm-matches*) *every* keyword  $k_i \in q$ , with  $1 \leq i \leq n$ . The set  $O_{SYN}^q$  of offers syntactically matching an input query  $q$  is defined as the union of all the offers *pref-matching*  $q$  and all the offers *asm-matching*  $q$ .

**Example 1.** Suppose to have a query  $q = \text{"nokia c"}$ , and two offers named "Nokia c5-03" and "Nokia 5250 Blue"; "Nokia c5-03" syntactically matches  $q$  because "nokia" *asm-matches* the keyword "Nokia" and "c" *asm-matches* the keyword fragment "c" (instead, observe that this is not a *pref-match* because "nokia" does not *pref-match* "Nokia"); instead "Nokia 5250 Blue" does not syntactically match  $q$  because there is no term in the offer lexicalization *pref-matching* or *asm-matching* the keyword fragment "c".

### Semantic Filters

The goal of the semantic filter is to identify a set of offers that are potentially relevant to a query where some type and/or facet value occur as keywords. We define two semantic filters, one based on type matching (*type-matching*), and one based on facet value matching (*fac-matching*). Before applying semantic filters, stemming and stop-words removal are applied to a query  $q$  submitted by the user, obtaining a new query, denoted by  $\bar{q}$  to which filters are applied.

Intuitively, the *type-matching* filter returns all the offers that are instance of some

## A. PRODUCT AUTOCOMPLETE

---

type matching a query  $\bar{q}$ . This set  $O_{TYPE}^{\bar{q}}$  is defined as follows. A type  $C$  matches a keyword  $k$  iff there exists a type term  $i \in I^C$  associated with  $C$  such that  $i = k$  (remember that type names can contain more terms, and that an indexed type term can be associated with more types); an offer  $o$  cat-matches a query  $\bar{q} = \{k_1, \dots, k_n\}$  iff its an instance of a type  $t$  that matches at least one keyword  $k \in \bar{q}$ .

Intuitively, the *fac-match* filter returns all the offers characterized by some facet value matching a query  $\bar{q}$ ; this set  $O_{FAC}^{\bar{q}}$  is defined as follows. A facet value  $F(o, v)$  matches a keyword  $k$  iff there exists a facet value term  $i \in I^F$  associated with  $v$  such that  $i = k$  (remember that only facet values are considered in the matching process); an offer  $o$  fac-matches a query  $\bar{q} = \{k_1, \dots, k_n\}$  iff it is annotated with *at least* one facet  $f$  that matches at least one keyword  $k \in \bar{q}$  (remember that an offer can be characterized with several facet values). Given a query  $q$ , we also identify the sets  $T^q$  and  $\mathcal{F}^q$ , respectively representing the set of all the types and facet values that match with at least a keyword of  $k$ .

**Example 2.** Suppose to have a query  $q$ ="players with mp3", which is transformed in the query  $\bar{q}$ ="player mp3" after stemming and stop-words removal; since the type Mp3Players matches the query, all the offers that are instances of this type are included in the  $O_{TYPE}^{\bar{q}}$  set; since the value of several faceted assertions such as  $\text{compression}(o, \text{mp3})$ ,  $\text{audioFormat}(o, \text{mp3})$ ,  $\text{recordingFormat}(o, \text{mp3})$ , match  $\bar{q}$  as well, every offer associated with any of these facets is included in the  $O_{FAC}^{\bar{q}}$  set.

Observe that while an offer is required to syntactically match all the terms in the query, an offer can semantically match only one keyword in order to be considered in the semantic matches for that query. The reason for such a difference is that, while for precise queries targeted to retrieve offers based on their lexicalizations the user has more control on the precision of the results (under the assumption that the user knows what he/she is looking for), for *exploratory* queries containing generic type and facet terms we want to consider all the offers potentially relevant to these vaguer terms. Although the semantic match significantly expands the set of matching offers, the ranking process will be able to discriminate the offers that are more relevant to the given query.

### A.4.3 Ranking and Top-k Retrieval

Several criteria are used to rank the set of matching offers. Each criterion produces a local relevance score; all the local scores are combined by a *linear weighted function*



which returns a global relevance score for every matching offer; the top-k offers according to the global score are finally selected and presented to the user.

### Basic Local Scores

Three basic local scores are introduced to assess the relevance of an offer with respect to a query submitted by a user.

**Popularity:**  $pop(o)$  of an offer  $o$  is a normalized value in the range  $[0, 1]$  that represents the popularity of an offer  $o$  based on the number of views the offer received. An absolute offer popularity score is assessed offline analyzing log files and it is frequently updated; the absolute popularity score is normalized at runtime by distributing the absolute popularity of the offers matching the input query into the range  $[0, 1]$ .

**Syntactic relevance:** offers retrieved by exact (prefix) matching should be rewarded with respect to the offers approximately matched (e.g., because the order of digits makes a difference in technical terms like product codes); we therefore define a syntactic relevance score that discriminates between the offers matched by the two methods. The syntactic relevance  $syn_q(o)$  of an offer  $o$  with respect to a query  $q$  assumes a value  $m$  if  $o$  is a pref-match for  $q$ ,  $n$  if  $o$  is a asm-match for  $q$ , and 0 elsewhere, where  $m > n$ .

**Semantic relevance:** semantic relevance is based on the principle that the more matching facet values occur in an offer, the more relevant this offer is with respect to the query. Semantic relevance does not consider types because the number of offers that are instances of specific types is generally high. The semantic relevance  $sem_q(o)$  of an offer  $o$  to an input query  $q$  is computed as the number of faceted assertions  $\mathcal{F}^o$  associated with  $o$  and matching with  $q$ , normalized over the total number of the faceted assertions whose value matches the query  $\mathcal{F}^q$ , according to the following formula:

$$sem_q(o) = \frac{|\mathcal{F}^q \cap \mathcal{F}^o|}{|\mathcal{F}^q|}. \quad (\text{A.1})$$

### Intersection Boost and Facet Saliency

When a user submits a *targeted* query (see Section A.3), the basic local scores introduced above perform quite well. When the matching is driven mostly by syntactic criteria, the accuracy of the syntactic match and the popularity become key ranking

## A. PRODUCT AUTOCOMPLETE

---

factors. However, *exploratory* queries are usually more ambiguous (each single keyword can match more types and more facet values at the same time, and matches for each keyword are combined by means of set union); in these cases, the semantic relevance score based on the matching facet values can be too weak, with matching types not even considered in the score.

**Example 3.** All the offers that instances of the type “vacuum cleaner” or annotated with the faceted assertion  $\text{typology}(o, \text{robot})$  match the query “robot vacuum cleaners”; however, some of these offers are about vacuum cleaners having  $\text{typology}(o, \text{robot})$ , while others are associated with kitchen tools having  $\text{typology}(o, \text{robot})$ . The ranking scores defined so far do not discriminate among the two kinds of offers because they do not consider their types; intuitively, offers describing vacuum cleaners of type robot should be better rewarded because they match on both the type and facet value dimensions.

Considering the general case when offers match the query according to more than one dimension, we define a new local score called *Intersection Boost*. This local score aims at rewarding offers that occur in many specific matching sets selected by different filters. Before formally defining the Intersection Boost score we introduce a new filter in addition to previously defined ones. In order to better assess the relevance of the offers, and in particular of the offers that have been matched by semantic filters, we introduce a filter which is based on of *Facet Domain Specificity*. Intuitively we define a special set of offers that consists of the ones characterized by assertions about facets particularly *specific* to some types (i.e., domains) matching the query submitted by the user. A facet is specific to a type if it is highly likely to characterize offers that are instances of that type.

Formally, the specificity of a facet  $F$  to a type  $C$  is defined by a function that represents the conditional probability of the occurrence of a facet  $F$  in the set of offers  $O^C$  belonging to the type  $t$ ; given also the set  $O^F$  of the offers characterized with the facet  $F$ ,  $\text{spec}$  is defined by the following formula:

$$\text{spec}(F, C) = \frac{|O^F \cap O^C|}{|O^C|}. \quad (\text{A.2})$$

The set  $\hat{\mathcal{F}}^q$  of facets specific to a query  $q$  is defined as the set of facets matching with  $q$  whose specificity to at least one type matching  $q$  is higher than a given threshold  $l$ :

$$\hat{\mathcal{F}}^q = \{F \in \mathcal{F}^q \mid \exists C \in C^q : \text{spec}(F, C) \geq l\}. \quad (\text{A.3})$$

Given a query  $q$ , the set  $O_{SPEC}^q$  of offers specific to  $q$  is defined as the set of offers that fac-match some facet  $F \in \hat{\mathcal{F}}^q$ .

