Data-driven Modelization and Knowledge Graph Generation within the Tourism Domain

Alessandro Chessa [a], Gianni Fenu [b], Enrico Motta [c], Francesco Osborne [c,d], Diego Reforgiato Recupero [b], Angelo Salatino [c] and Luca Secchi [a,b,*]

[a] *Linkalab, Viale Elmas, 142, 09122, Cagliari, Italy*
E-mail: alessandro.chessa@linkalab.it
[b] *Department of Mathematics and Computer Science, University of Cagliari, Cagliari, Italy*
E-mails: fenu@unica.it, diego.reforgiato@unica.it, luca.secchi@unica.it
[c] *Knowledge Media Institute, The Open University, Milton Keynes, United Kingdom*
E-mails: enrico.motta@open.ac.uk, francesco.osborne@open.ac.uk, angelo.salatino@open.ac.uk
[d] *University of Milano Bicocca, Milan, Italy*

Abstract. The tourism and hospitality sectors have become increasingly important in the last few years and the companies operating in this field are constantly challenged in providing new innovative services. At the same time (big) data has become the "new oil" of this century and Knowledge Graphs are emerging as the most natural way to collect, refine, and structure this heterogeneous information. In this paper, we present a methodology for semi-automatic generating a Tourism Knowledge Graph (TKG), which can be used for supporting a variety of intelligent services in this space, and a novel ontology for modelling this domain, the Tourism Analytics Ontology (TAO). Our approach processes and integrates data from Booking.com, AirBnB, DBpedia and GeoNames. Due to its modular structure, it can be easily extended to include new data sources or to apply new enrichment and refinement functions. We report a comprehensive evaluation of the functional, logical, and structural dimensions of TKG and TAO.

Keywords: Knowledge Graphs, Ontology Design, Tourism Ontology, Web Science, Web Mining, Tourism, Hospitality

1. Introduction

We are currently living in the age of big data, and the sheer volume of new data being generated is making the World Wide Web shifting from a web of content to a web of data. This gives all practitioners the opportunity to build more innovative and functional web services.

Semantic Web and Linked Data technologies aim to representing the web itself through a large global graph that can be queried using standard protocols and languages [31]. The World Wide Web Consortium (W3C) has developed and promoted different standards, like RDF/S, OWL and SPARQL, that are now widely adopted to create knowledge bases which represent data as knowledge graphs (KGs). A knowledge graph is a graph of data whose nodes represent entities of interest and whose edges represent relations between these entities [18]. A few examples of knowledge graphs publicly available are DBpedia [31], YAGO (Yet Another Great Ontology) [40] or WikiData [17]. Knowledge graphs can store data and metadata using a common structure and are often used in application scenarios that involve extracting and integrating information from multiple, and possibly heterogeneous, sources. Typically the data in the

*Corresponding author. E-mail: luca.secchi@unica.it.

knowledge graph are modelled according to a domain ontology, which gives meaning to the represented text and supports inferring new knowledge.

The field of tourism is a natural domain of application of these technologies since stakeholders in this space need to integrate data from several heterogeneous sources in order to generate a multifaceted characterisation of tourist destinations and all relevant actors [5, 21, 42].

A tourist destination can be thought of as the place or area which is central in the decision of a tourist to take the trip[1] and is usually characterised according to two aspects: supply and demand. The supply side is based on the willingness and ability of producers to create goods and services to take them to market. Understanding the supply side of tourism includes all aspects related to tourism offerings and attractions (e.g., accommodations, events, points of interest, restaurants, and so forth). On the other hand, demand refers to how much (quantity) of a product or service is desired by buyers. Understanding which factors influence the demand side of tourism includes all aspects related to tourists' choices and opinions or their characteristics (e.g., socio-demographic, classification, provenance).

This information is crucial for informing business and marketing decisions as well as supporting a variety of software and services in this space, such as search engines and recommendation systems [29, 41].

The creation of KGs in this domain is a time-consuming and costly process, even with the help of mapping languages such as RML [1, 12, 46]. Indeed, it is still a challenge to automatically generate KGs from multiple semi-structured and textual sources (e.g., descriptions of specific accommodations, reviews) in order to describe the many facets of this domain, such as the different kinds of accommodations and amenities. Therefore, many KGs in this space are no longer maintained [5, 21] or cannot be easily extended to other tourist destinations [1]. In addition, the relevant ontologies, such as STI Accommodation Ontology[2], Schema.org[3], and Hontology [7] are to some degree incompatible with each other (as discussed in section 3.3.2) and do not offer a fine-grained representation of some crucial entities (e.g., amenities).

In this paper, we illustrate a general, reproducible, and easily extendable methodology for KG generation and the resulting framework for semi-automatically creating a *Tourism Knowledge Graph (TKG)*, which integrates information from Booking.com, Airbnb.com, DBpedia, and GeoNames. This advanced characterisation of tourism can be used to enable the quantitative analyses of a tourist destination and support several intelligent services. In order to model this data, we developed the *Tourism Analytics Ontology (TAO)*, which offers a much more comprehensive characterisation of this domain than previous solutions and can be easily reused by similar initiatives.

We showcase our solution by applying it to touristic locations in Sardinia and London, producing over 10M triples describing almost 36K lodging facilities and 898K reviews. The resulting knowledge graph is available online via a SPARQL end-point[4]. The TAO ontology is also available online[5]. Finally, for the sake of reproducibility, we share the code-base for our knowledge graph generation pipeline, for engineering TAO, and the evaluation tests[6].

To summarise, the contributions of this manuscript are the following:

- a general data-driven methodology for the semi-automatically generation of knowledge graph that we applied to the tourism domain;
- an open-source pipeline for generating a tourism knowledge graph from (semi)-structured and unstructured data;
- the novel Tourism Analytics Ontology (TAO);
- an open-source program to produce the Tourism Analytics Ontology (TAO) using code and data;
- an instance of the tourism knowledge graph (TKG) with data relative to two Tourist Destinations (Greater London and Sardinia island in Italy);
- an evaluation assessing functional, logical, and structural dimensions of TAO and TKG.

[1] The World Tourism Organization (UNWTO) defines in its glossary a *destination* as "the place visited that is central to the decision to take the trip". See https://www.unwto.org/glossary-tourism-terms.

[2] http://ontologies.sti-innsbruck.at/acco/ns.html

[3] https://schema.org/docs/hotels.html

[4] http://tourism.sparql.linkalab-cloud.com/sparql access with login: paper password: journal_p4p3r2022!!

[5] See http://purl.org/tao/ns

[6] See https://github.com/linkalab/tkg

The remainder of this paper is organised as follows. Section 2 describes related works about knowledge graphs within the tourism domain. Section 3 explains the methodology adopted to guide the knowledge graph creation, explaining each of the six iterative phases in which we have subdivided the process. Section 4 presents the evaluation, and finally, Section 5 ends the paper with conclusions and future directions of work.

2. Related Work

In the previous years, various attempts have been made to build knowledge bases in several domains, including the tourism, using information extracted from websites and social media.

The platform 3cixty [42] was built during Expo Milano 2015 to create comprehensive knowledge bases that contain descriptions of events and activities, places and sights, transportation facilities, and social activities collected from numerous, near- and real-time local and global data providers, including hyper-local sources. In 2016-2017 new knowledge bases have been created for the cities of London, Madeira, and Singapore, as well as for the entire French Cote d'Azur area. The project now seems no longer maintained although a SPARQL endpoint remains active allowing to export data only in HTML and not as RDF.

The Tourpedia platform that was meant to be the DBpedia of tourism, was developed within the OpeNER Project [21]. OpeNER (Open Polarity Enhanced Name Entity Recognition) was a project funded under the 7th Framework Program of the European Commission whose main objective was to implement a pipeline to process natural language. The project is no longer maintained although anyone can run the proposed pipeline to view categories, places information, and create and manage events and tour plans for users. Also, on the main website, it is still possible to run the web demo application, showing the sentiment about places through an interactive map. Some datasets are still available for download, however other tools, including the SPARQL endpoint, are no longer working.

DBtravel [5] is a tourism-oriented knowledge graph generated from the collaborative travel site Wikitravel that takes advantage of the recommended guidelines for contributors provided by Wikitravel and extracts the named entities available in Wikitravel Spanish entries by using an NLP pipeline. As for the previous two projects, the knowledge graph and the source code used to produce it are no longer maintained nor available online.

Other projects demonstrate that semantic technologies and knowledge graphs can be successfully applied to tourism when information is extracted from curated proprietary data sources. In the case of *La Rioja Turismo* Knowledge Graph, Alonso-Maturana et al. [1] retrieve and integrate information referring to attractions, accommodation, tourism routes, activities, events, restaurants, and wineries from heterogeneous and diverse management systems. This approach is focused on the La Rioja Turismo ecosystem but cannot be easily extended to other tourist destinations.

In the case of the Tyrolean Tourism Knowledge Graph [28], data based on schema.org annotations are collected from destination management organisations (DMOs) and their IT service providers. In this case, the knowledge graph creation is based on the availability of coherent schema.org annotations in the source websites, which was possible thanks to the cooperation of Tyrolean DMOs. Once again, this scenario is not always applicable because it requires a central organisation to coordinate the different stakeholders.

Another proposed approach was to collect, enrich, and publish Linked Open Data for the Municipality of Catania, a city in Southern Italy, in the context of the project PRISMA, "PlatfoRms Interoperable cloud for SMArt-Government"[7] [9–12]. In this case, Consoli and his colleagues presented the collected city data, described the process and issues to create a semantic data model for emergency vehicle routing and geo-linked data, and discussed a developed prototype. In particular, they described the employed procedures, ontology design patterns, and tools used for ensuring semantic interoperability during the transformation process.

Other state-of-the-art solutions include the generation of a knowledge graph of tourism in the Chinese language [45, 46]. The authors constructed such knowledge graphs, by extracting knowledge from the existing encyclopedia knowledge graph and unstructured web pages in the Chinese language.

It is still a big challenge to automatically generate a knowledge graph that integrates the most important data sources in this field and can be easily extended to other touristic locations. We also lack a single ontology that would

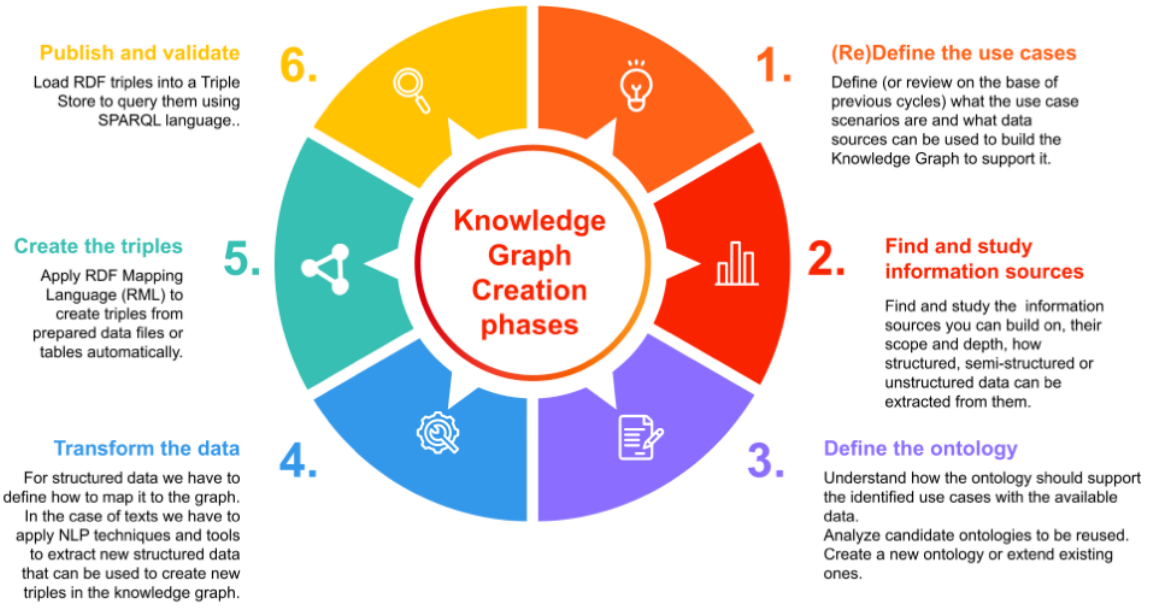
[7] http://www.ponsmartcities-prisma.it/

offer a fine-grained description of touristic accommodations (e.g., Hotel Splendor), locations (e.g., Regent Park), and destinations (e.g., London). The work presented in this paper aims to address this gap by introducing 1) TAO, a very comprehensive ontology of accommodations, locations, and destinations, 2) a general, reproducible, and easily extendable methodology to integrate relevant data sources and generate a knowledge graph, which we applied to the tourism domain.

3. Methodology

Our approach for KG construction is organised into six macro phases that can be iteratively repeated to refine the resulting KG. Specifically, the first three phases are the core of a **data-driven design process** that leverages the knowledge embedded in the data sources for guiding the use case refinement and ontology engineering. The last three phases drive the actual implementation of the knowledge graph and its publishing. Figure 1 displays the different phases.

Fig. 1. Tourism Knowledge Graph creation phases



The *first phase* is focused on the definition of the use cases that the knowledge graph should support, that is to say, what are the desired outcomes a user or an application should be able to produce from it. Because our process is driven by what we can find in the data, it is a preliminary definition that is always subject to further refinements and that should be revised multiple times until all use cases are positively supported by the KG.

The *second phase* is about understanding how the data at our disposal can support the use cases, but it is also about extracting knowledge from the data to support the ontology definition. On the one hand, the data is used to adapt the use cases to the actual information we have access to, thus extending the scope for some use cases or reducing it for others. For example, if we do not find in the data any information about the total number of rooms for a hotel, we cannot support any use case about the available accommodation capacity for a tourist destination unless we find new data sources. On the other hand, the data is analysed to guide the ontology design. As an example, the accommodations offered on AirBnB have specific types, like shared rooms, which are peculiar to a sharing

economy approach. They may also include amenities we seldom find in other forms of hospitality like hotel rooms. This information incorporates knowledge about the hospitality services for tourism through which we can leverage the process of ontology design and engineering together with the building of the knowledge graph itself.

The *third phase* focuses on the creation of an ontology that supports all the use cases defined in the first phase and incorporate the domain knowledge distilled in the second phase.

The *fourth phase* is about transforming the data extracted from the data sources in order to prepare it to be used for triple creation in the following phase. During this process, various data wrangling techniques are applied to semi-structured data, whereas natural language processing is applied to unstructured texts (e.g., language detection, named entity extraction, entity linking).

The *fifth phase* is concerned with triple creation using the data prepared in the previous phase. The triple creation is performed using RDF Mapping Language (RML) in order to include in the knowledge graph also the transformation process metadata.

Finally, the *sixth phase* focuses on the publication of the knowledge graph in a triple store.

In the following sub-sections, we describe each phase in detail.

3.1. Define the use cases

We start with a first general definition of some use cases that we want to cover when building the KG, also considering what data sources could be used to support them. We should also define which kind of applications we would need to implement on top of the KG to support the use cases. This analysis can give us a more general scenario on how the KG would be used. This, in turn, is useful to understand to what extent the data sources can support the scenario and guide the design process on how the KG should be structured. In fact this phase is intertwined with the second phase (i.e., *Find and study information sources*), discussed in Section 3.2, because we need to consider the information we can extract from the web to support the selected use cases. It is also related to the third phase (i.e., *Define the ontology*) in Section 3.3, because we can have different design approaches regarding the KG depending on what kind of methods and applications it should support (e.g., whether or not we want to apply reasoning techniques on the KG).

In order to generate a KG that can be used to support the analysis of tourist destinations with respect to the supply and demand side, we have identified the following use cases:

1. Support the identification of the topics of interest discussed by tourists in their reviews;
2. Support the identification of the topics of interest presented in the descriptions of lodging facilities[8] and accommodation[9] offers;
3. Support the recognition and linking of tourism entities in the KG for different applications revolving in the domain of social media, news and blogs;
4. Support sentiment analysis [2, 16] applications about tourists toward lodging businesses and destinations;
5. Support the classification of tourist destinations on the base of what they offer and on the base of tourist opinions.

We also identified a number of applications that can leverage the KG to produce better results (see [32] for a comprehensive overview of applications based on knowledge graphs). In turn, each one of the following applications can be used to better support one or more use cases:

1. **automatic reasoning**[10] and **graph learning**[11] on the KG allows for the entailment of new triples thus enriching the explicit knowledge other applications can work on; for this reason, it is indirectly related to all use cases;

[8] Lodging facilities mean any hotel, motel, motor inn, lodge, and inn or other quarters that provide temporary sleeping facilities open to the public. See https://www.lawinsider.com/dictionary/lodging-facilities)

[9] An accommodation is a place that can accommodate human beings, e.g., a hotel room, a camping pitch, or a meeting room. An accommodation is always part of a lodging facility (e.g., a hotel room is part of a Hotel.)

[10] Leveraging Description Logic and OWL.

[11] Using Graph Neural Networks or similar techniques.

2. **named entity recognition (NER) and entity linking (EL)** of tourist locations and lodging businesses using the KG have an immediate positive impact on use cases 3 and 5.
3. **relation extraction (RE) in a closed setting** for the tourism industry can be used to support a better understanding about the relations between users and touristic entities thus improving use cases 4 and 5.
4. **tourism-related Topic Modelling** (cluster words/phrases frequently co-occurring together in the tourism context) for texts and documents written in natural language can be used to support use cases 1 and 2.
5. **tourism related Topic Labelling** (for clusters of words identified as abstract topics, extract a single term or phrase that best characterises the topic) can also be used to support use cases 1 and 2.
6. **Text Classification** of documents concerning tourism topics can support use cases 1, 2, and 5.
7. **Semantic Annotation** of documents about tourism with entities, classes, and topics based on the KG can be used to support all the use cases by improving user interfaces and user interactions with the textual data.

It is important to note that, the actual feasibility of a use case can be confirmed only when the knowledge graph is built and one or more of the supporting applications are implemented. This validation phase is out of scope for the present work, which focuses on the design and construction of the knowledge graph.

3.2. Find and study information sources

To support the use cases described in Section 3.1 we need to identify a minimum set of information sources we need throughout the construction of a *core* version of the Tourist Knowledge Graph. After this core Knowledge Graph is created, new information sources could be added by applying the same process described in this work. This is because knowledge graphs have a flexible schema which makes them easily extendable.

Observing the use cases, we can see that we need information sources about:

- lodging facilities and the accommodation they offer;
- user reviews and opinions;
- tourist locations (i.e., points of interest for a tourist such as a train station or a beach);
- tourist destinations such as London or the Costa Smeralda (i.e., the place visited that is central to the decision to take the trip);

The first set of information sources adequately covering the listed items consists of:

- Booking.com, a digital travel company specialised in hotels, B&Bs, and other types of hospitality; from its website we can collect information about accommodations and related offers but also users' opinions expressed as reviews.
- AirBnB, an American company that connects hosts, offering their accommodation spaces (e.g., apartments, rooms, etc.), and travellers, looking for a place to stay; it adopts a peer-to-peer model that originates from sharing economy and represents a new emerging reality in the tourism and accommodation market; its website is a source of information about accommodations and related offers but also users' opinions expressed as reviews.
- DBpedia[12], an open knowledge graph built with structured content extracted from the information created in various Wikimedia projects (e.g., Wikipedia). Specifically, we link entities in TKG to the DBpedia entities of selected classes (e.g., `DBpedia:Places` or `DBpedia:Food`).
- GeoNames[13], a geographical database exposed through APIs and as RDFs documents. We connect entities in TKG with GeoNames entities representing places.

It is worth noticing that, although there are many other websites and applications for tourism and hospitality, Booking.com and AirBnB are market leaders and together cover both the traditional accommodation industry and the emerging sharing economy. A similar consideration could be made for DBpedia and GeoNames when we consider places (DBpedia and GeoNames) or general topics related to tourism (DBpedia).

[12] https://www.dbpedia.org/
[13] http://www.geonames.org/

For the present work, we build upon the results of an industrial project about Tourism 4.0 called *Data Lake Turismo* developed by Linkalab s.r.l.[14], which was the evolution of a previous research project promoted by the Digital Innovation Hub of Sardinia[15] and Fondazione Banco di Sardegna[16]. The project aimed at creating a digital platform for tourism data analysis. One of the main components of this platform was a data lake for collecting, transforming, and analysing data in this sector. However, the project lacked a semantic layer that could support and enhance the data analysis, which is the starting point and motivation of the present work.

Through this infrastructure, we have access to data assets related to lodging businesses, user reviews, and opinions; and we enrich them with DBPedia and Geonames.

The data source selection influences both the use case and the ontology definition phases. Although it could be possible to add new data sources to the mix from the beginning, it has a cost and should be postponed wherever possible, because our objective is to complete the construction of a core version of the knowledge graph before expanding its coverage. On the other hand, we should always select data sources that incorporate a rich and well-established model of the business sector (tourism in our case) in the data itself. This is important to support the ontology design with a data-driven analysis process.

Source data exploration

The first step of this phase is to understand what kind of data we can use. We should examine the documentation but we also need to perform an exploratory data analysis on the files and tables accessible in the source data lake in order to have a complete grasp of its contents. This analysis is focused on the following resources available in the data lake:

- data about hospitality:
  - information related to lodging facilities (e.g., hotels, b&bs, resorts) and their characteristics (e.g., name, address, type, hospitality features);
  - information related to accommodations offered by a lodging business (e.g., hotel room, b&b room, apartment).
  - rent offers for accommodation (e.g., price, number of people, etc.).
- data about user reviews (e.g., user, date, rating, text).

Data is extracted from the data lake in tables with nested structures and needs to be "flattened" to be used by the downstream tasks. This is due to the way the data lake stores information in a redundant and not normalised way.

The result of the exploratory analysis has shown:

- how data is organised in fields and sub structures;
- that structured and unstructured data (i.e., texts) is available;
- that texts can be in many different languages and it is not always specified in which one;
- that structured data fields can contain numbers, Boolean values, time/date values, or categorical values;
- that data is not always typed and can be represented internally as strings;
- that categorical data is not related to a lookup table or taxonomy;
- that in some cases there are no unique IDs that can be used to identify a resource.

This analysis led us to define some fundamental data pre-processing steps to be executed before building the Knowledge Graph and the Ontology:

- **data preparation**: in this step, we extracted the data from the source data lake via SQL queries; next, we stored it on a local file system to be prepared (cleaned, flattened, combined) so that it can be used for downstream tasks.

[14] Linkalab s.r.l. is an Italian small enterprise specialised in data science and data engineering. Home page https://www.linkalab.it/

[15] https://www.dihsardegna.eu/

[16] https://www.fondazionedisardegna.it/

– **data enrichment**: in this step, we augmented the data using various techniques; specifically, we applied NLP techniques to identify the language of the text (e.g., English, Italian, French, and so on), because downstream tasks depend on it to work properly.

We also found that the data lake source should be integrated with data about attractions and points of interest from other sources. To support this need we identified DBpedia and GeoNames as the most appropriate data sources for the following reasons: i) both sources are stable and constantly maintained, with a vast supporting community; ii) both sources cover the identified destinations (and many others) in depth; iii) both sources are exposed as linked open data and APIs.

3.3. Creation of the domain ontology

In order to generate a KG able to support the identified use cases and the related applications, we need an ontology that can satisfy all the relevant functional and non-functional requirements. We thus set up several requirements in collaboration with domain experts from Linkalab.

Concerning the functional requirements (FR), we envisaged that the ontology would need to:

FR 1 model lodging facilities and define a taxonomy[17] of their types (e.g., hotels, hostels, apartments),
FR 2 model accommodations and define a taxonomy of their types (e.g., room, entire apartment, suite);
FR 3 model amenities offered to tourists and define a taxonomy of their types (e.g., disable access, parking garage, baby monitor);
FR 4 model tourist locations (e.g., waterfall, beach, museum, park) and define a taxonomy of their types;
FR 5 model the relations among entities (e.g., geographic relations, mentions, composition/inclusion);
FR 6 model tourist reviews;
FR 7 model tourist destinations (e.g., Sardinia, London), which is the place that is central to the trip.

Concerning non-functional requirements (NFR) the ontology should support reasoning and be based on widely adopted technical and market standards. In particular:

NFR 1 should be defined in OWL[18];
NFR 2 should be based on two *de-facto* standards to model business data:

- Schema.org[19], which is a set of vocabularies developed through a collaborative effort for structuring data on the web. It was originally founded by Google, Microsoft, Yahoo, and Yandex.
- GoodRelations, which is a lightweight ontology for exchanging e-commerce information, namely data about products, offers, points of sale, prices, terms, and conditions, on the Web.

NFR 3 should be easy to extend in order to cover other use cases in the tourism domain.

We analysed several ontologies covering the tourism domain (detailed in Section 3.3.2) but none of them satisfies all these requirements. Therefore, we designed and implemented a new ontology: the Tourism Analytics Ontology (TAO).

We devote the following sections to describing: i) the competency questions that guided the design of TAO; ii) the ontologies which we used as a starting point; and iii) the final version of TAO and our design choices.

[17] From now on we refer to taxonomy as a hierarchy of classes connected with rdfs:subClassOf property.
[18] More specifically it should be based on OWL DL dialect which is designed to provide the maximum expressiveness possible while retaining computational completeness, decidability, and the availability of practical reasoning algorithms.
[19] See https://schema.org/

3.3.1. Competency questions

In order to design the TAO ontology, we first defined a set of competency questions, i.e., queries expressed in natural language [22, 35]. Competency Questions (CQ) are useful to express the functional requirements formulated above since they i) can be easily understood by non-technical people; ii) can guide the ontology engineering process working as a practical reference of what should be implemented, iii) can be easily tested during the validation process. We report in Section 4.1 the tests we performed using CQ to validate the ontology with respect to its functional requirements.

We adopted a data-driven design process and followed two complementary approaches when defining the competency questions: i) top-down, by developing new questions with a domain expert and then checking whether they could be answered with our data; and ii) bottom-up, by deriving them from the information available in the source data. Here, we report a list of the most relevant information available in the data sources (discussed in Section 3.2) that drove the CQs formulation:

1. information about lodging facilities:
   (a) name(s)
   (b) position
   (c) geographic relations with administrative divisions
   (d) geographic relations with tourist destinations
   (e) type (e.g., Hotel, Resort, Motel, B&B, Holiday Accommodations)
   (f) type of accommodation offered (e.g., room, apartment, villa, bungalow, etc.)
   (g) amenities (e.g., sauna, parking, swimming pool, breakfast, air conditioning, etc.)
   (h) accommodation prices exposed on the web
   (i) user ratings
   (j) textual descriptions (to perform Named Entity Recognition, Entity Linking and Relation Extraction, etc.)
2. information about tourist locations:
   (a) name (in multiple languages)
   (b) position
   (c) geographic relations with administrative divisions
   (d) geographic relations with tourist destinations
3. information about tourist destinations:
   (a) name (in multiple languages)
   (b) position
   (c) geographic relations with administrative divisions
   (d) geographic relations with tourist locations
4. tourist reviews about lodging businesses and locations
   (a) user votes
   (b) tourist nationality and type of tourist (family, couple, etc.)
   (c) textual review (to perform Named Entity Recognition, Entity Linking and Relation Extraction, etc.)

This list will also drive the process of ontology engineering since it defines the kind of entities and properties that should be modelled by the TAO ontology.

We defined the following 12 competency questions:

CQ 1 Which are the first 10 hotels with more than 1,000 reviews and the lowest mean value of users' review scores? (derived from the functional requirements FR1 and FR6)

CQ 2 Find three apartments with Wi-Fi, distant at most 2Km from at least two Parks. (FR1, FR2, FR3, FR4, FR5)

CQ 3 Which Tourist Destinations have the highest percentage of high-priced Lodging Facilities (at least one offer for an accommodation for two persons with a nightly price two times over the mean price)? (FR1, FR2, FR5, FR7)

CQ 4 What are the 10 tourist locations cited most by hotel descriptions that also offer a day Spa in a specific tourist destination? (FR3, FR4, FR5, FR6, FR7)
CQ 5 What are the most cited Tourist Locations in all Lodging Facility descriptions within a certain tourist destination? (FR1, FR4, FR5, FR7)
CQ 6 What are the Tourist Locations cited most in positive user reviews? (FR4, FR5, FR6)
CQ 7 What are the 10 cheapest apartments that offer at least two beds and secured parking and are within 10km from an airport? (FR2, FR3, FR4, FR5)
CQ 8 Which type of Lodging Facility is more reviewed by tourists in a specific Tourist Destination? (FR1, FR5, FR6, FR7)
CQ 9 What are the top Tourist Destinations with respect to positive sentiment about food (i.e., percentage of Lodging Facilities with positive reviews that cite food)? (FR1, FR5, FR6, FR7)
CQ 10 In which months do we have the highest number of user reviews for Hotels? (FR1, FR6)
CQ 11 What Tourist Locations can be found in a Tourist Destination? (FR4, FR5, F6)
CQ 12 How many beds are offered on lease in a certain Tourist Destination? (FR2, FR5, FR7)

3.3.2. Reuse of existing ontologies

We analysed several tourism ontologies to assess if they could be reused to support our use cases. We identified three main families of ontologies:

1. ontologies based on Open-Travel or other heavyweight industrial standards, typically focused on information exchange among tourism organisations (e.g., the Harmonise Ontology [19]).
2. ontologies produced by researchers to support specific tasks, such as question answering (e.g., QALL-ME Ontology [37]) and information retrieval (e.g., GETESS [39]) as well as ontologies that combine or build on them (e.g., cDOTT [3], Hontology [7]).
3. ontologies based on Schema.org [23] and GoodRelations [25], such as the STI Accommodation Ontology.

Based on the functional and non-functional requirements, we then selected three of them: (i) STI Accommodation Ontology, (ii) the Schema.org markup for hotels, and (iii) Hontology. The latter is currently not available as OWL serialisation at any specific URI and does not seem to be maintained anymore. TAO also reuse other two ontologies: (iv) GeoNames[20], which is used to specify the geographic locations, and (v) the DBpedia ontology[21], which is used for further characterising locations and food types (e.g., pizza, sushi).

In what follows, we will describe the selected ontologies and vocabularies and how they have been reused in TAO.

Accommodation Ontology (prefix `acco:`) is an extension of GoodRelations (prexif `gr:`) focused on describing accommodation offers from an e-commerce perspective. It provides the additional vocabulary elements for describing hotel rooms, hotels, camping sites, and other forms of accommodations as well as their features. However, it does not make a distinction between the lodging facility (e.g., a hotel as a whole), and the individual accommodations on a lease (e.g., the hotel rooms), because all lodging facility types and accommodation types are sub-classes of the same class (`acco:Accommodation`).

The Accommodation Ontology does not define a taxonomy of amenities (called accommodation features) but "provides a consolidated conceptual model for encoding proprietary feature information". So instead of defining classes for room and hotel features, the ontology provides the generic class `acco:AccommodationFeature` that can hold feature information in varying degrees of formality. A leasing offer is modelled using the GoodRelations relation `gr:Offering` specifying that the offering is a `gr:LeaseOut` using the property `gr:hasBusinessFunction`. Unfortunately, the Accomodation ontology does not cover several concepts that are required for our use case, including 1) tourist destinations (e.g., London), 2) tourist locations (e.g., beach, church, subway station), 3) tourist reviews.

We reused a few classes and properties from the Accomodation and GoodRelations ontologies, including `acco:AccommodationFeature`, `acco:BedDetails`, `acco:value`, `acco:bed`, `gr:Offering`, `gr:TypeAndQuantityNode`, `gr:hasPriceSpecification`, `gr:name`

[20] https://www.geonames.org/ontology/documentation.html
[21] https://www.dbpedia.org/resources/ontology/

Schema.org markup for hotels (prefix `schema:`), incorporates and extends many Accommodation Ontology [27] concepts. Schema.org models hospitality according to three main classes[22]:

1. A **lodging business**, (e.g., a hotel, hostel, resort, or a camping site): essentially it represents both the lodging facility, which is the place that houses the actual units of the establishment (e.g., hotel rooms) and the business organisation governing it. The lodging business can encompass multiple buildings but is in most cases a coherent place.
2. An **accommodation**, i.e., the relevant units of the establishment (e.g., hotel rooms, suites, apartments, meeting rooms, camping pitches, etc.). These are the actual objects that are offered for rental.
3. An **offer** to let a hotel room, or other forms of accommodations, for a particular price and a given type of usage (e.g., occupancy), typically further constrained by booking requirements and other terms and conditions.