Now we can introduce the Boosting with Facet Domain Specificity (BFS) score. Let  $M$  be the set of all the matching filters defined above (i.e., syntactic, type and facet match and Facet Domain Specificity) and  $M_q^o \subseteq M$  the subset of all matched filters for the offer  $o$  given the query  $q$ ; the  $boost_q(o)$  function can be formally defined as follows:

$$boost_q(o) = \frac{|M_q^o|}{|M|}. \quad (\text{A.4})$$

**Example 4.** Consider the query  $q$ ="robot vacuum cleaner". Suppose that the offer  $o_1$  with lexicalization  $lex(o_1)$  = "i-robot roomba", of type VacuumCleaners and characterized by a faceted assertion  $typology(o, robot)$  is matched by syntactic, type, facet and Domain Specificity. Therefore, since  $o_1$  is matched by 4 out of 4 filters,  $boost_q(o_1) = 4/4 = 1$ . On the other hand, suppose that the offer  $o_2$  with  $lex(o) =$  "kenwood cooking chef kitchen machine", instance of the type KitchenTools characterized by the faceted assertion  $typology(o, robot)$ , is only matched by facet filter. Thus,  $boost_q(o_2) = 1/4 = 0.25$ .

### Global Matching Score and Ranking

The global scoring function can therefore be defined considering the Intersection Boost with Domain Specificity. Formally, the *global score* can be defined by a function  $score_q$ :

$$\begin{aligned} score_q(o) = & w_{pop} \cdot pop(o) + w_{syn} \cdot syn_q(o) + \\ & + w_{sem} \cdot sem_q(o) + w_{boost} \cdot boost_q(o) \end{aligned} \quad (\text{A.5})$$

where all  $w_i$  with  $i \in pop, syn, sem, boost$  are in the range  $[0, 1]$  and sum up to 1.

### Top-k Selection and Presentation

The output of the matching algorithm is a ranked set  $O^q$  of offers matching a query  $q$ . To effectively present  $O^q$  to a user as result of an autocompletion operation we need to present a subset of the results, namely  $O^{q,k}$ , selected as the top- $k$  ranked offers according the *global score* function defined in Equation A.5. It must be noted that due to UI constrains, the list of results of the autocompletion operation is necessarily small and should only contain high quality completions [42, 7].

## A. PRODUCT AUTOCOMPLETE

---

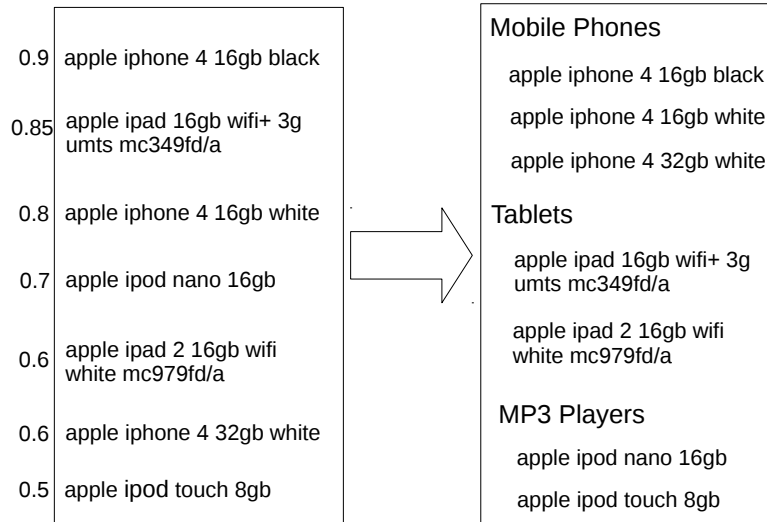


Figure A.2: The type grouping strategy.

A straightforward way to present the set  $O^{q,k}$  of the top-k results is to display a flat list of offers ordered by rank (*presentation by rank*). However, it has been shown that grouped presentation of autocompletion results helps users to perform search tasks more quickly [53, 3]. Thus, we defined a simple and effective *presentation by type* strategy that presents offers in  $O^{q,k}$  grouped by type. Offers within each group are ordered by rank and groups are ordered by the *score* of their top ranked offer. An example of the application of this grouping strategy is depicted in Figure A.2.

## A.5 Implementation and Deployment

---

COMMA has been implemented using the Java programming language, exploiting several functions of the Lucene search engine library<sup>1</sup>. As shown in Figure A.3, the system accepts the AJAX<sup>2</sup> calls from the client Web page and supplies results of the matching operation on offers in JSON<sup>3</sup> format. This deployment allowed for the provisioning of the autocompletion functionality as a remote service instead of requiring a deployment on the eMarketplace server. In this deployment configuration the size of the list of offers presented to the user as results of the autocompletion operation was set to 7, a good trade-off between compactness and coverage of the suggested

<sup>1</sup><http://lucene.apache.org/>

<sup>2</sup><http://www.w3schools.com/ajax/default.asp>

<sup>3</sup><http://www.json.org/>

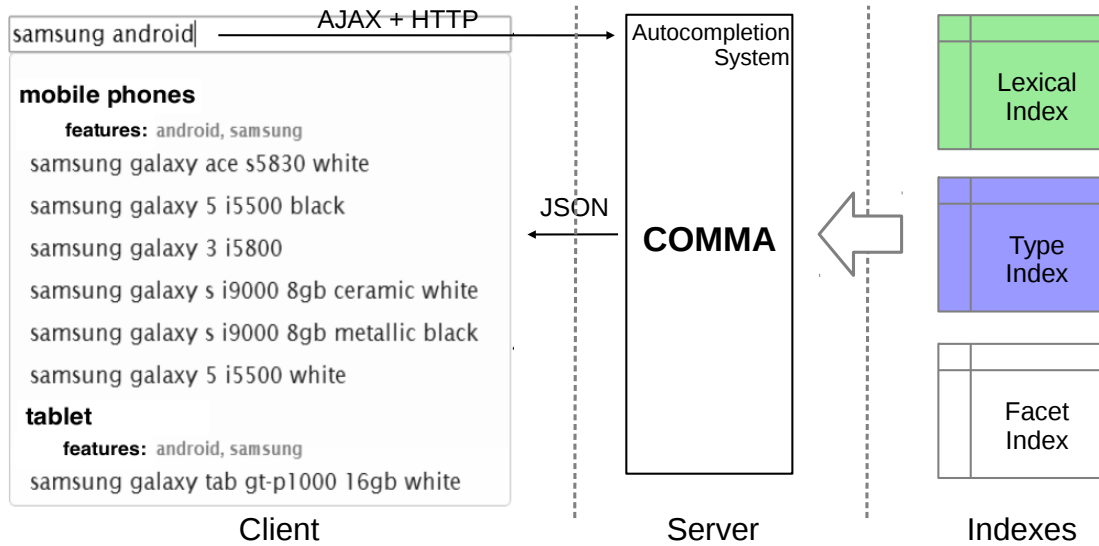


Figure A.3: COMMA evaluation deployment scenario.

results [115].

## A.6 Evaluation

We conducted several experiments to evaluate COMMA. In order to investigate the *effectiveness* of COMMA we evaluate the accuracy of the matching and ranking algorithms by 1) comparing COMMA with a baseline, defined by a syntactic matching approach based on the well-known TF-IDF weighting scheme [116] (TF-IDF from now on), 2) analyzing the behavior of COMMA under different parameter settings, and 3) studying the impact of different results presentation strategies. In order to evaluate the *efficiency* of COMMA, we investigate whether the AS satisfies the strict time constraints required by the domain, and to which extent it can scale up to handle a significant amount of data. Finally, we conducted experiments in a real world scenario, deploying COMMA to an Italian eMarketplace and measuring the impact on the *conversion rate*, i.e., the ratio of users that complete a purchase over the users that visit the eMarketplace.

The experiments were run on an Intel Core2 2.54GHz processor laptop, 4GB RAM, under the Ubuntu Linux Desktop 12.04 32bit operating system. For the real-world experiments only, COMMA has been deployed to an Ubuntu Linux Server 10.04 LTS 64bit, 2GB RAM, 2GB storage, virtualized machine.

### A.6.1 Effectiveness

In order to evaluate the effectiveness of our approach we evaluate the accuracy of the results ranked by COMMA with respect to an ideal rank, adopting the Normalized Discounted Cumulative Gain (nDCG) measure [2]. Discounted Cumulative Gain (DCG) is a well-known measure of effectiveness for ranking algorithms adopted in Information Retrieval. The rationale of the measurement is that highly relevant documents (offers, in this case) should appear in the top positions of the list of results and, more generally, that the list should be ordered according to the ideal relevance of the documents to a query. The nDCG measure is obtained by normalizing the DCG measure according to the ideal DCG measure of the result list. Intuitively, a higher nDCG value corresponds to a better agreement between the results proposed by the system and an ideal ordering based on human judgments.