In this case, we have a clear distinction between lodging business and accommodation because we have two distinct classes: `schema:Accommodation` and `schema:LodgingBusiness`. Unfortunately, Schema.org is not intended to be used as an OWL ontology because its data model is very generic and derived from RDF Schema[23]. The main purpose of Schema.org is to enable sharing of structured data on the Internet whereas OWL is based on formal semantics that enables reasoning on the knowledge graph. In addition, the `schema:LodgingBusiness` class cannot be used in conjunction with GoodRelations ontology without introducing logical contradictions. Specifically, Schema.org defines `schema:LodgingBusiness` as a subclass of `schema:LocalBusiness` which is a subclass of both `schema:Organisation` and `schema:Place`. On the other hand, GoodRelations states that `schema:Organization` and `schema:Place` are disjoint.

We reused Schema.org in TAO by importing and extending a few classes and properties, including `schema:PostalAddress`, `schema:UserReview`,`schema:address`, `schema:subjectOf`. We also selected appropriate schema.org types that describe places to enrich the tourism location taxonomy using `rdfs:seeAlso` to establish a mapping with them[24].

Hontology (prefix `ho:`) is a multilingual ontology for the accommodation sector (H stands for hotel, hostal, and hostel). It is a freely available domain-specific ontology in four languages: English, Portuguese, Spanish and French [7, 8]. It was partially aligned with QALL-ME and Schema.org and described several useful concepts in this domain such as Facilities (a.k.a. amenities), Services, Staff, and Points Of Interest. The ontology is not published as linked data but can be downloaded and used in a local environment. Its latest version dates back to 2012 and therefore it is not aligned with the most recent extensions of Schema.org. In addition, since it is not based on GoodRelations, it does not fulfill our non-functional requirements.

However, we were able to re-implement with TAO some of its classes describing location amenities, including `ho:Balance`, `ho:AirConditioning`, `ho:Ballroom`, `ho:BeautySalon`.

DBpedia Ontology[25] (prefix `dbpedia:`) is a shallow, cross-domain ontology, which has been manually created based on the most commonly used infoboxes within Wikipedia[26]. The ontology currently covers 685 classes which form a subsumption hierarchy and are described by 2,795 different properties. We used some of the classes from this ontology to model a taxonomy of location types (subclasses of `tao:TouristLocation`) also mapped to GeoNames geographic features.

GeoNames Ontology[27] (prefix `gn:`) provides elements of description for geographical features, in particular those defined in the geonames.org database. It has three key ontology classes: Feature (a set of all geospatial instances in GeoNames like cities and countries), Class (a set of all feature schemes defined in GeoNames), and Code (a set of abbreviation feature codes in different feature schemes). GeoNames Feature is used for describing

[22] Usually called types in schema.org.

[23] See https://schema.org/docs/datamodel.html

[24] In this respect we can consider TAO ontology an external extension of Schema.org as described in the page https://schema.org/docs/extension.html

[25] https://www.dbpedia.org/resources/ontology/

[26] As defined in the DBpedia ontology page http://web.archive.org/web/20210416134559/http://wikidata.dbpedia.org/services-resources/ontology

[27] https://www.geonames.org/ontology/documentation.html

concrete geospatial entities (UK, Washington, Colosseum, etc.), whereas GeoNames Class and Code are used for representing meta-information about features. All feature instances are uniquely identified by URI in GeoNames.

We used GeoNames `gn:Feature` class to model classes that are also places (e.g., lodging facilities, tourist locations) and to express their geographic relations using `gn:parentFeature`. We also used GeoNames to enrich the taxonomy of tourist location types with specific codes, for example, `tao:Park` was associated to the `gn:L.PRK` code.

3.3.3. The Tourism Analytics Ontology

In this section, we describe the new Tourism Analytics Ontology (TAO) and discuss our design choices. We aimed at developing an ontology that i) would be compatible with all the requirements listed in Section 3.3.1, ii) would be able to integrate all relevant information from the data sources, and iii) would be fully compatible with the Accommodation Ontology, GoodRelations, and Schema.org. Specifically, the Accommodation Ontology is explicitly imported using `owl:imports`, GoodRelations is imported indirectly through Accommodation Ontology and Schema.org is partially included by reusing specific classes and properties or making explicit mappings to it.

The new ontology has the following characteristics:

1. introduces the LodgingFacility class which represents any hotel, motel, inn, or other quarters that provide temporary sleeping facilities open to the public[28];
2. distinguishes between lodging facilities and specific accommodations within lodging facilities;
3. includes an extended taxonomy[29] of lodging facilities types (e.g., hotel, house, resort) ;
4. includes an extended taxonomy of the amenities (e.g., oven, parking garage, baby monitor) offered by lodging businesses;
5. includes an extended taxonomy of geographic features relevant to tourism (based on schema.org) and enriched with GeoNames feature taxonomy (leveraging the GeoNames mapping[30] data-set);
6. uses schema.org to model Tourist Destinations and Tourist Locations;
7. can be easily extended to model other kinds of entities relevant for tourism in the future (e.g., events or restaurants).

Figure 2 illustrates the schema of the TAO ontology. We will refer to TAO using the `tao:` prefix from now onward. The central classes are `tao:LodgingFacility` and `tao:Accommodation` that are respectively used to model lodging facilities and their accommodations. The `tao:LodgingFacility` class is related to the lodging business concept used in Schema.org, but only refers to the physical place where the accommodations within the facility are located (e.g., a hotel is considered as the building that contains rooms). In this way, there is a clear distinction with the business organisation that governs or owns the lodging facility and no inconsistencies are generated by GoodRelations disjunction between `schema:Place` and `schema:Organization` classes, as discussed in Section 3.3.2. A facility location is described according to its latitude and longitude literal properties and also using the `schema:PostalAddress` class, which favours very detailed specification of the address. To complete the facility description we have literal properties for its name (`schema:name`) and a relevant web page (`schema:mainEntityOfPage`). We can use the object property `tao:aggregateRating`[31] to associate a lodging facility to an overall rating, modelled with a node of type `tao:NormAggregateRating`[32] annotated using the data property `tao:normRatingValue` to specify a float value between 0 and 1. A lodging facility can also be associated, through the property `schema:subjectOf`, with a textual description modelled using the `tao:LodgingDescription` class[33]. Finally, lodging facilities can be connected, using the `schema:review` property, to one or more user reviews, modelled using the `schema:UserReview` class. Each review is characterised by the date of creation and associated, using the `schema:reviewRating` property, with a rating (vote)

[28] Definition from Law Insider, see https://www.lawinsider.com/dictionary/lodging-facilities

[29] As previously introduced we refer to taxonomy as a hierarchy of classes connected with rdfs:subClassOf property.

[30] https://www.geonames.org/ontology/mappings_v3.01.rdf

[31] `tao:aggregateRating` is defined as a subproperty of `schema:aggregateRating` (relation not shown in Figure 2).

[32] `tao:NormAggregateRating` is defined as a subclass of `schema:AggregateRating` (relation not shown in in Figure 2).

[33] `tao:LodgingDescription` is a subclass of `schema:CreativeWork`. (relation not shown in Figure 2).

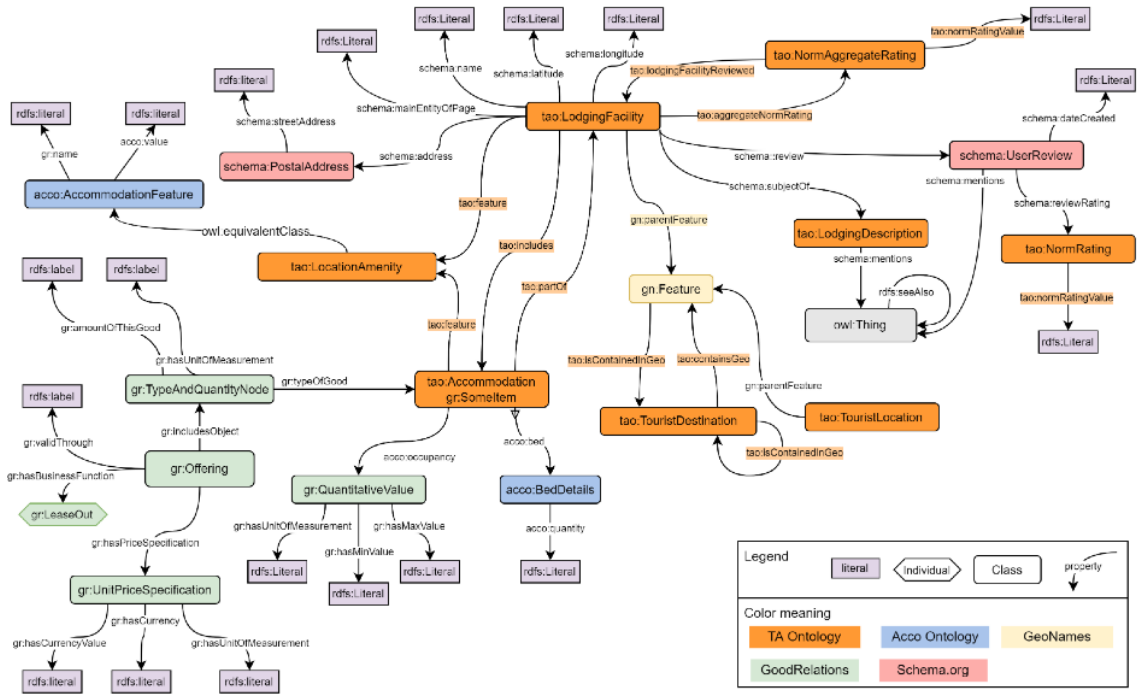


Fig. 2. TAO ontology schema

modelled with a `tao:NormRating` class[34], that can be used to specify the normalised rating in a specific review. The facility description and the reviews can mention every kind of entity, including those defined in other knowledge graphs (DBpedia and GeoNames) using the `schema:mentions` property.

This information will be typically extracted from the text of descriptions and reviews with various entity linking techniques.

The `tao:Accommodation` class, analogously to `schema:Accommodation`, represents the actual relevant units of the lodging facility that are offered for rental. It is formally distinct[35] from the physical place where the accommodations are located, which is modelled with the `tao:LodgingFacility` class instead. TAO uses the `tao:includes` object property to define the relation between a lodging facility and one of its accommodations. In order for the TAO ontology to maintain a certain degree of compatibility with the Accommodation Ontology, and potentially reuse semantic entities and annotations expressed using it, we defined the `tao:Accommodation` class as a subclass of `acco:Accommodation`. In this way, if a node in the KG is a member of `tao:Accommodation` it is also a member of `acco:Accommodation`, and all the properties defined in the Accomodation ontology for accommodations are still valid. On the contrary, not all the nodes that are members of `acco:Accommodation` are also members of `tao:Accommodation`.

Following GoodRelations best practices, a lease out offering a `tao:Accommodation` individual is modelled using a combination of GoodRelations classes to define the offering price, type, and quantity:

– the individual is also defined by type `gr:SomeItem`[36];

[34] `tao:NormRating` is defined as a subclass of `schema:Rating` (relation not shown in in Figure 2).
[35] Using `owl:disjointWith` property
[36] Besides being of type `tao:Accommodation`

- the offering itself is modelled with a node of type `gr:Offering`, which has an end of validity expressed with the `gr:validThrough` data property and which is characterised with a specific business function using `gr:hasBusinessFunction` to specify that is a `gr:LeaseOut`[37];
- the offering includes the accommodation indirectly through a `gr:TypeAndQuantityNode` node using the `gr:includesObject` property and can define its price through a `gr:UnitPriceSpecification` node;
- a `gr:TypeAndQuantityNode` node is used to specify which `tao:Accommodation` node is offered (through the `gr:typeOfGood` relation), the amount of the good included (using `gr:amountOfThisGood` data property) and the unit of measure for the amount included (using `gr:hasUnitOfMeasurement` data property);
- a `gr:UnitPriceSpecification` node is used to specify the price (using `gr:hasCurrencyValue` data property), the currency (using `gr:hasCurrency` data property), and what you are getting for the price (using `gr:hasUnitOfMeasurement`) i.e., a DAY in the accommodation.

The occupancy accommodation is modelled by using the `acco:occupancy` property whose value is a `gr:QuantitativeValue` object, which uses the `gr:hasUnitOfMeasurement` to specify "C62" literal (used by GoodRelations to indicate "one piece" of something, in this case, a person[38]) as well as the `gr:hasMinValue` and `gr:hasMaxvalue` relations to define the minimum and maximum number of allowed persons. To model an amenity offered by a lodging facility as a whole or as part of a specific accommodation TAO uses the `tao:LocationAmenity` class, which is defined as an equivalent class of `acco:AccommodationFeature` for compatibility with the Accommodation Ontology. It also uses the `tao:feature` property to associate a lodging facility or an accommodation with one or more amenities.

A tourist location (e.g. London's Big Ben or the city of Alghero) is a point or area of interest from a tourist point of view and is modelled with a `tao:TouristLocation` class, which is a subclass of both `schema:Place` and `gn:Feature`. A tourist destination (e.g., Sardinia) is defined as a place that is central to the decision to take the trip and is modelled with a `tao:TouristDestination` class, which is declared as `owl:equivalentClass` of `schema:TouristDestination` and as a subclass of `gn:Feature`. Tourist locations and lodging businesses can be included in a tourist destination using the property `tao:isContainedInGeo`.

For instance, if a tourist destination includes the City of London, all `tao:LodgingFacility` individuals in the City of London (according to `gn:parentFeature` property) are also considered within the same destination. This is because the TAO ontology includes an axiom that defines a chain of properties that state that if `X gn:parentFeature Y` and `Y tao:isContainedInGeo Z`, then `X tao:isContainedInGeo Z`, which can be expressed in functional-style syntax as: `SubObjectPropertyOf( ObjectPropertyChain( gn:parentFeature tao:isContainedInGeo ) tao:isContainedInGeo ).`

TAO includes several taxonomies describing the hierarchical relationships of relevant classes, including:

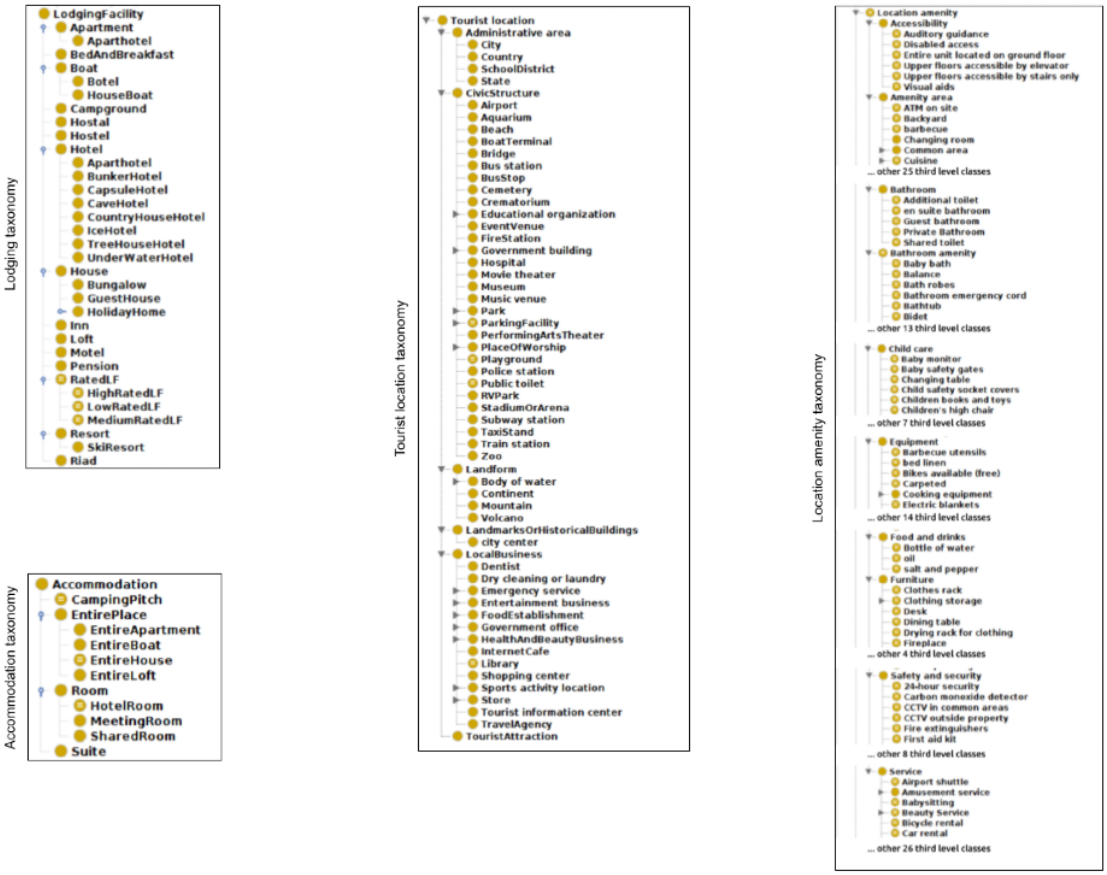
1. the *lodging taxonomy* with 35 types of lodging facilities (e.g., `tao:Hotel`, `tao:Apartment`, `tao:House`) across 4 levels;
2. the *accommodation taxonomy* with 17 types of accommodations (e.g., `Room`, `EntireApartment`, `Suite`) across 4 levels;
3. the *location amenity taxonomy* with 343 types of amenities (e.g., `Wifi`, `Minigolf`, `Dryer`) across 5 levels;
4. the *tourist location taxonomy* with 146 types of tourist locations (e.g., `City`, `Museum`, `Mountain`) across 5 levels;

Figure 3 reports the first three levels of each taxonomy. For each sub-class in a taxonomy we can have one or more of the following implementation:

- if a class is conceptually related to a similar class in other ontologies (e.g., DBpedia), this is modelled with the annotation property `rdfs:seeAlso`;

[37] An individual of type `gr:BusinessFunction` defined in the GoodRelations ontology
[38] http://www.heppnetz.de/ontologies/goodrelations/v1#UnitPriceSpecification

Fig. 3. A tree representation of the four taxonomies included in the TAO ontology expanded to the third level (some class removed in the location amenity taxonomy for sake of clarity and space).



– if a class is derived from other ontologies, we track the provenance using the `dc:source` property to indicate the original class [39];
– if a class extension[40] is the same as the extension of a class in other ontologies we link them with the `owl:equivalentClass` property [41], or the `rdfs:subClassOf` property if it is narrower[42];
– for each class, we use `rdfs:label` to indicate the primary label and `skos:altLabel` to indicate alternate labels;
– disjunctive axioms are added when appropriate to better support the reasoning.

The first taxonomy describes the different types of lodging facilities and their sub-types like in the case of Aparthotel, which is a special case of a hotel. We also introduce a special case with `tao:RatedLF` and its sub-classes which are used to automatically classify *Lodging facilities* according to their ratings

[39] Note that `dc:` stands for Dublin Core ;
[40] The set of individuals that are members of the class.
[41] It is the case of `tao:TouristDestination` which is declared to be `owl:equivalentClass` of `schema:TouristDestination`
[42] It is the case of `tao:EntireApartment` which is declared to be `rdfs:subClassOf` of `acco:Apartment` because in the Accommodation ontology `acco:Apartment` can refer to an apartment as a lodging facility or as an actual accommodation offered on lease.

(tao:NormAggregateRating). Specifically, tao:NormAggregateRating has 3 sub-classes that are defined using a data property restriction[43] on tao:normRatingValue:

- tao:LowNormRating class is defined for $0 \leqslant$ tao:normRatingValue < 0.6
- tao:MediumNormRating class is defined for $0.6 \leqslant$ tao:normRatingValue < 0.75
- tao:HighNormRating class is defined for $0.75 \leqslant$ tao:normRatingValue $\leqslant 1$

A rated lodging facility is also part of tao:RatedLF (rated lodging facility) class[44] and it can also be inferred whether it is part of one of the following three sub-classes:

- is part of tao:HighRatedLF class if it is associated[45] with a tao:HighNormRating node;
- is part of tao:MediumRatedLF class if it is associated with a tao:MediumNormRating node;
- is part of tao:LowRatedLF class if it is associated with a tao:LowNormRating node;

When modelling *accommodations*, we distinguished two general offerings: (i) entire place (i.e., EntirePlace), and (ii) room (i.e., Room). For these, we also defined sub classes (e.g., EntireHouse for EntirePlace, HotelRoom for Room). In addition, we modelled two special cases (i.e., CampingPitch and Suite), which are not covered by the general cases. When appropriate, we used equivalence axioms to add useful constraints as in the case of HotelRoom which must be part of one Hotel. Moreover, to support high compatibility between TAO and the Accommodation Ontology, we defined the accommodation classes of TAO as subclasses of the Accommodation Ontology ones (e.g., tao:CampingPitch is a subclass of acco:CampingPitch).

In the case of *location amenities*, we added equivalence axioms to support a certain degree of mapping with how specific accommodation features could be more probably defined using the Accommodation ontology approach[46]. To this end, each sub-class in this taxonomy is also declared as owl:equivalentClass to an anonymous class defined in accordance to Accommodation Ontology prescriptions[47]. Thus we define each anonymous class as a subclass of acco:AccommodationFeature and as an owl:intersectionOf of owl:Restriction based on gr:name and acco:value data properties from GoodRelations. An example is given below in Turtle:

```
tao:AirportShuttle rdf:type owl:Class ;
  owl:equivalentClass [
    rdf:type owl:Class
    owl:intersectionOf (
      acco:AccommodationFeature
      [
       rdf:type owl:Restriction ;
       owl:onProperty acco:value ;
       owl:hasValue "yes"@en
      ]
      [
       rdf:type owl:Restriction ;
       owl:onProperty gr:name ;
       owl:hasValue "Airport␣Shuttle"@en
      ]
    ) ;
  ] .
```

[43] See OWL2 specifications https://www.w3.org/TR/2012/REC-owl2-syntax-20121211/#Data_Property_Restrictions.

[44] Because this class is defined using an existential quantification on the object property tao:aggregateNormRating that has some tao:NormAggregateRating.

[45] Using tao:aggregateNormRating object property

[46] Because there is not a defined taxonomy but a textual label is used to define a specific feature we can only try to guess the label most probably used.

[47] It is defined as "a structured value representing the feature of an accommodation as a property-value pair of varying degrees of formality"; see http://ontologies.sti-innsbruck.at/acco/ns.html#AccommodationFeature

In this way a reasoner can map to the appropriate `tao:LocationAmenity` sub-class an accommodation feature defined using `acco:value` and `gr:name` as prescribed in the Accommodation ontology specifications.

Tourist locations are modelled, whenever possible, according to their respective GeoNames feature codes. This is done by declaring them as `owl:equivalentClass` to an anonymous class which is a restriction on the property `gn:featureCode` that must have an appropriate value from the GeoName feature codes list[48]. An example is given below:

```
tao:Zoo rdf:type owl:Class ;
 owl:equivalentClass [
    rdf:type owl:Restriction ;
    owl:onProperty <http://www.geonames.org/ontology#featureCode> ;
    owl:hasValue <http://www.geonames.org/ontology#S.ZOO>
 ] ;
 rdfs:subClassOf <http://www.geonames.org/ontology#Feature> ;
 rdfs:label "Zoo"@en .
```

3.3.4. TAO enrichment

The TAO ontology was produced using a programmatic approach instead of manual editing. Specifically, we developed a building process in Python[49]. This approach allowed us to automate some aspects of the ontology building process (e.g., creation of axioms), to version the code instead of just the final ontology, to reduce human errors, and to easily produce inline documentation about the ontology creation process. We also release an open-source version of the Python code that builds the TAO ontology as a Jupyter Notebook[50].

The TAO ontology has to be able to model information derived from typical data sources in this domain, such as Booking.com and AirBnB, which provide (semi)structured data as key/value properties and unstructured data as text regarding lodging facilities, accommodations, amenities, and user reviews. Therefore, we developed a human in the loop strategy, reported in Figure 4, to produce new versions of TAO by continuously enriching the ontology with new types of `tao:LodgingFacility`, `tao:Accommodation` and `tao:LocationAmenity` or new labels for existing types which are derived from the source data. This solution allows us to keep the ontology updated and well aligned with the actual data.

We start with the basic version of the ontology (orange bullet 1 in the figure), set up external imports, and define classes, properties, and axioms (bullet 2). To further enrich TAO, our ontology engineers analyse several analytics about the most frequent terms associated with facilities, accommodations, and amenities. Then they use them to create new relevant classes in the ontology (bullet 5) or add additional labels to an existing class (bullet 6). For example, the mini-golf amenity class was identified in the amenities list extracted from Booking.com, while the holiday home lodging facility alternative label “holiday house” was extracted from AirBnB texts.

The analytics are produced by two automatic pipelines (3 and 4). The first processes structured data, extracting a list of all possible values for categorical fields that refer to accommodation types, accommodation features, or type of lodging facilities. The second processes the unstructured text, extracting and ranking frequent uni-grams and bi-grams from the text descriptions of lodging facilities or user reviews. To achieve this, we relied on Spacy Python library[51] to perform the following sub-tasks: 1) identify language to filter English text only (bullet A), 2) clean the text from special characters (bullet B), 3) perform text frequency analysis (bullet C), and 4) perform TF-IDF analysis (bullet D).

Finally, the ontology engineers produce a mapping file that is used (bullet 7) to create new classes, sub-class relations (using the `rdfs:subClassOf` property), or add labels to existing classes (using the `skos:altLabel` property). We also track the provenance of these changes using the `dc:source` property for classes and the `rdfs:comment` property for labels. The final process (bullet 8) produces a new version of the TAO ontology.

[48] See https://www.geonames.org/export/codes.html
[49] The code is based on owlready2 [30].
[50] See https://github.com/linkalab/tkg/tree/main/tao_modelling
[51] https://spacy.io/

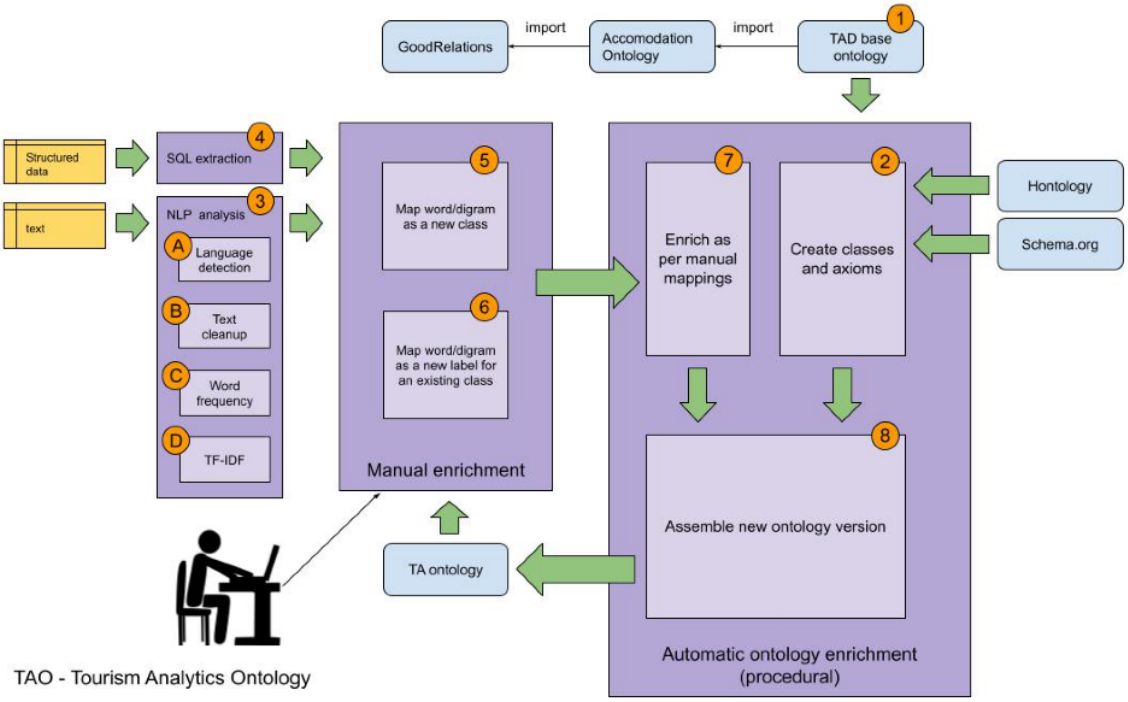


Fig. 4. Ontology enrichment workflow

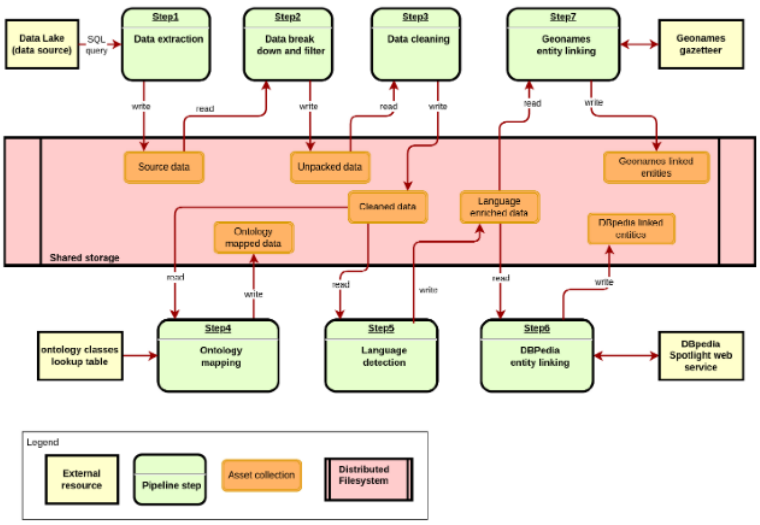
3.4. *Transform the data*

The transformation of data is the fourth phase in our approach to building our Tourism Knowledge Graph. Specifically, this phase consists of transforming the information extracted from the data sources into a set of tables, which will be used in the next phase (described in Section 3.5) to produce the actual knowledge graph triples. We devote this section to describing the data transformation process and the technologies for implementing it. Depending on the source data structure and the desired output, we can apply different transformation steps organised as data pipelines. A data pipeline is a series of computational steps organised as direct acyclic graph where the output of one step becomes the input of one or more downstream steps.

Figure 5 depicts the complete data transformation workflow. Each step can materialise its output (henceforth referred to as *asset*), saving it as a file or storing it in a database application. From the diagram, we can observe four types of components:

1. external resources that are used during the pipeline execution (yellow boxes) representing
   (a) tables in the data lake,
   (b) files mapping text strings to TAO ontology classes,
   (c) DBpedia Spotlight public web service,
   (d) GeoNames gazetteer exposed as an Elasticsearch end-point;
2. pipeline execution steps (green boxes);
3. collections of data assets (files) produced by the execution steps (orange boxes);
4. a distributed file system that stores all the data assets produced and consumed by one or more processing step (pink box).