### Experimental Setup

To the best of our knowledge there is no available benchmark and dataset defined to evaluate a result-oriented AS, and in particular considering the types of queries addressed by COMMA. We used a dataset from a portion of the real KG of the Italian CSE TrovaPrezzi (see Section 1.1), which contains 5594 offers belonging to 92 different types and characterized with 463 different facets. The ideal ranks for 30 keyword-based queries (13 *targeted*, 13 *exploratory* and 4 *hybrid* queries) have been defined by collecting graded relevance judgments (adopting a 3-point Likert Scale from 0=Bad, to 2=Excellent) from 10 average users and aggregating the individual judgments into mean graded relevance scores.

All the queries have been manually constructed, considering frequent types of queries made by Italian eCommerce users. For example, the query “nako coolpix” (notice the misspelling) is classified as *targeted*, the query “nokia wifi” is classified as *exploratory*, and the query “lcd samsung” is classified as *hybrid* because all the keywords can refer both to the name and the facets of an offer. For each query the user was asked to fill in a questionnaire assigning a graded relevance judgment (adopting the previously introduced scale) to a set of distinct 20 offers from the dataset. This set contained 15 offers selected by a pool of experts and was enriched with the top 5 ranked offers according to TF-IDF<sup>4</sup>, which did not appear in the list selected by the experts already.

---

<sup>4</sup>An archive containing the dataset used for our experiments, including the questionnaire and the user judgments can be downloaded from <http://www.lintar.disco.unimib.it/COMMA2012/>

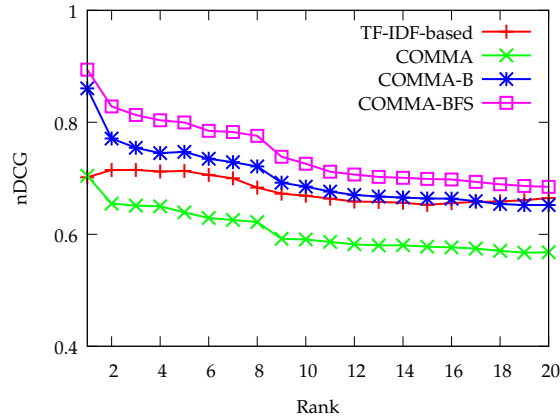


Figure A.4: nDCG for the 3 different configurations of COMMA compared to the baseline at different rank thresholds (all query types).

### Comparison with a Baseline

In the first experiment we consider as baseline the naive solution that can be proposed to the problem of responding to both *exploratory* and *targeted* queries: lexicalizations, types and facet values are indexed and the offers that have at least one term in common with the query (considering also misspellings) are selected and ranked according to a well-known weighting function. Here we use the result obtained using TF-IDF. The baseline is compared to three different configurations of COMMA: simple scoring, that is without ranking improvements (i.e., the *Intersection Boost* and *Facet Domain Specificity*), with *Intersection Boost* and without *Facet Domain Specificity* (COMMA-B), and with both *Intersection Boost* and *Facet Domain Specificity* (COMMA-BFS). Figure A.4 shows that all the configurations of COMMA that include at least one method for ranking improvement perform better than the baseline, at all the considered rank thresholds, while TF-IDF approach outperforms the basic COMMA algorithm. In the following we discuss more in depth the behavior of COMMA when different types of queries are considered.

Using both *Intersection Boost* and *Facet Saliency* in the global ranking function leads to significant improvements, especially if we consider the results for *exploratory* queries, plotted in Figure A.5a. The accuracy in ranking results in the top positions, which is noticeable for any type of query, is even more remarkable for *exploratory* queries; this behavior is particularly beneficial to an AS, which presents a limited number of results to the user. Rewarding the offers selected by more than one filter (*Intersection*

## A. PRODUCT AUTOCOMplete

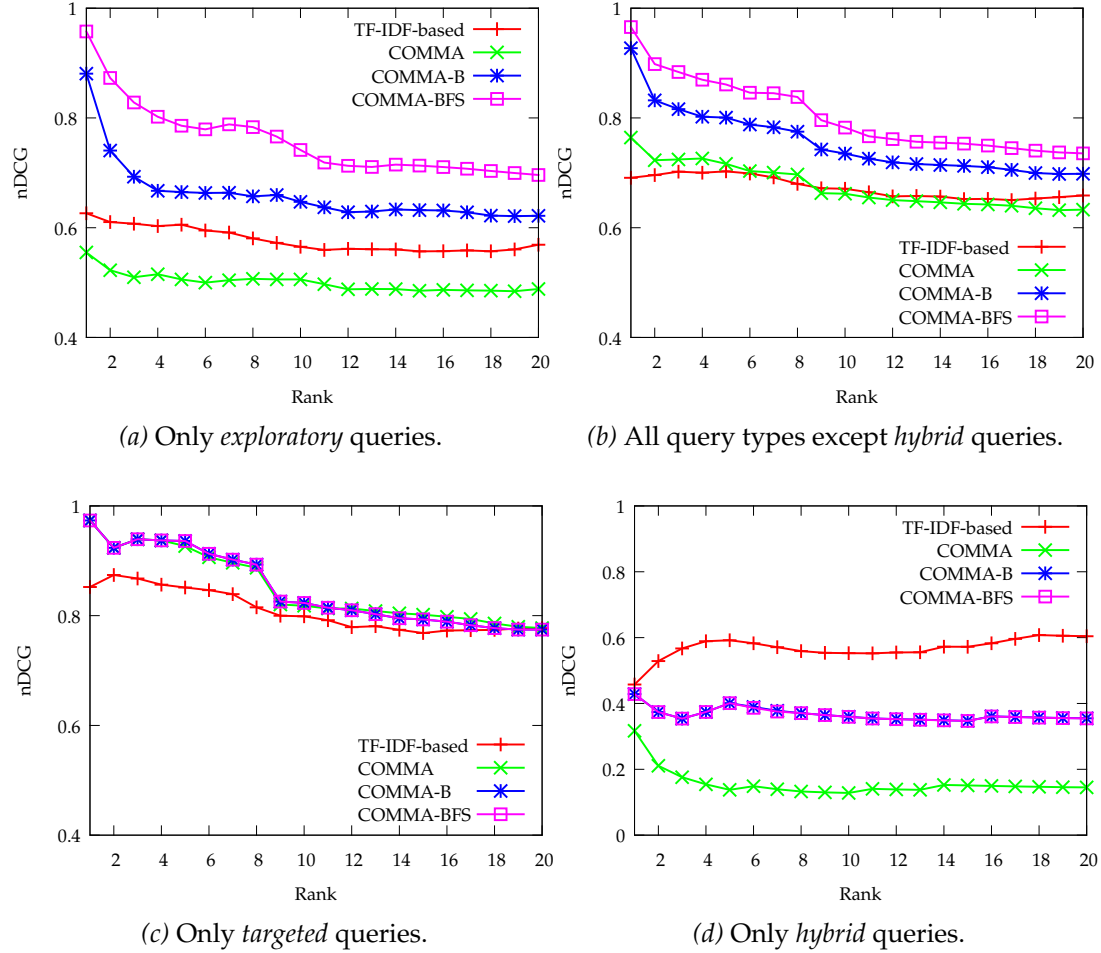


Figure A.5: nDCG for the 3 different configurations of COMMA compared to the baseline at different rank thresholds.

*Boost*), and particularly relevant because of the specificity of their faceted assertions to the query enables COMMA to be sufficiently selective achieve a high performance for higher rank thresholds.

In general, the COMMA configuration with simple scoring performs as good as the baseline when considering both *targeted* and *exploratory* queries, as plotted in Figure A.5b, while both *Intersection Boost* and *Facet Domain Specificity* improve the effectiveness of COMMA with negligible impact on performance in responding to *targeted* queries, as plotted in Figure A.5c. This can be explained by considering that *targeted* queries do not contain keywords referred to types and facet values.

The baseline performs better than COMMA only on *hybrid* queries, as shown in



Figure A.5d. The limited performance of COMMA on these queries can be explained by considering that *hybrid* queries contain keywords that refer to offer lexicalizations *and* keywords that refer to types or facet values. The syntactic filter used in COMMA is quite restrictive because it selects only offers that match every keyword in the query. The rationale behind this choice is to favor recall on *exploratory* queries, without sacrificing precision on *targeted* queries. Therefore given a *hybrid* query such as “galaxy mobile phone”, the COMMA syntactic filter does not provide any results; results for this query, i.e., all mobile phones, are returned by COMMA semantic filters (“mobile phone” does not occur in offers’ lexicalizations but it does in the lexicalization of types). Therefore, intersection boost does not reward enough offers such as SamsungGalaxyS3 with respect to other mobile phones.

### Parameters Tuning

In the second experiment, the effect of the weights  $w_{boost}$ ,  $w_{pop}$ ,  $w_{syn}$  and  $w_{sem}$  on the overall effectiveness of COMMA is evaluated. Figure A.6a presents the nDCG values obtained at fixed ranks (1, 10 and 20 respectively) for different configurations of COMMA, each one with a different assignment to  $w_{boost}$  (ranging from 0.1 to 1), and with an uniform distribution of the other weights according to the formula  $w_{pop} = w_{syn} = w_{sem} = \frac{1-w_{boost}}{3}$ . This experiment shows that the best results are obtained when  $w_{boost}$  is assigned values between 0.6 and 0.9. In other words, the higher the weight of the *Intersection Boost* factor is, the better COMMA ranks according to human judgment, with the exception of  $w_{boost} = 1$ . Given the optimal *Intersection Boost* weight, the best performing configuration of COMMA is the one that emphasizes the syntactic score, as depicted in the Figure A.6b. Although several criteria have to be introduced to refine the overall ranking scores (especially for offers selected by semantic filters, which extend the functionality of the AS) the syntactic score represents the second most important ranking factor after *Intersection Boost*.

### Effectiveness of Result Presentation

In the last experiment we investigate the effectiveness of the two results presentation approaches introduced in Section A.4: *presentation by rank* and *presentation by type*. The experiment shows that a grouped presentation of the outcomes of the autocompletion process does not decrease the effectiveness of COMMA, as depicted in Figure A.7. Usability tests would be necessary to understand if the presentation by type is actually

## A. PRODUCT AUTOCOMPLETE

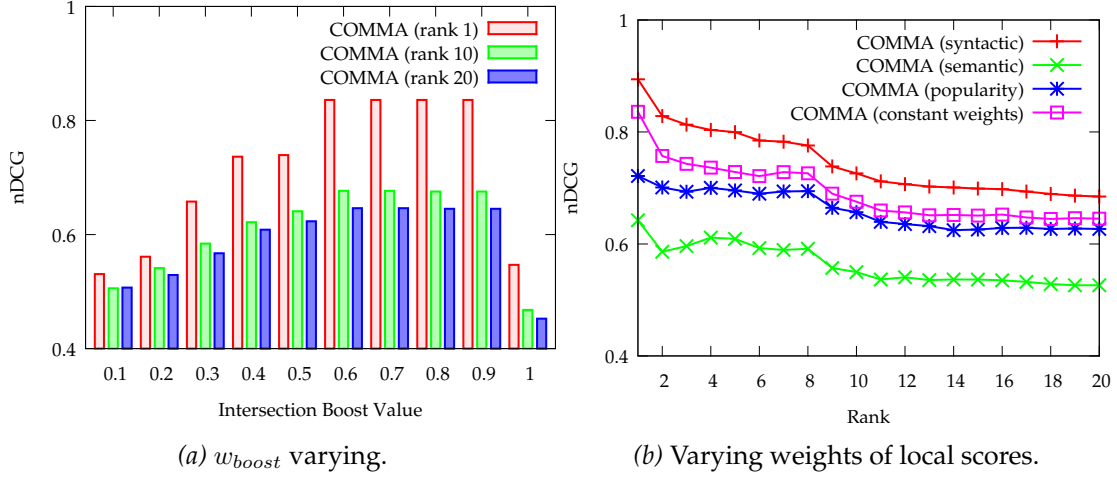


Figure A.6: Normalized DCG for different configurations of COMMA with different weights at fixed rank values.

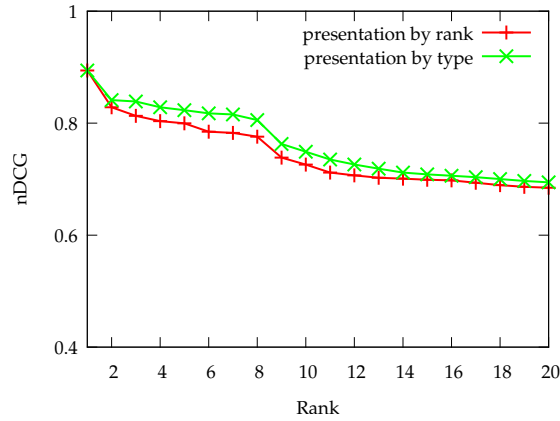


Figure A.7: nDCG at different rank thresholds for COMMA using two different ranking functions.

preferred by users in real world settings.

### A.6.2 Efficiency

We remind that the auto completion operation must take place in a relatively short time span (100 ms) in order to be perceived as instant by the user. We assess the efficiency of COMMA with respect to time elapsed for the computation of an auto completion operation. For these experiments we use a dataset of 30725 offers, belonging to 206 different types, characterized with 936 different facets. In order to analyze the

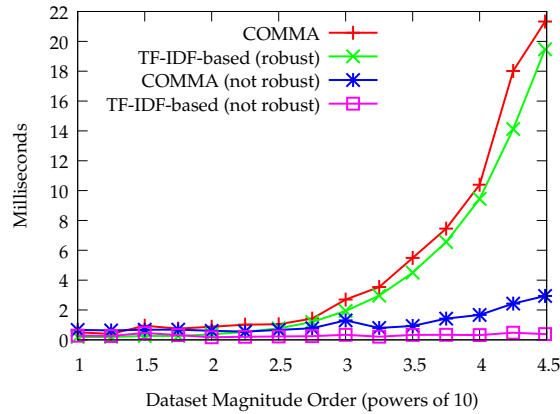


Figure A.8: Average execution times with set of offers of different size.

scalability of the COMMA approach, the execution times for completing a query are evaluated with sets of offers of different size. Test queries for this experiment are taken from a set of 200 queries of variable length chosen among the most common queries actually submitted by users of the Italian CSE TrovaPrezzi. All the results of the single query computation are merged to a mean elapsed time measure. Results of this test are shown in Figure A.8: COMMA is compared to the approach based on TF-IDF, and to other configurations of COMMA that does not include any approximate string matching algorithm (referred to as “not robust” in Figure A.8).

The results of this experiment show that COMMA is able to respect the strict requirements of an AS, completing the autocompletion operation in less than 25 ms and leaving a reasonable time for network overhead. A second kind of analysis is related to the impact of the semantic filters and ranking scores adopted by COMMA on the overall execution times, which can be estimated by considering the difference between COMMA and TF-IDF. Within the experimented field, this difference remains quite limited and it does not grow significantly with the growth of the size of the dataset. Finally, the computational costs of the approximate string matching algorithms dominate the overall costs of computation. This problem is due to a sub-optimal implementation of the approximate string matching algorithm in the Lucene framework used in the experiments; however, an improved implementation has announced to be included in Lucene, the overall performance of COMMA is expected to significantly improve.

## A. PRODUCT AUTOCOMPLETE

---

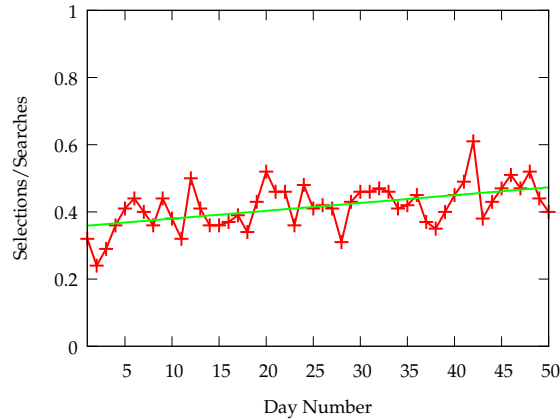


Figure A.9: Ratio between users AS result selection and eMarketplace search engine usage.

### A.6.3 Real World Experimentation

The experiments in real world scenario were run to evaluate COMMA from a business perspective. Two different experiments were run considering two different configurations of COMMA deployed to an Italian eMarketplace dealing with consumer electronics and photography related goods. The results reported in this section are not as extensive as the ones previously described due to confidentiality agreements with the eMarketplace company.

The goal of the first experiment was to evaluate the impact of the addition of the autocompletion function to the eMarketplace Web site, with particular reference to the effects on usability of the Web site search feature. The autocompletion function is a form of site adaptation and we were concerned that it could reduce the overall site usability. We used Google Analytics<sup>5</sup> to gather data about the usage of the autocompletion function, adopting a purely syntactic configuration of COMMA (we ran the test in the early phases of the work): results showed that during the test period (50 days) users increasingly employed the autocompletion function (considering the frequency of activation of the autocompletion function on the overall number of page views). Users also increasingly selected offers returned by the AS instead of submitting queries to the eMarketplace, as depicted in Figure A.9. During this experiment we found also that the 41.21% of the top 200 unanswered autocompletion queries were *exploratory* queries.