Fig. 5. High level data transformation workflow diagram



At a high level, the workflow consists of 7 steps. The first 3 steps are executed on both structured (key/values) and unstructured (text) data:

1. **Data extraction**: acquires the source data and produces the *Source data assets collection*;
2. **Data break down and filter**: rearranges the data structure and filters out unnecessary data; works in combination with the Data cleaning step and materialises the *Unpacked data assets collection*;
3. **Data cleaning**: reads from the *Unpacked data assets collection*; corrects or removes corrupt, duplicated or inaccurate data; produces the *Cleaned data assets collection*;

The cleaned data is processed differently depending on if it is structured or unstructured. For structured data, the final step is:

4. **Ontology mapping**: uses heuristic rules to identify what ontology class should be used to model each entity described in the data; it produces the *Ontology mapped data assets collection*;

For unstructured data, our objective is to enrich TKG with links from lodging descriptions and user reviews to semantic entities in DBpedia and GeoNames. In this way, TKG would be connected to external knowledge graphs revealing what tourists and business owners are considering important and worth noting. To perform this enrichment we perform entity linking, in three more steps:

5. **Language detection**: identifies the language used in texts to process only English text; produces the *Language enriched data assets collection*;
6. **DBpedia entity linking**: descriptions and reviews texts are processed to recognise and link DBpedia entities; produces the *DBpedia linked entities data assets collection*;
7. **GeoNames entity linking**: descriptions and reviews texts are processed to recognise and link GeoNames entities; produces the *GeoNames linked entities data assets collection*.

In the following subsections, we describe each processing step as well as the employed technological architecture.

3.4.1. Data extraction

As the first step, we extracted the relevant data from the source data lake. The extraction process is performed using a SQL big data engine[52]. During this process, the data is also combined and arranged to be more easily

[52] Amazon Athena, see https://aws.amazon.com/en/athena

processed in the following steps (e.g., unique ids are calculated, nested columns are exploded). This produces the *Source data* assets collection which consists of:

1. hospitality_supply_assets: containing information about lodging facilities, accommodation, and offers.
2. hospitality_demand_assets: containing information about user reviews.

3.4.2. Data break down and filter

This second step organises and structures the information produced in the previous step. Specifically, we need to:

1. break down the information so that we have a distinct asset for each semantic entity we want to model as triples (e.g., lodging facility, accommodation, offer, review);
2. apply a flat structure to the data, because some columns contain complex data structures as arrays or key/value structures;
3. separate text blobs from the other data preserving their relation to the semantic entity they refer to (e.g., the lodging facility description, the review content).

We can obtain the right structure using specific data pipelines that produce multiple assets out of a single one, flattening the data and filtering out unnecessary columns. This produces an unpacked version of the assets for each source:

1. hospitality_unpacked_supply_assets: containing unpacked information about lodging facilities, accommodation, and offers.
2. hospitality_unpacked_demand_assets: containing unpacked information about user reviews.

3.4.3. Data cleaning

Here we correct or remove corrupt or inaccurate records from the assets produced in the previous step. In particular, we need to drop duplicated records, remove special characters, normalize categorical fields, normalize date and numeric fields.

From hospitality_unpacked_supply_assets, the Data Cleaning step produces:

1. lodging_assets - containing all structured data relative to lodging facility entities (i.e., entities of type `tao:LodgingFacility`); for each lodging facility a unique ID is produced;
2. lodging_description_assets - containing all descriptions relative to a lodging facility (used to perform Named Entity Extraction and Linking);
3. accommodation_assets - containing all structured data relative to accommodation entities (i.e., entities of type `tao:Accommodation`) in a lodging facility; for each accommodation, a unique ID is produced;
4. offers_assets - containing all structured data relative to accommodation offers (i.e., entities of type `gr:Offering` that will be modelled as prescribed by the Accommodation Ontology); for each offer, a unique ID is produced;
5. amenities_assets - containing all accommodation features (a.k.a. amenities) that are related to a lodging facility and/or to an accommodation.

Instead, from hospitality_unpacked_demand_assets, the Data Cleaning produces:

1. reviews_assets - containing all structured data relative to user reviews about a lodging facility; for each review a unique ID is produced;
2. reviews_content_assets - containing all text content for user reviews about a lodging facility (used to perform Named Entity Extraction and Linking);

3.4.4. Ontology mappings

At this stage, we identify and map the classes of the structured data to transform them into triples.

For instance, if a lodging business is represented as a record like:

hotel_id	name	structure_type
9f40f613d308cf80	Chelsea BnB	Bed and breakfast

after the ontology mapping step, a new field lf_class (lodging facility class) is added with the "BedAndBreakfast" class name:

hotel_id	name	structure_type	lf_class
9f40f613d308cf80	Chelsea BnB	Bed and breakfast	BedAndBreakfast

Structured data include categorical columns that refer to taxonomic concepts in the TAO ontology. In particular there are three taxonomies in the ontology that we have to reconcile with categorical columns in the data:

1. lodging facility types: for each lodging table record we have a text field that contains the name of the lodging facility type; this field can be used to associate the correct `tao:LodgingFacility` subclass to the individual lodging facility the record is about;
2. accommodation types: for each accommodation table record we have a text field that contains the name of the accommodation facility type; this field can be used to associate the correct `tao:Accommodation` subclass to the individual accommodation the record is about;
3. accommodation features (amenities) types: for each amenity table record we have an accommodation feature associated with a specific lodging facility (via an external key ID that refers to the lodging table). This field can be used to associate the correct `tao:LocationAmenity` subclass to the individual amenity the record is about.

To perform the reconciliation we use a heuristic process based on rules that can identify the most appropriate class to use to model an entity. The heuristic process uses lookup tables extracted from the ontology where we have each class associated with each of its labels. In this way, we leverage the ontology enrichment we already described in Section 3.3.4. The reconciliation is thus performed by adding the correct class name in a new column of the data table so that it can be used during the triple creation phase. The ontology mapping step produces new types of assets that are part of the *Ontology mapped data* asset collection:

1. classified_lodging_assets;
2. classified_accommodation_assets;
3. classified_amenities_assets.

These assets will be fed into the triple creation process.

3.4.5. Language detection

This step applies a language detection algorithm [38] to the text contained in the lodging description and reviews content tables. The detected language is used to enrich *lodging_description_assets* and *reviews_content_assets* with a new language column so that subsequent steps can process only English texts. The enriched assets are part of the *Language enriched data* asset collection.

3.4.6. DBpedia entity linking

To perform the Entity Linking task against DBpedia we have applied DBpedia Spotlight [13, 33] APIs[53] to the English text contained in the lodging description and reviews content tables. DBpedia Spotlight identifies and annotates entities based on the following pipeline process:

- Spotting: identifies possible entity mentions (surface forms) from the original input text.
- Candidate selection: selects the DBpedia resources that are candidate meanings for each surface form.
- Disambiguation: determines which candidate is the most likely resource for each surface form.
- Filtering: adjusts the annotation task based on the user requirements.

For the filtering step, we restricted the annotation scope to the following type of entities: `DBpedia:Activity`, `DBpedia:Food`, `DBpedia:Holiday`, `DBpedia:MeanOfTransportation`, `DBpedia:Place`, `Schema:Event`, `Schema:Place`. The result of the DBpedia entity linking process produces two new types of assets which are part of the *DBpedia linked entities* asset collection:

[53] https://www.dbpedia.org/resources/spotlight/

1. lodging_dbpedia_linked_assets - containing a record for each DBpedia entity linked to a lodging facility identified by its unique ID;
2. review_dbpedia_linked_assets - containing a record for each DBpedia entity linked to a user review identified by its unique ID.

We can use these assets in the triple creation process.

3.4.7. GeoNames entity linking

This step performs an Entity Linking task against GeoNames so that places named in the lodging descriptions or the reviews are linked to the GeoNames corresponding entities.

To this end, we employed an open-source software called Mordecai[54] [24], a full-text geoparsing system that extracts place names from the text, resolves them to their correct entries in a gazetteer, and returns structured geographic information for the resolved place name. Mordecai is based on a language-agnostic architecture that uses word2vec [34] for inferring the correct country for a set of locations in a piece of text. As a gazetteer, it uses a custom-built Elasticsearch database populated with GeoNames data. Mordecai is integrated within the Spacy library[55]. Analogously to what is described in Section 3.4.6 for DBpedia, we used Mordecai to process all English text contained in the lodging description and review content tables. The result of the GeoNames entity linking process produces two new types of assets which are part of the *GeoNames linked entities* asset collection:

1. lodging_geonames_linked_assets - containing a record for each GeoNames entity linked to a lodging facility identified by its unique ID;
2. review_geonames_linked_assets - containing a record for each GeoNames entity linked to a user review identified by its unique ID.

3.4.8. Implementation strategy

To support the data transformation described in the previous sections, we identified the following requirements for our technological architecture:

- Data-driven,
- Flexible and easily extensible,
- Scalable in a distributed computing environment,
- Easily manageable,
- Easily instrumented for lineage (a.k.a. provenance) metadata collection.

Following the requirements, the data computation is organised using the pipeline approach already described. This approach is optimal to create a distributed computation if the intermediate and final materializations are stored on a distributed file system. This is the same approach adopted by Apache Spark and other big data frameworks.

To manage the execution of a set of data pipelines, we used Dagster[56], an open-source orchestrator service. Dagster can be deployed on a single machine or a distributed environment like Kubernetes or AWS Elastic Container Service clusters. Thanks to this flexibility we started using a single machine to simplify the deployment process, without losing the opportunity to switch to a distributed architecture in the future. Dagster can also expose metadata about the execution of each pipeline and the produced assets, enabling our system to generate provenance information for the Knowledge Graph. The data transformation code is developed using Python Pandas[57] library. We released the pipelines built on Dagster as an open-source resource for the paper[58].

[54] https://github.com/openeventdata/mordecai
[55] Only Spacy v2.x is supported at the moment
[56] https://dagster.io/
[57] https://pandas.pydata.org/
[58] See https://github.com/linkalab/tkg/tree/main/kg_pipelines

3.5. Triples creation

This section presents the fifth phase for the creation of the Tourism Knowledge Graph, shown in Fig. 1, which deals with the creation of the RDF triples. For this, we leveraged the RDF Mapping Language (RML) [14], to build data pipelines for producing RDF triples[59] from text files, and subsequently save them in a serialised format. The RML language is a declarative language used to define how Linked Data is generated from corresponding data sources, using annotations provided through vocabulary terms. RML can use also files as data sources, which is very useful for our scenario. An RML transformation requires the following elements[60]:

1. an RML processor that performs the actual transformation;
2. an input to the RML mapping which is called input data source;
3. an RML mapping, that defines the rules of conversion from any input (structured) data to RDF.

These rules define how to convert an input record (or row, XML element, JSON object) to one or more RDF triples. They are independent of the process of executing the conversion, thus decoupling the implementation from the rules themselves.

In our implementation, we used RMLMapper [15] which is an open-source RML processor developed in Java[61]. We designed different mappings to handle the different sources, i.e., Booking.com and AirBnB.

The output of the RML processor is a set of files containing the RDF triples serialisation in n-quads[62].

To improve the development, debugging and maintenance of RML triple maps we adopted YARRRML [26], a human-readable text-based representation for declarative generation rules[63]. In the following paragraphs, we will examine an example of how a Lodging Facility and all the other related entities can be expressed in TKG by a set of triples created through the process described above. We will represent triples in a graphical form to better understand the knowledge graph structure.

3.5.1. High level Tourism Knowledge Graph triples structure

The triples creation process for describing accommodation offers follows the Accommodation Ontology prescriptions and is compliant with GoodRelations and Schema.org best practices. Figure 6 shows an example of TKG structure at a high level. We can observe a lodging facility (`:lodging_1`) that is the subject of a descriptive text (`:lodging_description_1`), that has one review (`:review_1`), and contains one accommodation (`:accommodation_1`). A description is a special kind of creative work (modelled using the `tao:LodgingDescription` class) that can mention one or more real entities like places or food. In the example, the description mentions the Big Ben tower (through the `schema:mentions` property). Also, reviews are considered creative works in Schema.org[64] and are thus related to other real-world entities using `schema:mentions` property. There is an offer (`:offer_1`) to lease out an accommodation that is contained in the lodging facility; `:offer_1` is related to the offered accommodation (`:accommodation_1`) utilizing (`:quantity_1`) node whose properties define what is offered using the property `gr:hasUnitOfMeasurement` (e.g., DAY) and in what quantity using the property `gr:amountOfThisGood` (e.g., 2).

3.5.2. Lodging facility entities triple structure

In Figure 6, we can steer our focus to observe triples modelling a lodging facility, which includes:

1. an address entity (`:address_1`), modelled as a `schema:PostalAddress` class that gives us great flexibility to define the facility position;
2. one or more accommodation features entities that are associated with the lodging facility using the `tao:feature` property; in our example, we have the node `:amenity_1` of type `tao:Parking`[65].

---

[59] https://www.w3.org/TR/2004/REC-rdf-concepts-20040210/#dfn-rdf-triple

[60] See https://rml.io/specs/rml/

[61] https://github.com/RMLio/rmlmapper-java

[62] https://www.w3.org/TR/n-quads/

[63] https://rml.io/yarrrml/

[64] See https://schema.org/UserReview

[65] In general the class of the amenity should be the most appropriate TAO ontology class among all the subclasses of `tao:LocationAmenity` as detected during the Ontology mapping step described in Section 3.4.4

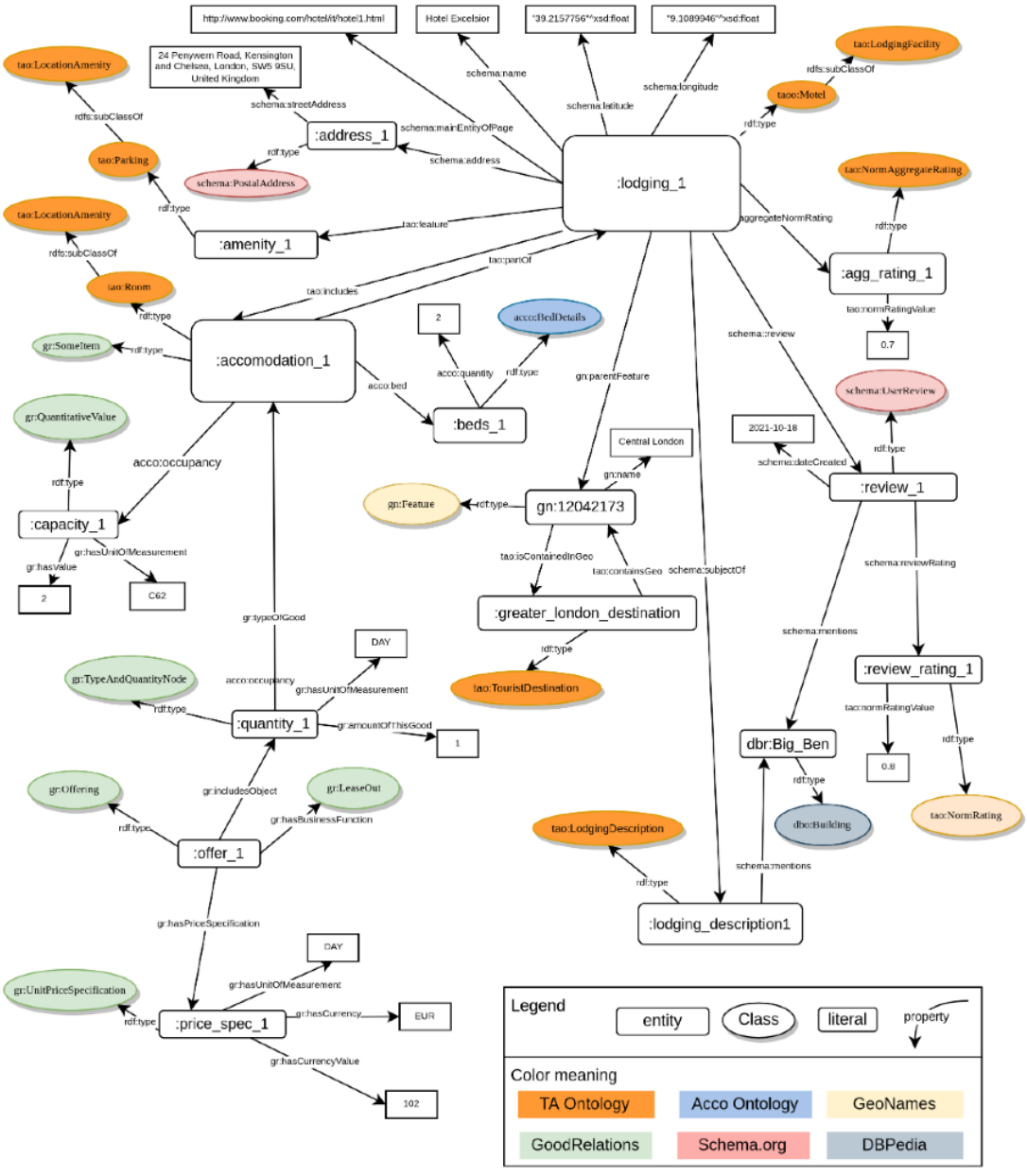


Fig. 6. A high level example of the main entities used in TKG

3. an aggregated rating entity (`:agg_rating_1` in our example) that is used to model the overall user rating for the lodging facility (which is related to the ratings expressed by the single users' reviews) that specifies the vote in a normalised range from 0 to 1.