The percentage of *exploratory* unanswered queries from the last experiment con-

---

<sup>5</sup><http://www.google.com/analytics/>

firmly the importance of handling these queries for AS in the eCommerce domain. This intuition is further confirmed by the second experiment. In a later phase, we evaluated the impact of the full autocompletion function on the (previously introduced) conversion rate by means of an A/B test [58]<sup>6</sup> during a shorter time period (5 days, for a total of 7,339 visitors): the introduction of this function led to an increase of 6.67% of the conversion rate.

## A.7 Comparison with Prior Art

---

The problem of developing an effective and efficient AS has been analyzed in the literature from many different perspectives; nonetheless, AS applications in the eCommerce area are rarely described in the scientific literature and they are often protected by patents (see e.g. [96]). After an analysis of the features of the available commercial AS, we found that the only system that explicitly uses types and facets to support *query-driven* autocompletion is Bing Shopping: when a query matches one or more type the system helps the user to refine the query by adding (i) types (e.g., Mp3Players), and (ii) constraints over facet values. Also our *result-driven* approach matches keywords with relevant types and facet values. However, our approach is fully automatic and uses these matches to suggest meaningful results instead of suggesting query refinements.

In general, autocompletion techniques have been developed both for *finding information* and for *helping the user to express search queries*. In the field of Data Management the autocompletion problem has been interpreted as the problem of matching a query fragment with searchable data represented by strings, and different techniques of both exact and approximate string matching [90], such as *prefix matching* [12] *q-grams* [25] and *trie* based matching [52] have been proposed. Our approach can potentially leverage any of these techniques. However, the experimental results presented in Section A.6 show that an approach to autocompletion that employs only syntactic matching is not able to deal with *exploratory* and *hybrid* queries in a satisfactory way.

Autocompletion techniques have also been studied in the field of Semantic Search (see [79]). Hyvönen et al. formalize the autocompletion problem as the problem of matching a query to KG types [48]. From this perspective, the AS proposed in this dissertation can be classified as a form of *Semantic Autocompletion Search*. Autocom-

---

<sup>6</sup>The A/B test was carried out using Visual Website Optimizer A/B testing tool. More information can be found at <http://visualwebsiteoptimizer.com/>

## A. PRODUCT AUTOCOMPLETE

---

pletion in this area have been introduced to suggest types from is-a (or part-of) hierarchies [125], and to retrieve data characterized using different types [80]. Semantic autocompletion techniques are significantly related to the approach proposed in this dissertation, since COMMA exploits two different orthogonal representations of data to handle several query types. However, the above mentioned approaches to semantic autocompletion do not deal with the problem of ranking the proposed results (a full list of matches is returned).

The use of semantics in search-related tasks has been studied by different research communities [139], as testified by recent workshops hosted by different international conferences [54, 8, 140]. In this field large body of work has been devoted to the so called *Entity Search* (ES) task, that is the search for instances of a KG. ES is also known in traditional Information Retrieval community as *Ad-Hoc Object Retrieval* [109]. The goal of ES techniques is to retrieve and rank instances from large KGs such that they are more relevant to a keyword-based query submitted by users. The best performing approaches [18, 103] use an adaptation of the BM25F ranking model [114] that considers different sections of documents representing an entity, each one characterized by specific relational assertions and assigns different weights depending on the assertion where the terms matching the query appear, (e.g., assertion of the foaf:name or rdfs:label relations). A more recent approach models the instances using an improved generative language modeling framework [106] in order to preserve the semantics associated with instances without sacrificing retrieval effectiveness [94]. Despite being close to our approach, ES techniques do not consider time efficiency issues, such as those that must be considered when implementing an AS and do not leverage the co-occurrence of different instance representations (which can be mapped to facets and types in our domain) to rank results. Moreover, BM25F considers different sections of documents representing an instance, each one defined by values of specific relations. However, we consider all the faceted assertions (relational assertions in a ES context) and we do not assign different weights to specific relations.

The task of helping user in formulating queries to search engines has been also studied in the *Interactive Query Expansion* (IQE) research area. In this context autocompletion is interpreted as the problem of retrieving the terms that, when added to the already expressed query, improve the precision and recall of the results. The suggested terms can be retrieved considering their conditional probability of belonging to the same topic that the previously specified terms belong to [36]. Other ways to determine a set of terms to suggest is to consider reciprocal semantic relations such as *synonymy*, *hyponymy* or *hypernymy* [40] or to consider user behaviour in submitting

multiple queries [23]. A more recent approach, acknowledging the limits of purely syntactic approaches, uses query logs to evaluate the similarity between query contexts [11]. A query context consists of all the query terms previously stated by the user plus the initial letters of the term that the user is currently typing. All of these approaches are intrinsically different from the COMMA approach, since they all suggest queries (not results). Moreover COMMA deals with structured data instead of textual documents.

## A.8 Summary

---

This appendix introduced COMMA, an approach for exploiting type and domain specific facets for result-oriented autocompletion. The presence of the domain specific representation is crucial in order to support the elaboration of *exploratory* queries, without decreasing the precision in the management of precise queries. The approach is formally described and evaluated according both to its effectiveness and efficiency, both in experimental and real world scenarios. The results show that COMMA outperforms purely syntactic approaches to autocompletion in presence of *exploratory* queries without causing a significant increase in computational costs. COMMA is example of how crucial a rich and domain specific representation of KG instances is, when providing advanced data access features to data of a dataspace.





# Bibliography

- [1] Ziawasch Abedjan, Johannes Lorey, and Felix Naumann. Reconciling ontologies and the web of data. In *CIKM*, pages 1532–1536, 2012.
- [2] Rakesh Agrawal, Sreenivas Gollapudi, Alan Halverson, and Samuel Jeong. Diversifying search results. In *WSDM*, pages 5–14, 2009.
- [3] Alia Amin, Michiel Hildebrand, Jacco van Ossenbruggen, Vanessa Evers, and Lynda Hardman. Organizing suggestions in autocompletion interfaces. In *ECIR*, pages 521–529, 2009.
- [4] Sören Auer, Jan Demter, Michael Martin, and Jens Lehmann. LODStats - An extensible framework for high-performance dataset analytics. In *EKAW (2)*, pages 353–362, 2012.
- [5] Sören Auer and Jens Lehmann. What have Innsbruck and Leipzig in common? extracting semantics from wiki content. In *ESWC*, pages 503–517, 2007.
- [6] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.
- [7] Brian P. Bailey, Joseph A. Konstan, and John V. Carlis. The effects of interruptions on task performance, annoyance, and anxiety in the user interface. In *INTERACT*, pages 593–601, 2001.
- [8] Krisztian Balog, David Carmel, Arjen P. de Vries, Daniel M. Herzig, Peter Mika, Haggai Roitman, Ralf Schenkel, Pavel Serdyukov, and Duc Thanh Tran. The first joint international workshop on entity-oriented and semantic search (JIWES). *SIGIR Forum*, 46(2):87–94, 2012.
- [9] Michele Banko and Oren Etzioni. The tradeoffs between open and traditional relation extraction. In *ACL*, pages 28–36, 2008.
- [10] Mohit Bansal, David Burkett, Gerard de Melo, and Dan Klein. Structured learning for taxonomy induction with belief propagation. In *ACL*, pages 1041–1051, 2014.
- [11] Ziv Bar-Yossef and Naama Kraus. Context-sensitive query auto-completion. In *WWW*, pages 107–116, 2011.

## BIBLIOGRAPHY

---

- [12] Holger Bast, Christian W. Mortensen, and Ingmar Weber. Output-sensitive autocompletion search. *Inf. Retr.*, 11:269–286, 2008.
- [13] Dave Beckett and Jan Grant. Semantic web scalability and storage: Mapping semantic web data with RDBMSes. Technical report, 2003.
- [14] Khalid Belhajjame, Norman W. Paton, Alvaro A. A. Fernandes, Cornelia Hedeler, and Suzanne M. Embury. User feedback as a first class citizen in information integration systems. In *CIDR*, pages 175–183, 2011.
- [15] Tim Berners-Lee. Relational databases on the semantic web. Technical report, 1998.
- [16] Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data - The story so far. *Int. J. Semantic Web Inf. Syst.*, 5(3):1–22, 2009.
- [17] Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. Dbpedia - a crystallization point for the web of data. *J. Web Sem.*, 7(3):154–165, 2009.
- [18] Roi Blanco, Peter Mika, and Sebastiano Vigna. Effective and efficient entity search in RDF data. In *ISWC (1)*, pages 83–97, 2011.
- [19] Christoph Böhm, Gjergji Kasneci, and Felix Naumann. Latent topics in graph-structured data. In *CIKM*, pages 2663–2666, 2012.
- [20] Ronald Brachman and Hector Levesque. *Knowledge Representation and Reasoning*. Morgan Kaufmann Publishers Inc., 2004.
- [21] Michael J. Cafarella, Alon Y. Halevy, Yang Zhang, Daisy Zhe Wang, and Eugene Wu. Uncovering the relational web. In *WebDB*, 2008.
- [22] Stephane Campinas, Thomas E. Perry, Diego Ceccarelli, Renaud Delbru, and Giovanni Tummarello. Introducing RDF graph summary with application to assisted SPARQL formulation. In *DEXA*, pages 261–266, 2012.
- [23] Huanhuan Cao, Daxin Jiang, Jian Pei, Qi He, Zhen Liao, Enhong Chen, and Hang Li. Context-aware query suggestion by mining click-through and session data. In *KDD*, pages 875–883, 2008.
- [24] Pierre-Antoine Champin, Geert-Jan Houben, and Philippe Thiran. Cross: An OWL wrapper for reasoning on relational databases. In *ER*, pages 502–517, 2007.

- [25] Surajit Chaudhuri and Raghav Kaushik. Extending autocompletion to tolerate errors. In *SIGMOD*, pages 707–718, 2009.
- [26] Michelle Cheatham and Pascal Hitzler. String similarity metrics for ontology alignment. In *ISWC*, pages 294–309, 2013.
- [27] Michelle Cheatham and Pascal Hitzler. The properties of property alignment. In *OM*, pages 13–24, 2014.
- [28] C. L. Philip Chen and Chun-Yang Zhang. Data-intensive applications, challenges, techniques and technologies: A survey on big data. *Inf. Sci.*, 275:314–347, 2014.
- [29] Dario Colazzo, François Goasdoué, Ioana Manolescu, and Alexandra Roatiş. RDF analytics: Lenses over semantic graphs. In *WWW*, pages 467–478, 2014.
- [30] Isabel F. Cruz, Francesco Loprete, Matteo Palmonari, Cosmin Stroe, and Aynaz Taheri. Pay-as-you-go multi-user feedback model for ontology matching. In *EKAW*, pages 80–96, 2014.
- [31] Aba-Sah Dadzie and Matthew Rowe. Approaches to visualising linked data: A survey. *Sem. Web J.*, 2(2):89–124, 2011.
- [32] Wisam Dakka and Panagiotis G. Ipeirotis. Automatic extraction of useful facet hierarchies from text databases. In *ICDE*, pages 466–475, 2008.
- [33] Zhicheng Dou, Sha Hu, Yulong Luo, Ruihua Song, and Ji-Rong Wen. Finding dimensions for queries. In *CIKM*, pages 1311–1320, 2011.
- [34] Songyun Duan, Achille Fokoue, and Kavitha Srinivas. One size does not fit all: Customizing ontology alignment using user feedback. In *ISWC (1)*, pages 177–192, 2010.
- [35] Martin Ester, Hans P. Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, pages 226–231, 1996.
- [36] Ju Fan, Hao Wu, Guoliang Li, and Lizhu Zhou. Suggesting topic-based query terms as you type. In *APWeb*, pages 61–67, 2010.
- [37] Christiane Fellbaum. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.
- [38] Alfio Ferrara, Andriy Nikolov, and François Scharffe. Data linking. *J. Web Sem.*, 23:1, 2013.

## BIBLIOGRAPHY

---

- [39] Michael J. Franklin, Alon Y. Halevy, and David Maier. From databases to dataspace: a new abstraction for information management. *SIGMOD Record*, 34(4):27–33, 2005.
- [40] Zhiguo Gong, Chan Wa Cheang, and Leong Hou U. Web query expansion by WordNet. In *DEXA*, pages 166–175, 2005.
- [41] Alon Y. Halevy. Why your data won't mix. *ACM Queue*, 3(8):50–58, 2005.
- [42] Micheline Hancock-Beaulieu. Experiments with interfaces to support query expansion. *J. Doc.*, 53(1):8–19, 1997.
- [43] John A. Hartigan and Anthony M. Wong. Algorithm AS 136: A k-means clustering algorithm. *Appl. Stat-J. Roy. St. C*, 28(1):100–108, 1979.
- [44] Marti A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *COLING*, pages 539–545, 1992.
- [45] Mamoun Abu Helou, Matteo Palmonari, and Mustafa Jarrar. Effectiveness of automatic translations for cross-lingual ontology mapping. *J. Artif. Intell. Res.*, 55:165–208, 2016.
- [46] Laurie J. Heyer, Semyon Kruglyak, and Shibu Yooseph. Exploring expression data: identification and analysis of coexpressed genes. *Genome res.*, 9(11):1106–1115, 1999.
- [47] Aidan Hogan, Andreas Harth, Alexandre Passant, Stefan Decker, and Axel Polleres. Weaving the pedantic web. In *LDOW*, 2010.
- [48] Eero Hyvönen and Eetu Mäkelä. Semantic autocompletion. In *ASWC*, pages 739–751, 2006.
- [49] Krzysztof Janowicz, Frank van Harmelen, James A. Hendler, and Pascal Hitzler. Why the data train needs semantic rails. *AI Magazine*, 36(1):5–14, 2015.
- [50] Marijn Janssen, Yannis Charalabidis, and Anneke Zuiderwijk. Benefits, adoption barriers and myths of open data and open government. *Inform. Syst. Manage.*, 29(4):258–268, 2012.
- [51] Mustafa Jarrar and Marios D. Dikaiakos. A query formulation language for the data web. *IEEE Trans. Knowl. Data Eng.*, 24(5), 2012.
- [52] Shengyue Ji, Guoliang Li, Chen Li, and Jianhua Feng. Efficient interactive fuzzy keyword search. In *WWW*, pages 371–380, 2009.

- [53] Hideo Joho, Claire Coverson, Mark Sanderson, and Micheline Beaulieu. Hierarchical presentation of expansion terms. In *SAC*, pages 645–649, 2002.
- [54] Jaap Kamps, Jussi Karlgren, Peter Mika, and Vanessa Murdock. Fifth workshop on exploiting semantic annotations in information retrieval: ESAIR’12. In *CIKM*, pages 2772–2773, 2012.
- [55] Yu Kawano, Hiroaki Ohshima, and Katsumi Tanaka. On-the-fly generation of facets as navigation signs for web objects. In *DASFAA*, pages 382–396, 2012.
- [56] Craig A. Knoblock and Pedro A. Szekely. Exploiting semantics for big data integration. *AI Magazine*, 36(1):25–38, 2015.
- [57] Craig A. Knoblock, Pedro A. Szekely, José Luis Ambite, Aman Goel, Shubham Gupta, Kristina Lerman, Maria Muslea, Mohsen Taheriyani, and Parag Mallick. Semi-automatically mapping structured sources into the semantic web. In *ESWC*, pages 375–390, 2012.
- [58] Ron Kohavi, Roger Longbotham, Dan Sommerfield, and Randal M. Henne. Controlled experiments on the web: survey and practical guide. *Data Min. Knowl. Disc.*, 18:140–181, 2009.
- [59] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models - Principles and Techniques*. MIT Press, 2009.
- [60] Weize Kong and James Allan. Extracting query facets from search results. In *SIGIR*, pages 93–102, 2013.
- [61] Mathias Konrath, Thomas Gottron, Steffen Staab, and Ansgar Scherp. SchemEX - Efficient construction of a data catalogue by stream-based indexing of linked data. *J. Web Sem.*, 16:52–58, 2012.
- [62] Dimitris Kontokostas. Making sense out of the wikipedia categories (GSoC2013). <http://blog.dbpedia.org/?p=74>, 2013. Accessed: 2016-02-12.
- [63] Maksym Korotkiy and Jan L. Top. From relational data to RDFS models. In *ICWE*, pages 430–434, 2004.
- [64] Andreas Kupfer, Silke Eckstein, Karl Neumann, and Brigitte Mathiak. Handling changes of database schemas and corresponding ontologies. In *ER Workshops*, pages 227–236, 2006.