3.5.3. Accommodation entities triple structure

Accommodation is always related to a lodging facility, in compliance with the Accommodation ontology, and it includes:

1. its maximum and minimum occupancy capacity, using a `gr:QuantitativeValue` node (`:capacity_1` in our example);
2. its provision of beds, using an `acco:BedDatails` node (`:beds_1` in our example);
3. the type of accommodation[66] (using one of the TAO ontology classes like `tao:Room`).

3.5.4. Offer entities triple structure

We describe a commercial offer for leasing out an accommodation leveraging GoodRelations. As shown in Figure 6 an offer can be expressed in terms of:

1. a node (`:quantity_1`) of type `gr:TypeAndQuantityNode` used to specify the number of days it is offered using `gr:amountOfThisGood` and `gr:hasUnitOfMeasurement` properties;
2. a node (`:price_spec_1`) of type `gr:UnitPriceSpecification` used to specify the price and currency for each day using the `gr:hasUnitOfMeasurement`, `gr:hasCurrency` and `gr:hasCurrencyValue` properties.

3.5.5. User reviews triple structure

A user review of the lodging facility is represented in TKG by two entities:

1. a node (`:review_1`) of type `schema:UserReview` with a `schema:dateCreated` property used to specify the review creation date;
2. a node (`:review_rating_1`) of type `tao:NormRating` that is used to specify the actual rating normalised to 1 (using `tao:normRatingValue`) property.

3.6. Knowledge Graph publishing and validation

In this section, we present the triple store publishing TKG, discuss how to identify the different resources in the knowledge graph, and finally how we encoded the provenance. For publishing the knowledge graph we relied on Ontotext GraphDB. The knowledge graph itself is a collection of multiple RDF graphs. Each RDF graph has an associated URI which defines its graph name. For both Booking.com and AirBnB we created two named graphs:

1. a hospitality named graph that contains all the triples from the `hospitality_triples_assets` produced from a specific source for a certain tourist destination (e.g., London or Sardinia);
2. a linked entities named graph that contains all the triples from the `entity_linked_triples_asset` produced from a specific source for a certain tourist destination.

A named graph has a custom URI with this structure:
`base_url/tourist_destination/source/enrichment`
Specifically:

1. base_url: `http://tourism.kg.linkalab-cloud.com/ng/`[67]
2. tourist_destination: is used to identify a tourist destination by name (e.g., London or Sardinia)
3. source:
   (a) bkg: is used to identify the source Booking.com ;
   (b) air: is used to identify the source AirBnB;
4. enrichment:
   (a) internal: is used for all the RDF assets that are produced with no entity linking during the transformation phase;

[66] As detected during the Ontology mapping step described in 3.4.4
[67] ng stands for named graph.

(b) dbpedia_el: on assets that are enriched with Entity Linking against DBpedia;
(c) geonames_el: on assets that are enriched with Entity Linking against GeoNames.

As an example, the named graph name which is a collection of triples about London hospitality, produced from Booking.com (semi-)structured data (with no entity linking) would be: `http://tourism.kg.linkalab-cloud.com/ng/london/bkg/internal`.

The use of named graphs implemented as described simplifies the distinction of resources related to a specific tourist destination because we can use the named graphs in SPARQL queries and identify subsets of data trough Implicit Graphs using Triple Pattern Fragments[68] (TPF) [43, 44]. This distinction is also useful to express provenance metadata at the named graph level as described in Section 3.6.1.

Concerning identifying a resource in the knowledge graph, we use URIs that explicitly contain the external source (e.g., Booking, AirBnB), and the type of resource. The resource URI is structured as follows: `base_url/resource_type/source/unique_id`

In particular:

1. base_url: `http://tourism.kg.linkalab-cloud.com/`
2. resource_type:
   (a) lf: is used to identify Lodging Facility entities;
   (b) ac: is used to identify Accommodation entities;
   (c) of: is used to identify Offering entities;
   (d) rv: is used to identify User Reviews entities.
3. source:
   (a) bkg: the resource is derived from Booking.com;
   (b) air: the resource is derived from AirBnB.
4. unique_id: is an identifier produced by the data transformation phase which is unique for the data source.

As an example, the following URI identifies a lodging facility derived from AirBnB: `http://tourism.kg.linkalab-cloud.com/lf/air/30840569`.

The Tourism Analytics ontology is published as a turtle file at the following URI: `http://purl.org/tao/ns`[69]. To access a specific class or property the hash URI approach is adopted[70] (e.g., `http://purl.org/tao/ns#LodgingFacility` is the URI for LodgingFacility class).

3.6.1. Provenance and dataset metadata

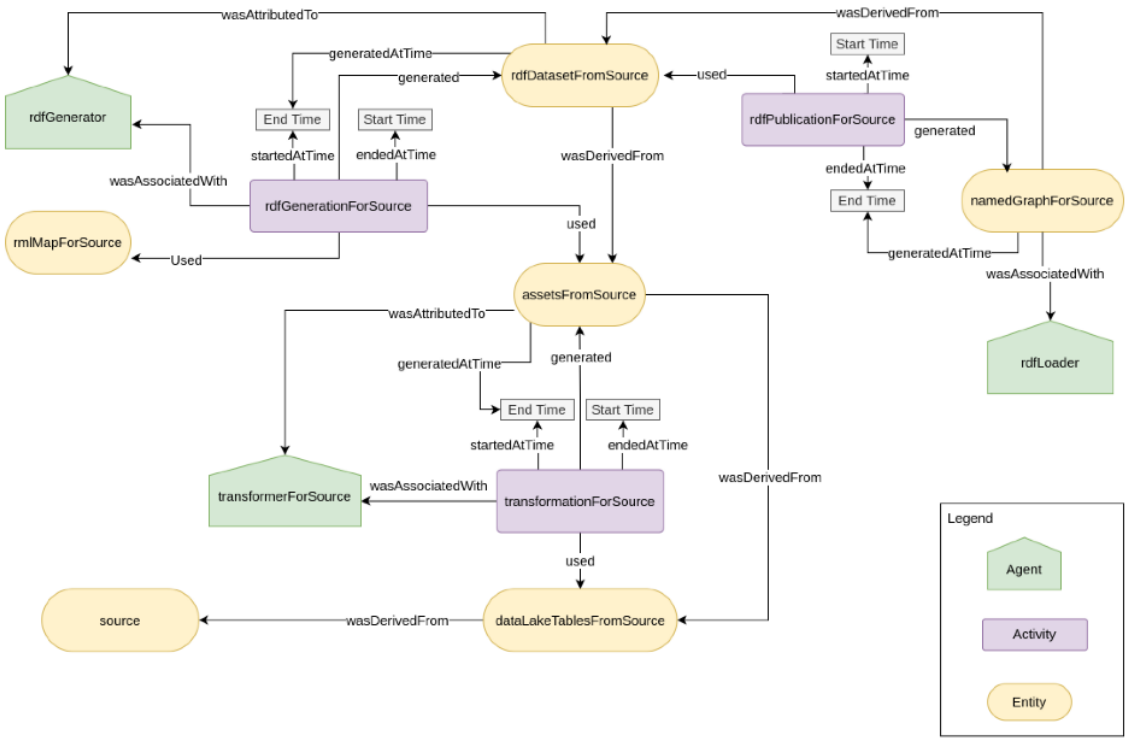
In a dedicated named graph, we loaded also the metadata triples describing the other named graphs and their provenance: `http://tourism.kg.linkalab-cloud.com/ng/meta/prov`. A named graph can be referenced using Quad Pattern Fragments[71] with a URI with the following structure: `base_url?graph=graph_name` where we have:

1. base_url: `http://tourism.ldf.linkalab-cloud.com/graph`
2. graph_name: is the URI associated with the named graph as its name

As an example, the named graph containing the triples about London hospitality produced from Booking.com (semi-)structured data (with no entity linking) would be:
`http://tourism.ldf.linkalab-cloud.com/graph?graph=http://tourism.kg.linkalab-cloud.com/ng/london/bkg/internal`. To express the provenance information we used the W3C PROV provenance model. This allows us to track the lineage of data assets produced during the data transformation and triple creation phases following a similar approach as that described in [15]. In PROV we have three main classes:

[68] http://linkeddatafragments.org/
[69] This is a redirect to http://schema.linkalab-cloud.com/tao.ttl
[70] See https://www.w3.org/TR/cooluris/#hashuri for an in-depth explanation.
[71] https://linkeddatafragments.org/specification/quad-pattern-fragments/

Fig. 7. A high-level provenance view of Tourism Knowledge Graph creation and publishing workflow



- `prov:Entity` - a physical, digital, conceptual, or other kinds of thing with some fixed aspects; entities may be real or imaginary;
- `prov:Activity` - something that occurs over a while and acts upon or with entities; it may include consuming, processing, transforming, modifying, relocating, using, or generating entities;
- `prov:Agent` - something that bears some form of responsibility for an activity taking place, for the existence of an entity, or another agent's activity.

Figure 7 shows a high level provenance schema describing how each of the named graphs is produced. Specifically, we can recognise the following PROV entities:

1. `source` - represents the web source for our data (e.g., Booking.com);
2. `dataLakeTablesFromSource` - represents the tables exposed by the data lake containing the data extracted from the source;
3. `assetsFromSource` - represents all the assets created during the transformation phase which are used to produce the RDF triples for a specific named graph;
4. `rmlMapForSource` - represents the RML map document used to produce the RDF triples for a specific named graph;
5. `rdfDatasetFromSource` - represents the RDF graph (serialised as one or more files) that is produced from the source using specific `assetsFromSource` and `rmlMapForSource` entities;
6. `namedGraphForSource` - represents the published named graph.

Moreover, in the same schema, we can identify the PROV Activities involved in the production of a specific named graph:

1. `transformationForSource` - performed to prepare/enrich the data for the triple creation;
2. `rdfGenerationForSource` - performed to produce the triples;
3. `rdfPublicationForSource` - performed to load the triples in the triple store as named graphs.

Finally, we can identify in the schema the following PROV Agents:

1. `transformerForSource` - represents the entire transformation pipeline described in Section 3.4;
2. `rdfGenerator` - represents the RML processor software (RMLMapper in our case);
3. `rdfLoader` - represents the agent that loads the RDF graph in the triple store.

The proposed PROV schema can be easily adapted to specify a particular named graph provenance information and can track: (i) when all triples in the named graph are created/updated, (ii) what assets are used to generate the triples, (iii) what RML mapping document was used to generate them. The same can be specified for all the assets produced by the transformation pipeline. The agent entities are also useful to track the software version used to produce each named graph.

4. Evaluation

We evaluated TAO and TKG according to functional, logical, and structural dimensions as suggested by previous works [6, 20]. In our case, the functional dimension is related to the intended use of TKG in the context of the tourism destination's analysis. It allows us to assert its ability to address requirements and offer a useful representation of the domain. For assessing the logical dimension, we verified that TKG can be successfully processed by a reasoner and produce sound additional knowledge. To conclude, the structural analysis of TKG focuses on assessing the topological properties of the graph. These analyses provide useful insights on design choices and can be used to iterative refine the knowledge graph.

To organise and document the evaluation activities, we identified a set of tests to be executed. Each test is described by a test case that specifies its inputs, conditions for the execution, testing procedure, and expected results. All the RDF files produced to test the functional and logical dimensions are available at https://github.com/linkalab/tkg/tree/main/validation.

4.1. Functional dimensions

To verify that the functional requirements are satisfied, we followed the **CQ (Competency Question) verification** approach proposed by Carriero et al. [6]. Specifically, this approach aims at testing whether the competency questions can be answered by running SPARQL queries on the KG. To evaluate this dimension we translated the 12 competency questions, formulated in Section 3.3.1, into SPARQL queries.

We used this process to drive the creation and refining of TAO, identifying missing classes or properties and adding them to the ontology. We also used it for verifying that TKG can answer in a meaningful way to all competency questions.

We implemented the test cases as RDF files modelled with the TestCase OWL meta-model (prefix `test:`), following Blomqvist et al. [4].

The execution consists of performing the relative SPARQL queries against TKG end point[72]. Queries were manually executed and the results were checked against the expected values. Some CQs required the execution of federated queries to access triples from DBpedia and GeoNames. To this end, we used the `SERVICE` keyword to access Ontotext FactForge SPARQL endpoint[73], which exposes both of them.

All the 12 competency question tests ran successfully. The following example shows a federated SPARQL query that aims to answer "*What are the apartments with wi-fi near at least 2 parks?*"[74]. As we can see by using the `SERVICE` directive we can query both knowledge graphs together:

[72] To access the SPARQL endpoint use http://tourism.sparql.linkalab-cloud.com/ with username:paper and password:journal_p4p3r2022!!

[73] See http://factforge.net/

[74] In this case a park is considered near the apartment if it is within a distance of 1 km.

```
PREFIX gdb-geo: <http://www.ontotext.com/owlim/geo#>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX gn: <http://www.geonames.org/ontology#>
PREFIX tao: <http://purl.org/tao/ns#>
PREFIX acco: <http://purl.org/acco/ns#>
PREFIX schema: <http://schema.org/>
PREFIX onto: <http://www.ontotext.com/>

SELECT ?lodge (SAMPLE(?name) AS ?apartment) (COUNT(?park) AS ?num_parks_nearby)
FROM onto:explicit  ## use only explicit statement without any inference
WHERE {
    { SELECT DISTINCT ?lodge  ?name ?lat ?long WHERE {
            ?lodge a tao:Apartment ; schema:latitude ?lat ;
            schema:longitude ?long ; schema:name ?name; tao:feature ?b.
            ?b  a tao:Wi-FiZone .  } }
     SERVICE <http://factforge.net/repositories/ff-news> {
            ?park gdb-geo:nearby(?lat ?long "1km"); gn:featureCode gn:L.PRK .
     }
}
GROUP BY ?lodge HAVING ( ?num_parks_nearby > 1)
ORDER BY DESC(?num_parks_nearby)
LIMIT 3
```

By running this query on our KG we obtain the following results.

Lodge	Apartment	Num_parks_nearby
http://tourism.kg.linkalab-cloud.com/lf/bkg/9bd5bef8f50e0e03	"1 Bedroom Luxury Apartment Chancery Lane"	"3"^^xsd:integer
http://tourism.kg.linkalab-cloud.com/lf/air/42701380	"2 bedroom basement apartment with 50 inch TV"	"3"^^xsd:integer
http://tourism.kg.linkalab-cloud.com/lf/bkg/51e2e2d011d57200	"3 Bedroom Palatial Apartment Chancery Lane"	"3"^^xsd:integer

All competency question test cases are available at https://github.com/linkalab/tkg/tree/main/validation/competency_questions[75]

4.2. Logical dimensions

To assess the logical dimension, we ran a reasoner on TKG and checked for any inconsistency. Specifically, we adopted two strategies suggested in Carriero et al. [6]:

1. **inference verification**, which checks if the inference over the KG produces the expected results (as an example, if a `tao:HotelRoom` accommodation is part of a generic `tao:LodgingFacility` we can infer that the latter is a Hotel);
2. **error provocation**, which aims to provoke an inconsistency error by injecting data that violates the requirements (as an example, an instance of a lodging facility can not be defined of type `tao:Hotel` and `tao:BedAndBreakfast` at the same time).

In the following subsection, we will describe more in detail how we conducted these two tests.

[75] We suggest to use Protégé for opening the competency questions test cases files.