## BIBLIOGRAPHY

---

- [65] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, pages 282–289, 2001.
- [66] Heidi Lam, Enrico Bertini, Petra Isenberg, Catherine Plaisant, and Sheelagh Carpendale. Empirical studies in information visualization: Seven scenarios. *IEEE T. Vis. Comput. Gr.*, 18(9):1520–1536, 2012.
- [67] Andreas Langeegger and Wolfram Wöß. RDFStats - an extensible RDF statistics generator and library. In *DEXA*, pages 79–83, 2009.
- [68] Claudia Leacock and Martin Chodorow. Combining local context and wordnet similarity for word sense identification. In *MIT Press*, pages 265–283, 1998.
- [69] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morse, Patrick van Kleef, Sören Auer, and Christian Bizer. DBpedia - A large-scale, multilingual knowledge base extracted from wikipedia. *Sem. Web J.*, 2014.
- [70] Maurizio Lenzerini. Data integration: A theoretical perspective. In *PODS*, pages 233–246, 2002.
- [71] Juanzi Li, Jie Tang, Yi Li, and Qiong Luo. Rimom: A dynamic multistrategy ontology alignment framework. *IEEE Trans. Knowl. Data Eng.*, 21(8):1218–1232, 2009.
- [72] Tianyu Li, Pirooz Chubak, Laks V. S. Lakshmanan, and Rachel Pottinger. Efficient extraction of ontologies from domain specific text corpora. In *CIKM*, pages 1537–1541, 2012.
- [73] Xiao Li, Ye-Yi Wang, and Alex Acero. Extracting structured information from user queries with semi-supervised conditional random fields. In *SIGIR*, pages 572–579, 2009.
- [74] Girija Limaye, Sunita Sarawagi, and Soumen Chakrabarti. Annotating and searching web tables using entities, types and relationships. *PVLDB*, 3(1):1338–1347, 2010.
- [75] Xueqing Liu, Yangqiu Song, Shixia Liu, and Haixun Wang. Automatic taxonomy construction from keywords. In *KDD*, pages 1433–1441, 2012.
- [76] Vanessa Lopez, Christina Unger, Philipp Cimiano, and Enrico Motta. Evaluating question answering over linked data. *J. Web Sem.*, 21:3–13, 2013.

- [77] Jayant Madhavan, Shirley Cohen, Xin Luna Dong, Alon Y. Halevy, Shawn R. Jeffery, David Ko, and Cong Yu. Web-scale data integration: You can afford to pay as you go. In *CIDR*, pages 342–350, 2007.
- [78] Alexander Maedche and Steffen Staab. Measuring similarity between ontologies. In *EKAW*, pages 251–263, 2002.
- [79] Eetu Mäkelä. Survey of semantic search research. In *Seminar on Knowledge Management on the Semantic Web*, 2005.
- [80] Eetu Mäkelä, Kim Viljanen, Petri Lindgren, Mikko Laukkanen, and Eero Hyvönen. Semantic yellow page service discovery: The veturi portal. In *ISWC (Posters)*, 2005.
- [81] Laurent Mazuel and Nicolas Sabouret. Semantic relatedness measure using object properties in an ontology. In *ISWC*, pages 681–694, 2008.
- [82] Olena Medelyan, Steve Manion, Jeen Broekstra, Anna Divoli, Anna-Lan Huang, and Ian H. Witten. Constructing a focused taxonomy from a document collection. In *ESWC*, pages 367–381, 2013.
- [83] Sergey Melnik, Hector Garcia-Molina, and Erhard Rahm. Similarity flooding: A versatile graph matching algorithm and its application to schema matching. In *ICDE*, pages 117–128, 2002.
- [84] Nandana Mihindukulasooriya, Maria Poveda Villalon, Raul Garcia-Castro, and Asuncion Gomez-Perez. Loupe - An online tool for inspecting datasets in the linked data cloud. In *ISWC (Posters & Demo)*, 2015.
- [85] Robert B. Miller. Response time in man-computer conversational transactions. In *AFIPS (Fall, part I)*, pages 267–277, 1968.
- [86] Varish Mulwad, Tim Finin, and Anupam Joshi. Semantic message passing for generating linked data from tables. In *ISWC (1)*, pages 363–378, 2013.
- [87] David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Lingvist. Invest.*, 30(1):3–26, 2007.
- [88] Jay Nath. Reimagining government in the digital age. *National Civic Review*, 100(3):19–23, 2011.
- [89] Felix Naumann. Data profiling revisited. *SIGMOD Rec.*, 42(4):40–49, 2014.

## BIBLIOGRAPHY

---

- [90] Gonzalo Navarro and Mathieu Raffinot. *Flexible pattern matching in strings: practical on-line search algorithms for texts and biological sequences*. Cambridge University Press, 2002.
- [91] Roberto Navigli and Simone Paolo Ponzetto. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Art. Int.*, 193:217–250, 2012.
- [92] Roberto Navigli and Paola Velardi. Learning word-class lattices for definition and hypernym extraction. In *ACL*, pages 1318–1327, 2010.
- [93] Roberto Navigli, Paola Velardi, and Stefano Faralli. A graph-based algorithm for inducing lexical taxonomies from scratch. In *IJCAI*, pages 1872–1877, 2011.
- [94] Robert Neumayer, Krisztian Balog, and Kjetil Nørkvåg. On the modeling of entities for ad-hoc entity search in the web of data. In *ECIR*, pages 133–145, 2012.
- [95] Csongor Nyulas and Samson Tu. Datamaster – a plug-in for importing schemas and data from relational databases into protégé. In *10th International Protégé Conference*, 2007.
- [96] Ruben E. Ortega, John W. Avery, and Robert Frederick. Search query autocompletion. Patent, 05 2003. US 6564213.
- [97] Sebastian Padó and Mirella Lapata. Cross-lingual annotation projection for semantic roles. *J. Artif. Intell. Res.*, 36(1):307–340, 2009.
- [98] Matteo Palmonari, Anisa Rula, Riccardo Porrini, Andrea Maurino, Blerina Spahiu, and Vincenzo Ferme. ABSTAT: Linked Data Summaries with ABstraction and STATistics. In *ESWC Posters & Demo*, 2015.
- [99] Matteo Palmonari, Giuseppe Vizzari, Andrea Broglio, Nicola Lamberti, and Riccardo Porrini. COMMA: A result-oriented composite autocompletion method for e-marketplaces. In *WI-IAT*, pages 545–552, 2012.
- [100] Marius Pasca and Enrique Alfonseca. Web-derived resources for web information retrieval: from conceptual hierarchies to attribute hierarchies. In *SIGIR*, pages 596–603, 2009.
- [101] Heiko Paulheim and Christian Bizer. Type inference on noisy RDF data. In *ISWC*, pages 510–525, 2013.



- [102] Adam Perer and Ben Shneiderman. Integrating statistics and visualization: case studies of gaining clarity during exploratory data analysis. In *SIGCHI*, pages 265–274, 2008.
- [103] José R. Pérez-Agüera, Javier Arroyo, Jane Greenberg, Joaquin Perez Iglesias, and Victor Fresno. Using bm25F for semantic search. In *SEMSEARCH*, pages 1–8, 2010.
- [104] Nathalie Pernelle, Fatiha Saïs, Brigitte Safar, Maria Koutraki, and Tushar Ghosh. N2R-part: Identity link discovery using partially aligned ontologies. In *WOD*, pages 1–4, 2013.
- [105] Emmanuel Pietriga, Christian Bizer, David Karger, and Ryan Lee. Fresnel: A browser-independent presentation vocabulary for rdf. In *ISWC*, pages 158–171, 2006.
- [106] Jay M. Ponte and W. Bruce Croft. A language modeling approach to information retrieval. In *SIGIR*, pages 275–281, 1998.
- [107] Riccardo Porrini, Matteo Palmonari, and Carlo Batini. Extracting facets from lost fine-grained categorizations in dataspace. In *CAiSE*, pages 580–594, 2014.
- [108] Riccardo Porrini, Matteo Palmonari, and Giuseppe Vizzari. Composite match autocompletion (COMMA): A semantic result-oriented autocompletion technique for e-marketplaces. *Web Intelligence and Agent Systems*, 12:35–49, 2014.
- [109] Jeffrey Pound, Peter Mika, and Hugo Zaragoza. Ad-hoc object retrieval in the web of data. In *WWW*, pages 771–780, 2010.
- [110] Jeffrey Pound, Stelios Pappas, and Panayiotis Tsaparas. Facet discovery for structured web search: a query-log mining approach. In *SIGMOD*, pages 169–180, 2011.
- [111] Valentina Presutti, Lora Aroyo, Alessandro Adamou, Balthasar A. C. Schopman, Aldo Gangemi, and Guus Schreiber. Extracting core knowledge from linked data. In *COLD*, pages 37–48, 2011.
- [112] Gianluca Quercini and Chantal Reynaud. Entity discovery and annotation in tables. In *EDBT*, pages 693–704, 2013.
- [113] S. Krishnamurthy Ramnandan, Amol Mittal, Craig A. Knoblock, and Pedro Szekely. Assigning semantic labels to data sources. In *ESWC*, pages 403–417, 2015.

## BIBLIOGRAPHY

---

- [114] Stephen E. Robertson, Hugo Zaragoza, and Michael J. Taylor. Simple bm25 extension to multiple weighted fields. In *CIKM*, pages 42–49, 2004.
- [115] Thomas. L. Saaty and Mujgan S. Ozdemir. Why the magic number seven plus or minus two. *Math. Comput. Model.*, 38(3-4):233–244, 2003.
- [116] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Inf. Process. Manage.*, 24(5):513–523, 1988.
- [117] Sunita Sarawagi. Information extraction. *Found. Trends Databases*, 1(3):261–377, 2008.
- [118] Hansen Andrew Schwartz and Fernando Gomez. Evaluating semantic metrics on tasks of concept similarity. In *FLAIRS*, pages 324–340, 2011.
- [119] Wei Shen, Jianyong Wang, and Jiawei Han. Entity linking with a knowledge base: Issues, techniques, and solutions. *IEEE Trans. Knowl. Data Eng.*, 27(2):443–460, 2015.
- [120] Wei Shen, Jianyong Wang, Ping Luo, and Min Wang. Liege: : link entities in web lists with knowledge base. In *KDD*, pages 1424–1432, 2012.
- [121] Feng Shi, Juanzi Li, Jie Tang, Guotong Xie, and Hanyu Li. Actively learning ontology matching via user interaction. In *ISWC*, pages 585–600, 2009.
- [122] Pavel Shvaiko and Jérôme Euzenat. Ontology matching: State of the art and future challenges. *IEEE Trans. Knowl. Data Eng.*, 25(1):158–176, 2013.
- [123] Pavel Shvaiko, Jérôme Euzenat, Ernesto Jiménez-Ruiz, Michelle Cheatham, and Otkie Hassanzadeh, editors. *Proceedings of the 10th International Workshop on Ontology Matching*, volume 1545 of *CEUR Workshop Proceedings*, 2016.
- [124] Renee E. Sieber and Peter A. Johnson. Civic open data at a crossroads: Dominant models and current challenges. *Government Information Quarterly*, 32(3):308–315, 2015.
- [125] Reetta Sinkkilä, Eetu Mäkelä, Tomi Kauppinen, and Eero Hyvönen. Combining context navigation with semantic autocompletion to solve problems in concept selection. In *SeMMA*, pages 61–68, 2008.
- [126] Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. Semantic taxonomy induction from heterogenous evidence. In *ACL*, pages 801–808, 2006.

- [127] Blerina Spahiu, Riccardo Porrini, Matteo Palmonari, Anisa Rula, and Andrea Maurino. Abstat: Ontology-driven linked data summaries with pattern minimization. In *SumPre*, 2016. to appear.
- [128] Dimitrios Emmanouel Spanos, Periklis Stavrou, and Nikolas Mitrou. Bringing relational databases into the semantic web: A survey. *Sem. Web J.*, 3(2):169–209, 2012.
- [129] Steffen Staab and Rudi Studer. *Handbook on ontologies*. Springer, 2010.
- [130] Emilia Stoica, Marti A. Hearst, and Megan Richardson. Automating creation of hierarchical faceted metadata structures. In *HLT-NAACL*, pages 244–251, 2007.
- [131] Fabian M. Suchanek, Serge Abiteboul, and Pierre Senellart. Paris: Probabilistic alignment of relations, instances, and schema. *PVLDB*, 5(3):157–168, 2011.
- [132] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: A large ontology from wikipedia and wordnet. *J. Web Sem.*, 6(3):203–217, 2008.
- [133] Fabian M. Suchanek and Gerhard Weikum. Knowledge harvesting in the big-data era. In *SIGMOD*, pages 933–938, 2013.
- [134] Fabian M. Suchanek and Gerhard Weikum. Knowledge bases in the age of big data analytics. *PVLDB*, 7(13):1713–1714, 2014.
- [135] Pedro A. Szekely, Craig A. Knoblock, Jason Slepicka, Andrew Philpot, Amandeep Singh, Chengye Yin, Dipsy Kapoor, Prem Natarajan, Daniel Marcu, Kevin Knight, David Stallard, Subessware S. Karunamoorthy, Rajagopal Bojanapalli, Steven Minton, Brian Amanatullah, Todd Hughes, Mike Tamayo, David Flynt, Rachel Artiss, Shih-Fu Chang, Tao Chen, Gerald Hiebel, and Lidia Ferreira. Building and using a knowledge graph to combat human trafficking. In *ISWC (2)*, pages 205–221, 2015.
- [136] Mohsen Taheriyani, Craig A. Knoblock, Pedro A. Szekely, and José Luis Ambite. Rapidly integrating services into the linked data cloud. In *ISWC (1)*, pages 559–574, 2012.
- [137] Mohsen Taheriyani, Craig A. Knoblock, Pedro A. Szekely, and José Luis Ambite. A graph-based approach to learn semantic descriptions of data sources. In *ISWC (1)*, pages 607–623, 2013.
- [138] Arlene G. Taylor. *Wynar's introduction to cataloging and classification*. Libraries Unlimited, 2004.

## BIBLIOGRAPHY

---

- [139] Tran Thanh and Peter Mika. Semantic search - Systems, concepts, methods and the communities behind it. Technical report, Karlsruhe Institute of Technology, Germany and Yahoo! Research, Barcelona, Spain, 2013.
- [140] Thanh Tran, Peter Mika, Haofen Wang, and Marko Grobelnik. Semsearch'11: the 4th semantic search workshop. In *WWW (Companion Volume)*, pages 315–316, 2011.
- [141] Georgia Troullinou, Haridimos Kondylakis, Evangelia Daskalaki, and Dimitris Plexousakis. RDF Digest: Efficient Summarization of RDF/S KBs. In *ESWC*, pages 119–134, 2015.
- [142] Christina Unger, Corina Forascu, Vanessa Lopez, Axel-Cyrille Ngonga Ngomo, Elena Cabrio, Philipp Cimiano, and Sebastian Walter. Question answering over linked data (QALD-4). In *CLEF*, pages 1172–1180, 2014.
- [143] Petros Venetis, Alon Y. Halevy, Jayant Madhavan, Marius Pasca, Warren Shen, Fei Wu, Gengxin Miao, and Chung Wu. Recovering semantics of tables on the web. *PVLDB*, 4(9):528–538, 2011.
- [144] Raphael Volz, Siegfried Handschuh, Steffen Staab, Ljiljana Stojanovic, and Nenad Stojanovic. Unveiling the hidden bride: deep annotation for mapping and migrating legacy data to the semantic web. *J. Web Sem.*, 1(2):187–206, 2004.
- [145] Jingjing Wang, Haixun Wang, Zhongyuan Wang, and Kenny Qili Zhu. Understanding tables on the web. In *ER*, pages 141–155, 2012.
- [146] Bifan Wei, Jun Liu, Jian Ma, Qinghua Zheng, Wei Zhang, and Boqin Feng. DFT-extractor: a system to extract domain-specific faceted taxonomies from wikipedia. In *WWW (Companion Volume)*, pages 277–280, 2013.
- [147] Bifan Wei, Jun Liu, Qinghua Zheng, Wei Zhang, Xiaoyu Fu, and Boqin Feng. A survey of faceted search. *J. Web Eng.*, 12(1&2):41–64, 2013.
- [148] Fei Wu and Daniel S. Weld. Automatically refining the wikipedia infobox ontology. In *WWW*, pages 635–644, 2008.
- [149] Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Qili Zhu. Probbase: a probabilistic taxonomy for text understanding. In *SIGMOD*, pages 481–492, 2012.
- [150] Zhibiao Wu and Martha Stone Palmer. Verb semantics and lexical selection. In *ACL*, pages 133–138, 1994.

- [151] Ning Yan, Chengkai Li, Senjuti Basu Roy, Rakesh Ramegowda, and Gautam Das. Facetedpedia: enabling query-dependent faceted search for wikipedia. In *CIKM*, pages 1927–1928, 2010.
- [152] Xiang Zhang, Gong Cheng, and Yuzhong Qu. Ontology summarization based on rdf sentence graph. In *WWW*, pages 707–716, 2007.
- [153] Ziqi Zhang. Learning with partial data for semantic table interpretation. In *EKAW*, pages 607–618, 2014.
- [154] Ziqi Zhang. Towards efficient and effective semantic table interpretation. In *ISWC*, pages 487–502, 2014.
- [155] Ziqi Zhang. Effective and efficient semantic table interpretation using TableMiner<sup>+</sup>. *Under transparent review: Sem. Web J.*, 2015.
- [156] Stefan Zwicklbauer, Christoph Einsiedler, Michael Granitzer, and Christin Seifert. Towards disambiguating web tables. In *ISWC Posters & Demo*, pages 205–208, 2013.