4.2.1. Inference verification

For evaluating this dimension, we analysed the inferences made by the reasoner and compared them with the expected results. For instance, let us consider a `LodgingFacility` individual (named `Hotel Splendor`) which is related to `Greater London`, a second-level administrative division defined in GeoNames[76], through the ObjectProperty `gn:parentADM2`. Let us also suppose that there exists a `TouristDestination` individual called `GreatLondonDestination` which includes (via the `tao:containsGeo` property) Greater London. Then, the reasoner should infer that `Hotel Splendor` is also part of `GreatLondonDestination`.

As before, we modelled 10 test cases as OWL files using the TestCase OWL meta-model. These files are identified by a unique IRI and contain only the ABox, relying[77] on the TBox of the TAO ontology and the TestCase metamodel[78]. The ABox contains a set of individuals necessary to execute the test and obtain the expected results.

We loaded the test files in Protégé and run the Pellet reasoner[79].

All 10 test cases yielded the expected results.

These tests are useful to understand if the ontology can be successfully used to extend the knowledge graph with reasoning e.g., using inverse properties definitions to materialize backlinks[80], using a chain of object properties to infer new relationships[81], inferring the type of an entity from its properties[82].

It is worth noting that the creation of inference verification tests has been used during the ontology engineering process for guiding the introduction and refinement of new axioms in TAO.

All inference verification test cases and the related data sets are available at https://github.com/linkalab/tkg/tree/main/validation/inference_verification.

4.2.2. Error provocation

This test aims at understanding how the knowledge graph (TKG) reacts to the injection of inconsistent data. As an example, since an entity cannot be at the same time an `tao:Hotel` and a `tao:BedAndBreakfast`, we can validate the ontology with regards to this requirement by injecting an individual which is defined as belonging to both classes. The test is successful if the reasoner finds an inconsistency because the appropriate disjointedness axiom is defined in the ontology.

We followed the same strategy used in the inference verification tests described above. In addition, for some tests we developed also a SHACL file defining further constraints[83].

We implemented 8 test cases for error provocation, testing the identification of wrong patterns in the knowledge graph such as the inclusion of hotel rooms as accommodations in a lodging facility that is not a hotel, the inclusion of accommodation to multiple disjoint lodging facilities, the presence of isolated nodes like a location amenity not connected to any accommodation or lodging facility[84].

Finally, we loaded the test file within Protégé, and then we ran both reasoner and the SHACL rules engine[85]. A test is successful if the injected inconsistencies are detected by the reasoner and/or the SHACL validator.

We used this same error provocation technique to test the correct creation of triples during the triple creation process (see section 3.5) and to refine axioms and constraints in TAO.

All error provocation tests cases and the related data sets are available at https://github.com/linkalab/tkg/tree/main/validation/error_provocation.

---

[76] See Greater London http://www.geonames.org/2648110/greater-london.html

[77] Using `owl:imports`.

[78] http://www.ontologydesignpatterns.org/schemas/testannotationschema.owl

[79] We used the Pellet reasoner, see the Protégé plug-in https://github.com/stardog-union/pellet/tree/master/protege/plugin

[80] As an example if an Accommodation is `tao:partOf` a lodging facility the inverse relation `tao:includes` can be added to the knowledge graph.

[81] A TouristDestination can be expressed as the composition of other geographic features (using `gn:parentFeature`) so that all lodging facilities contained in those features become also part of the TouristDestination itself.

[82] A lodging facility can be inferred to be of type LowRatedFacility if its normalised rating value is less or equal than 0.6.

[83] In some test we use SHACL language to test for integrity constraints that are not limited by the Open World Assumption (OWA)

[84] This case requires the use of SHACL rules because of the open world assumption in OWL.

[85] Using SHACL4Protege Constraint Validator, see https://github.com/fekaputra/shacl-plugin

4.3. Structural dimension

We assessed the structural dimension of TAO and TKG by computing different metrics for assessing ontologies and KG that have been defined and used in the literature [6, 20]. In particular, we followed a similar approach to Carriero et al. [6], which considered both base and topological metrics. Base metrics are used to assess the following quantitative aspects:

- *number of axioms* - the total number of axioms defined for classes, properties, datatype definitions, assertions, and annotations;
- *number of logical axioms* - the number of axioms that affect the logical meaning of an ontology;
- *number of classes* - the total number of classes defined in the ontology;
- *number of object properties* - the total number of object properties defined in the ontology;
- *number of datatype properties* - the total number of datatype properties defined in the ontology;
- *number of annotation assertions* - the total number of annotations in the ontology;
- *DL expressivity* - the description logic expressivity of the ontology.

On the other hand, topological metrics are useful to understand ontology richness, width/depth, inheritance structure, cohesion, and multi-hierarchical degree.

In particular, we adopted the following metrics:

- *Inheritance Richness (IR)* - measures the average number of sub-classes per class[86]. Low values indicate a vertical (deep) ontology whereas high values indicate a horizontal (shallow) ontology.
- *Relationship Richness* - measures the ratio of the number of non-inheritance relationships divided by the number of relationships of all kinds[87]. Values are normalised to one, where 0 indicates that only inheritance relations exist in the ontology and 1 that no inheritance relations are present.
- *Axiom Class Ratio* - measures the ratio of the number of axioms divided by the number of classes[88]. A scarcely axiomatised ontology has a low value of this metric (near zero); higher values are an indication of a better axiomatisation, but very high values can state an excessive axiomatisation.
- *Class/property ratio* - measures the ratio of the number of classes divided by the number of relations[89]. Low values (i.e., ~ 0) are found in ontologies with many properties connecting a few concepts. On the contrary, high values indicate that the ontology has many classes connected by few properties.
- *NoR* - number of root classes (a class which is not a subclass of other classes)[90]. The interpretation of NoR depends on the total number of classes. We expose (i) the ordinal values of NoR and (ii) the ratios between NoR and the number of classes between parenthesis.
- *NoL* - number of leaf classes (all classes that have no sub-classes)[91]. The interpretation of NoL depends on the total number of classes. We expose (i) the ordinal values of NoL and (ii) the ratios between NoL and the number of classes between parenthesis.
- *NoC* - number of external classes[92] defined by [36]. A low value of NoC can indicate that the ontology is semantically independent; a high value can indicate that the ontology depends on concepts defined in other ontologies. The interpretation of NoC depends also on the number of classes in ontology. We expose (i) the ordinal values of NoC and (ii) the ratios between NoC and the number of classes between parenthesis.
- *ADIT-LN* (Average depth of inheritance tree of leaf nodes) - is the average depth of the graph constructed considering classes as nodes and `subClassOf` properties as arcs [93].

[86] See https://ontometrics.informatik.uni-rostock.de/wiki/index.php/Schema_Metrics#Inheritance_Richness
[87] See https://ontometrics.informatik.uni-rostock.de/wiki/index.php/Schema_Metrics#Relationship_Richness
[88] https://ontometrics.informatik.uni-rostock.de/wiki/index.php/Schema_Metrics#Axiom_Class_Ratio
[89] See https://ontometrics.informatik.uni-rostock.de/wiki/index.php/Schema_Metrics#Class_Relation_Ratio
[90] See https://ontometrics.informatik.uni-rostock.de/wiki/index.php/Knowledgebase_Metrics#Number_of_root_classes_.28NoR.29
[91] See https://ontometrics.informatik.uni-rostock.de/wiki/index.php/Knowledgebase_Metrics#Number_of_leaf_classes_.28NoL.29
[92] A class is considered external when it is defined in a different ontology. This metric has been calculated using Protégé.
[93] See https://ontometrics.informatik.uni-rostock.de/wiki/index.php/Knowledgebase_Metrics#Average_depth_of_inheritance_tree_of_leaf_nodes_.28ADIT-LN.29

- *Max breadth* - the maximal value of breadth computed on the graph constructed as for the *ADIT-LN* metric[94]. The value of *Max breadth* should be considered concerning the number of classes in ontology.
- *Average breadth* - the average breadth computed on the graph constructed as for the *ADIT-LN* metric[95].
- *Max depth* - the maximal depth obtained by traversing the graph constructed as for the *ADIT-LN* metric.[96] The value of *Max depth* should be considered concerning the number of classes in ontology.
- *Tangledness* - is the degree of multi-hierarchical classes (which are classes with more than one super-class). It is related to the multi-hierarchical nodes of the graph constructed for the *ADIT-LN* metric[97]. A value of 0 indicates no tangledness; a value of 1 indicates that each class has multiple super-classes.

Tables 1 and 2 report the base and topological metrics measured on TAO, Hontology, and the Accommodation Ontology (Acco). It should be noted that when analysing TAO we considered only the classes and properties defined in TAO and not the ones imported from other ontologies (i.e., the Accommodation Ontology, GoodRelations). This was done to allow a fair comparison with the Accommodation Ontology, which we extensively reuse.

All metrics are calculated using OntoMetrics[98] web tool which computes several statistics on ontologies.

Table 1
Base metrics.

metric name	tao	hontology	acco
# axioms	3853	1453	344
# logical axioms	1222	448	111
# classes	590	284	31
# object properties	16	8	21
# datatype properties	3	31	14
# annotation assertions	1982	682	161
DL expressivity	SROIQ(D)	ALCHQ(D)	ALUH(D)

Table 2
Topological metrics.

Metric name	TAO	Hontology	Acco
Inheritance Richness	1.173	0.961	0.742
Relationship Richness	0.412	0.321	0.477
Axiom Class Ratio	6.531	5.116	11.097
Class/propery ratio	0.502	0.706	0.705
NoR	15 (0.03)	17 (0.06)	13 (0.42)
NoL	496 (0.84)	247 (0.87)	23 (0.74)
NoC	19 (0.03)	0 (0.00)	2 (0.06)
ADIT-LN	3.913	2.725	2.439
Max depth	6	5	3
Average breadth	6.615	7.375	5.077
Max breadth	54	29	13
Tangledness	0.176	0.018	0.097

From Table 1 we can observe that TAO is significantly larger than Hontology and Accommodation Ontology in terms of number of classes, axioms, logical axioms[99], and annotation assertions. The additional classes mostly describe different types of lodging facilities (35 classes), accommodations (17 classes), amenities (343 classes), and tourist locations (146 classes).

In terms of properties, TAO introduces only a few of new ones, since it reuses most of them from Acco (4), GoodRelations (15), Schema.org (11) and GeoNames (1) as discussed in Section 3.3.3.

Finally, in terms of expressivity, TAO is similar to Hontology because they share ALCQU features and with Acco because they share ALCU features; TAO does not have H feature because it does not express role hierarchies

[94] See https://ontometrics.informatik.uni-rostock.de/wiki/index.php/Graph_Metrics#Maximal_breadth
[95] See https://ontometrics.informatik.uni-rostock.de/wiki/index.php/Graph_Metrics#Average_breadth
[96] See https://ontometrics.informatik.uni-rostock.de/wiki/index.php/Graph_Metrics#Maximal_depth
[97] See https://ontometrics.informatik.uni-rostock.de/wiki/index.php/Graph_Metrics#Tangledness
[98] See https://ontometrics.informatik.uni-rostock.de/ontologymetrics/index.jsp
[99] Logical axioms affect the logical meaning of an ontology. See https://ontometrics.informatik.uni-rostock.de/wiki/index.php/Base_Metrics#Logical_Axiom. On the other hand, non-logical axioms, like entity declarations or annotations, do not affect the consequences of an OWL 2 ontology. See https://www.w3.org/TR/owl2-syntax/#Entity_Declarations_and_Typing

(SubPropertyOf) as Hontology and Acco; to conclude TAO has the IS features, which indicate the presence of inverse and transitive roles (relations), that the other two ontologies do not have.

Several indicators in Tables 1 and 2 suggest that TAO offers a very good *transparency*, *flexibility*, and *cognitive ergonomics* [20] in comparison with Hontology and the Accommodation Ontology. *Transparency* has been defined as "the property of an ontology to be analysed in detail, with a rich formalisation of conceptual choices and motivation". *Flexibility* is related to how easy is to change and evolve the ontology with limited side-effects. Finally, *cognitive ergonomics* is the ability of an ontology to be "easily understood, manipulated, and exploited by final users". In the following we discuss the main indicators of these properties.

The indicators of transparency include:

- a relative *high number of axioms per class* (6.531). This is higher than Hontology, but lower than the Accommodation Ontology, mostly due to the much lower number of classes in the latter;
- a *small coupling* with external ontologies (0.03), similarly to Hontology (0) and the Accommodation Ontology (0.06). This is computed as the number of external classes defined in other ontologies (NoC) normalized by the total number of classes. Low coupling allows users to more easily inspect and understand an ontology.
- a *strong cohesion* (i.e., relatedness among classes) due to the low depth of the class hierarchy (ADIT-LN = 3.913), the small number of root classes (NoR = 15), and the high number of leaf classes (NoL = 496);
- a *high inheritance richness* (1.173), which accounts for a more vertical structure, reflecting a more comprehensive coverage of the tourism domain. This is higher than both Hontology (0.961) and the Accommodation Ontology (0.742).

The combination of *low coupling* and *strong cohesion* are also indicators of *flexibility* [20].

Finally, the indicators of *cognitive ergonomics* are the following:

- a relatively *low class/property ratio* (0.502), also smaller than Hontology (0.706) and Accommodation Ontology (0.705);
- a *sub-class tree with low depth and breadth* as indicated by ADIT-LN (3.913), max depth (6), and average breadth (6.615);
- a relatively *low tangledness* (0.176 in a range from 0 to 1) that suggests that the inheritance tree has a low complexity.

Table 3 reports some statistics about the current prototype of TKG, which includes over 10M triples describing about 35K facilities and almost 898K reviews.

Figure 8 shows the distribution of individuals in terms of classes. The most frequent classes are (i) `tao:NormRating` and `schema:UserReview` which are used for reviews; (ii) `acco:AccommodationFeature`[100] that is used as a generic class for amenities together with a specific class from `tao` (e.g., `tao:Kitchen`, `tao:Television`); (iii) the classes used to model an offer such as `gr:Offering`, `gr:TypeAndQuantityNode`, and `gr:UnitPriceSpecification`; (iv) `tao:Accommodation`, `gr:QuantitativeValue`, `gr:SomeItems`, and `acco:BedDetails` are the classes used to model an accommodation; (v) `tao:LodgingDescription`, `tao:LodgingFacility` (and its subclasses), `schema:PostalAdress`, and `tao:NormAggregateRating` that are used to model the lodging facilities. The other classes in the diagram are sub-classes of `tao:LocationAmenity`, `tao:Accommodation` or `tao:LodgingFacility`, which are used to specify precisely their type.

5. Conclusions

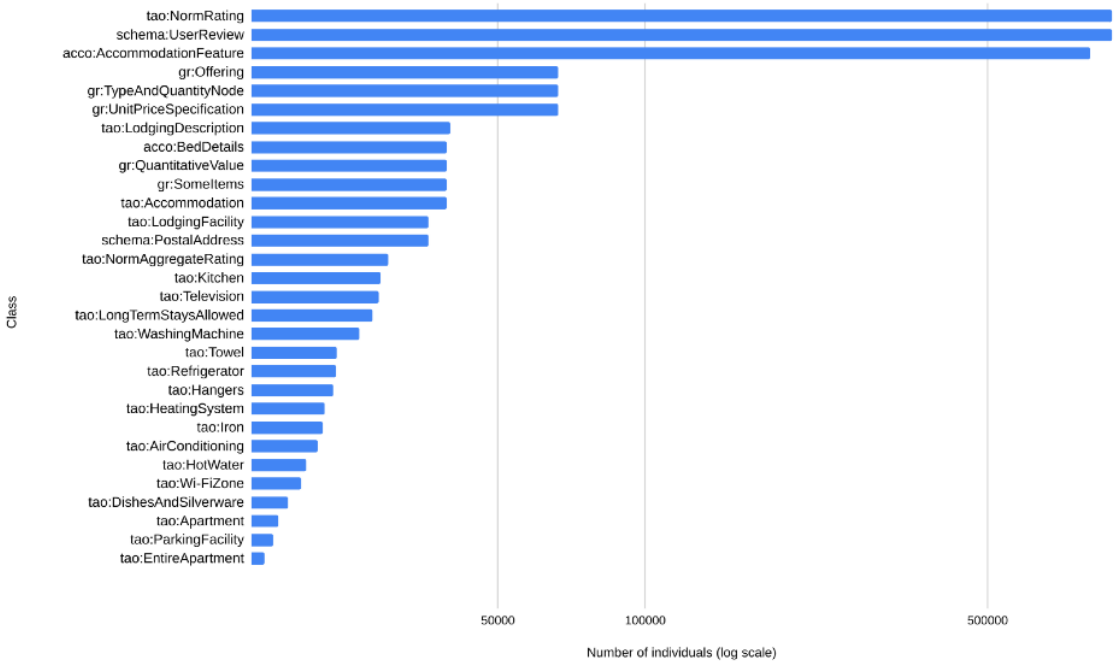
In this paper, we presented a framework for the semi-automatic construction of a Tourism Knowledge Graph (TKG) and introduced a novel ontology for modelling this domain: the Tourism Analytics Ontology (TAO). We have evaluated TKG and TAO according to functional, logical, and structural dimensions.

[100] `tao:LocationAmenity` is defined as an equivalent class to `acco:AccommodationFeature`.

Table 3
Knowledge graph metrics

Metric	Value
Number of triples	10,917,081
Number of distinct relations	146
Number of links to DBPedia entities	210,245
Number of unique DBpedia enities linked	3,851
Number of links to GeoNames entities	142,043
Number of unique GeoNames enities linked	3,487
Number of AirBnB reviews entities	358,005
Number of Booking.com reviews entities	539,834
Number of AirBnB LodgingFacility entities	29,870
Number of Booking.com LodgingFacility entities	6,126

Fig. 8. Top 30 classes by number of individuals in the knowledge graph



The evaluation suggests that TAO is 1) larger than the alternatives (Hontology and the Accommodation Ontology) in terms of the number of classes and axioms and 2) also offers higher transparency, flexibility, and cognitive ergonomics.

In future work, we aim to pursue three main pathways. First, we are working on developing NLP solutions to improve the extraction of entities from text, such as descriptions and reviews, so to further enrich the representation of accommodation facilities. This step includes the extraction of data from other sources related to several other touristic destinations. Second, we want to develop a more scalable solution for integrating data about millions of facilities and users. Third, we want to develop a range of intelligent services based on TKG, including an entity linking application for automatically annotating accommodations according to reviews and a conversational agent

able to answer questions regarding the tourism sector. Transversally to them, we are working on automatising as much as possible the pipeline we have used intending to create knowledge graphs with related ontologies in any domain and sources.

References

[1] R. Alonso-Maturana, E. Alvarado-Cortes, S. López-Sola, M. O. Martínez-Losa, and P. Hermoso-González. La Rioja turismo: The construction and exploitation of a queryable tourism knowledge graph. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11153 LNCS:213–220, 2018. ISSN 16113349. .

[2] M. Atzeni and D. R. Recupero. Multi-domain sentiment analysis with mimicked and polarized word embeddings for human-robot interaction. *Future Gener. Comput. Syst.*, 110:984–999, 2020. . URL https://doi.org/10.1016/j.future.2019.10.012.

[3] R. Barta, C. Feilmayr, B. Pröll, C. Grün, and H. Werthner. Covering the semantic space of tourism : An approach based on modularized ontologies. *ACM International Conference Proceeding Series*, 2009. .

[4] E. Blomqvist, A. Seil Sepour, and V. Presutti. Ontology testing – Methodology and tool. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7603 LNAI(November 2020):216–226, 2012. ISSN 03029743. .

[5] P. Calleja, F. Priyatna, N. Mihindukulasooriya, and M. Rico. DBtravel: A tourism-oriented semantic graph. In C. Pautasso, F. Sánchez-Figueroa, K. Systä, and J. M. Murillo Rodríguez, editors, *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 11153 LNCS of *Lecture Notes in Computer Science*, pages 206–212, Cham, 2018. Springer International Publishing. ISBN 9783030030551. . URL http://link.springer.com/10.1007/978-3-030-03056-8.

[6] V. A. Carriero, A. Gangemi, M. L. Mancinelli, A. G. Nuzzolese, V. Presutti, and C. Veninata. Pattern-based design applied to cultural heritage knowledge graphs. *Semantic Web*, 12(2):313–357, 2021. ISSN 22104968. .

[7] M. S. Chaves and C. Trojahn. Towards a multilingual ontology for ontology-driven content mining in Social Web sites. *CEUR Workshop Proceedings*, 687, 2010. ISSN 16130073.

[8] M. S. Chaves, L. Freitas, and R. Vieira. Hontology: A multilingual ontology for the accommodation sector in the tourism industry. *KEOD 2012 - Proceedings of the International Conference on Knowledge Engineering and Ontology Development*, pages 149–154, 2012. .

[9] S. Consoli, A. Gangemi, A. G. Nuzzolese, S. Peroni, V. Presutti, D. R. Recupero, and D. Spampinato. Towards emergency vehicle routing using geolinked open data: the case study of the municipality of catania. In A. Gangemi, H. Alani, M. Nissim, E. Cambria, D. R. Recupero, V. Lanfranchi, and T. Kauppinen, editors, *Joint Proceedings of the 1th Workshop on Semantic Sentiment Analysis (SSA2014), and the Workshop on Social Media and Linked Data for Emergency Response (SMILE 2014) co-located with 11th European Semantic Web Conference (ESWC 2014), Crete, Greece, May 25th, 2014*, volume 1329 of *CEUR Workshop Proceedings*, pages 31–42. CEUR-WS.org, 2014. URL http://ceur-ws.org/Vol-1329/preface-SM.pdf.

[10] S. Consoli, A. Gangemi, A. G. Nuzzolese, S. Peroni, V. Presutti, D. R. Recupero, and D. Spampinato. Geolinked open data for the municipality of catania. In R. Akerkar, N. Bassiliades, J. Davies, and V. Ermolayev, editors, *4th International Conference on Web Intelligence, Mining and Semantics (WIMS 14), WIMS '14, Thessaloniki, Greece, June 2-4, 2014*, pages 58:1–58:8. ACM, 2014. . URL https://doi.org/10.1145/2611040.2611092.

[11] S. Consoli, A. Gangemi, A. G. Nuzzolese, S. Peroni, D. R. Recupero, and D. Spampinato. Setting the course of emergency vehicle routing using geolinked open data for the municipality of catania. In V. Presutti, E. Blomqvist, R. Troncy, H. Sack, I. Papadakis, and A. Tordai, editors, *The Semantic Web: ESWC 2014 Satellite Events - ESWC 2014 Satellite Events, Anissaras, Crete, Greece, May 25-29, 2014, Revised Selected Papers*, volume 8798 of *Lecture Notes in Computer Science*, pages 42–53. Springer, 2014. . URL https://doi.org/10.1007/978-3-319-11955-7_4.

[12] S. Consoli, V. Presutti, D. R. Recupero, A. G. Nuzzolese, S. Peroni, M. Mongiovì, and A. Gangemi. Producing linked data for smart cities: The case of catania. *Big Data Res.*, 7:1–15, 2017. . URL https://doi.org/10.1016/j.bdr.2016.10.001.

[13] J. Daiber, M. Jakob, C. Hokamp, and P. N. Mendes. Improving efficiency and accuracy in multilingual entity extraction. *ACM International Conference Proceeding Series*, pages 121–124, 2013. .

[14] A. Dimou, M. V. Sande, P. Colpaert, R. Verborgh, E. Mannens, and R. Van De Walle. RML: A generic language for integrated RDF mappings of heterogeneous data. In *CEUR Workshop Proceedings*, volume 1184, 2014.

[15] A. Dimou, T. De Nies, R. Verborgh, E. Mannens, and R. de Walle. Automated Metadata Generation for Linked Data Generation and Publishing Workflows. *Proceedings of the 9th Workshop on Linked Data on the Web*, 1593, 2016. ISSN 1613-0073.

[16] A. Dridi and D. R. Recupero. Leveraging semantics for sentiment polarity detection in social media. *Int. J. Mach. Learn. Cybern.*, 10(8): 2045–2055, 2019. . URL https://doi.org/10.1007/s13042-017-0727-z.

[17] F. Erxleben, M. Günther, M. Krötzsch, J. Mendez, and D. Vrandečić. Introducing wikidata to the linked data web. In P. Mika, T. Tudorache, A. Bernstein, C. Welty, C. Knoblock, D. Vrandečić, P. Groth, N. Noy, K. Janowicz, and C. Goble, editors, *The Semantic Web – ISWC 2014*, pages 50–65, Cham, 2014. Springer International Publishing. ISBN 978-3-319-11964-9.

[18] D. Fensel, U. Şimşek, K. Angele, E. Huaman, E. Kärle, O. Panasiuk, I. Toma, J. Umbrich, and A. Wahler. Knowledge Graphs. *Knowledge Graphs*, mar 2020. . URL http://arxiv.org/abs/2003.02320.

[19] O. Fodor and H. Werthner. Harmonise: A step toward an interoperable e-tourism marketplace. *International Journal of Electronic Commerce*, 9(2):11–39, 2005. . URL https://doi.org/10.1080/10864415.2005.11044324.

[20] A. Gangemi, C. Catenacci, M. Ciaramita, and J. Lehmann. Modelling Ontology Evaluation and Valilidation - in Proceedings of ESWC2006. *Eswc2006*, page 15, 2006.

[21] D. Gazzè, A. L. Duca, A. Marchetti, and M. Tesconi. An overview of the tourpedia linked dataset with a focus on relations discovery among places. *ACM International Conference Proceeding Series*, 16-17-Sept:157–160, 2015. .

[22] M. Grüninger, M. S. Fox, and M. Gruninger. Methodology for the design and evaluation of ontologies. In *International Joint Conference on Artificial Inteligence (IJCAI95), Workshop on Basic Ontological Issues in Knowledge Sharing*, pages 1–10, 1995. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.44.8723.

[23] R. V. Guha, D. Brickley, and S. Macbeth. Schema.org: Evolution of Structured Data on the Web. *Communications of the ACM*, 59(2): 44–51, jan 2016. ISSN 0001-0782. . URL https://dl.acm.org/doi/10.1145/2844544.

[24] A. Halterman. Mordecai: Full Text Geoparsing and Event Geocoding. *The Journal of Open Source Software*, 2(9):91, 2017. ISSN 2475-9066. .

[25] M. Hepp. GoodRelations: An ontology for describing products and services offers on the web. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 5268 LNAI, pages 329–346, 2008. ISBN 3540876952. .

[26] P. Heyvaert, B. De Meester, A. Dimou, and R. Verborgh. Declarative rules for linked data generation at your fingertips! *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11155 LNCS(August): 213–217, 2018. ISSN 16113349. .

[27] E. Kärle, U. Simsek, Z. Akbar, M. Hepp, and D. Fensel. Extending the schema.org vocabulary for more expressive accommodation annotations. In R. Schegg and B. Stangl, editors, *Information and Communication Technologies in Tourism 2017*, pages 31–41, Cham, 2017. Springer International Publishing. ISBN 978-3-319-51168-9.

[28] E. Kärle, U. Şimşek, O. Panasiuk, and D. Fensel. Building an ecosystem for the tyrolean tourism knowledge graph. *arXiv*, pages 260–267, 2018. ISSN 23318422. .

[29] R. L R and T. Bomatpalli. A survey of travel recommender system. *International Journal of Computer Sciences and Engineering*, 7(3): 356–362, 2019. URL https://doi.org/10.26438/ijcse/v7i3.356362.

[30] J. B. Lamy. Owlready: Ontology-oriented programming in Python with automatic classification and high level constructs for biomedical ontologies. *Artificial Intelligence in Medicine*, 80:11–28, 2017. ISSN 18732860. .

[31] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. Van Kleef, S. Auer, and C. Bizer. DBpedia - A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web*, 6(2):167–195, 2015. ISSN 22104968. .

[32] J. L. Martinez-Rodriguez, A. Hogan, and I. Lopez-Arevalo. Information extraction meets the semantic web: A survey. *Semantic Web Journal*, 11:255–335, 2020. .

[33] P. N. Mendes, M. Jakob, A. García-Silva, and C. Bizer. DBpedia spotlight: Shedding Light on the Web of Documents. In *Proceedings of the 7th International Conference on Semantic Systems - I-Semantics '11*, pages 1–8, New York, New York, USA, 2011. ACM Press. ISBN 9781450306218. . URL http://dl.acm.org/citation.cfm?doid=2063518.2063519.

[34] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013. URL http://dblp.uni-trier.de/db/journals/corr/corr1301.html#abs-1301-3781.

[35] N. F. Noy and D. L. McGuinness. Ontology Development 101: A Guide to Creating Your First Ontology. *Stanford Knowledge Systems Laboratory*, page 25, 2001. ISSN 09333657. .

[36] A. Orme, H. Tao, and L. Etzkorn. Coupling metrics for ontology-based system. *IEEE Software*, 23(2):102–108, mar 2006. ISSN 0740-7459. . URL http://ieeexplore.ieee.org/document/1605186/.

[37] S. Ou, V. Pekar, C. Orasan, C. Spurk, and M. Negri. Development and alignment of a domain-specific ontology for question answering. *Proceedings of the 6th International Conference on Language Resources and Evaluation, LREC 2008*, pages 2221–2228, 2008.

[38] N. Shuyo. Language detection library for java, 2010. URL http://code.google.com/p/language-detection/.

[39] S. Staab, C. Braun, I. Bruder, A. Düsterhöft, A. Heuer, M. Klettke, G. Neumann, B. Prager, J. Pretzel, H. P. Schnurr, R. Studer, H. Uszkoreit, and B. Wrenger. GETESS—searching the web exploiting German Texts. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 1652, pages 113–124, 1999. ISBN 3540663258. . URL http://link.springer.com/10.1007/3-540-48414-0_7.

[40] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: A core of semantic knowledge. In *16th International World Wide Web Conference, WWW2007*, pages 697–706, 2007. ISBN 1595936548. .

[41] M. Tenemaza, J. Limaico, and S. Luján-Mora. Tourism recommender system based on natural language classifier. In T. Z. Ahram, W. Karwowski, and J. Kalra, editors, *Advances in Artificial Intelligence, Software and Systems Engineering*, pages 230–235, Cham, 2021. Springer International Publishing. ISBN 978-3-030-80624-8.

[42] R. Troncy, G. Rizzo, A. Jameson, O. Corcho, J. Plu, E. Palumbo, J. C. Ballesteros Hermida, A. Spirescu, K. D. Kuhn, C. Barbu, M. Rossi, I. Celino, R. Agarwal, C. Scanu, M. Valla, and T. Haaker. 3cixty: Building comprehensive knowledge bases for city exploration. *Journal of Web Semantics*, 46-47:2–13, 2017. ISSN 15708268. . URL http://dx.doi.org/10.1016/j.websem.2017.07.002.

[43] R. Verborgh, O. Hartig, B. De Meester, G. Haesendonck, L. De Vocht, M. Vander Sande, R. Cyganiak, P. Colpaert, E. Mannens, and R. Van de Walle. Querying datasets on the web with high availability. In P. Mika, T. Tudorache, A. Bernstein, C. Welty, C. Knoblock, D. Vrandečić, P. Groth, N. Noy, K. Janowicz, and C. Goble, editors, *The Semantic Web – ISWC 2014*, pages 180–196, Cham, 2014. Springer International Publishing. ISBN 978-3-319-11964-9.

[44] R. Verborgh, M. V. Sande, P. Colpaert, S. Coppens, E. Mannens, and R. Van De Walle. Web-scale querying through linked data fragments. In *CEUR Workshop Proceedings*, volume 1184, 2014.

[45] D. Xiao, N. Wang, J. Yu, C. Zhang, and J. Wu. A Practice of Tourism Knowledge Graph Construction Based on Heterogeneous Information. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 12522 LNAI(c):159–173, 2020. ISSN 16113349. .

[46] W. Zhang, H. Cao, F. Hao, L. Yang, M. Ahmad, and Y. Li. The Chinese Knowledge Graph on Domain-Tourism. *Lecture Notes in Electrical Engineering*, 590(November):20–27, 2020. ISSN 18761119. .