Neural Networks forBuilding Semantic Models and Knowledge Graphs

Doctoral Dissertation
Doctoral Program in Computer and Control Engineering (32.th cycle)

Neural Networks for Building Semantic Models and Knowledge Graphs

Giuseppe Futia
* * * * * *

Supervisors
Prof. Juan Carlos De Martin, Supervisor
Dr. Antonio Vetro', Advisor

Doctoral Examination Committee:
Dr. Francesco Osborne (Referee) Knowldege Media Institute, The Open University
Prof. Matteo Luigi Palmonari (Referee) Università degli Studi di Milano-Bicocca
Prof. Fulvio Corno, Politecnico di Torino
Dr. Anastadia Dimou, RUG - Universiteit Gent
Dr. Federico Morando, Synapta SRL

Politecnico di Torino
2020

I hereby declare that, the contents and organisation of this dissertation constitute my own original work and does not compromise in any way the rights of third parties, including those relating to the security of personal data.

. . . . . .

Giuseppe Futia
Turin, 2020

Summary

Knowledge Graphs (KGs) have emerged as a core abstraction for incorporating human knowledge into intelligent systems. This knowledge is encoded in a graph-based structure whose nodes represent real-world entities, while edges define meaningful and binary relations between these entities. KGs are gaining attention from both the industry and academia because they provide a flexible way to capture, organize, and query a large amount of multi-relational data.

The structured knowledge that shapes KGs can be composed of simple statements, such as "Socrates is a human", or quantified statements, such as "All humans are mortal". Simple statements represent a collection of *facts* that are arranged as edges within the KGs, while quantified statements require a more advanced and expressive form to represent information. Ontologies define a standard formalism that enables this meaningful representation, specifying the semantics of entities and relations adopted to label nodes and edges of KGs.

Deductive and inductive reasoning approaches allow extending a KG, improving its underlying knowledge. Deductive methods employ the simple statements and the set of rules defined by ontologies to derive additional knowledge. For instance 'Socrates is mortal", which is logically interpretable and understandable, combining "Socrates is a human" and "All humans are mortal". Inductive techniques involve simple and quantified statements to create further knowledge and discover and generalize patterns available in the KG. These patterns can be inferred by applying statistical learning methods on multi-relational data, which are less interpretable than deductive approaches. However, they are capable of exploiting latent factors in the KG that are not directly extracted as quantities in the data, but whose variations influence every single piece of information we are able to observe.

The most recent implementation of the statistical learning methods includes representation learning techniques, based on deep architectures of Neural Networks (NNs). These architectures contributed to reaching unprecedented results in the prediction and classification tasks of modern Artificial Intelligence (AI) systems. More specifically, at the time of writing this thesis, NNs natively-built for graph structures, the so-called Graph Neural Networks (GNNs), are gaining momentum and empowering the cutting-edge research on graph data.

The thesis's primary goal is to investigate the role of NN architectures, particularly the GNNs, to support the publication of KGs. More precisely, this thesis intends to address two main open problems in the KGs research field: (i) the mapping of data source schemas to reference ontologies, considering a semantic modeling perspective. This task represents a key factor for materializing original data as KG statements or virtualizing the access to the source as a KG; (ii) the refinement of existing KGs by inferencing soft but consistent knowledge in terms of new edges (or links). Such new edges are hard to encode into deductive and logic-based reasoning, but they are beneficial to develop tools on top of KGs, e.g., recommendation systems. Furthermore, this thesis reports the results within two different application domains: public procurement and academic publications.

This thesis illustrates novel contributions to the automatic semantic modeling with NN architectures in regards to the first open problem. An initial study is conducted applying a simple, but efficient neural language model, such as Word2Vec, on SPARQL queries performed on different KGs. However, this approach does not take full advantage of the graph structure for the learning process: SPARQL queries include a limited number of graph patterns, and Word2Vec treats these patterns as plain text. Therefore, a more in-depth investigation is conducted, developing a tool called SeMi (SEmantic Modeling machIne), which employs a novel method based on GNNs combined with a scoring function, trained on available multi-relational data repositories. This approach aims to produce a latent representation of the entities and the relations between these entities, from the graph structure of the multi-relation data adopted as a training set. This investigation shows that the adoption of these latent representations increases the accuracy of the computed semantic models, compared to manually-selected features. SeMi has been adopted in a real scenario to support the building of a novel KG in the public procurement domain.

In regards to the second open problem, the thesis reports an approach based on GNNs to predict new edges within a novel KG developed in academic publications. In this context, the thesis presents Geranium, a semantic platform for collecting and organizing the scientific knowledge of the Politecnico di Torino (Polito). The research achievements obtained with Geranium are the following: (i) an academic KG that semantically connects information on researchers and publications of Polito; (ii) a semantic search engine that aggregates such information and enables advanced features for the content exploration; (iii) a recommendation system which exploits a link prediction mechanism to suggest, for instance, novel collaboration opportunities between researchers of different disciplines, who worked on the same topics.

Inductive techniques exploiting neural architectures do not provide human-understandable insights on how a specific result was achieved. Furthermore, the application domains such as those analyzed in this thesis — public procurement and academic publications — are contexts where the impact of NNs is relevant:

the interpretability of the results is not only a desirable property, but it is a fundamental requirement for the involved stakeholders. Nevertheless, most of the available approaches to implement an eXplainable Artificial Intelligence (XAI) focus on technical solutions usable only by experts able to understand and manipulate the computational architectures of NNs. A complementary approach could incorporate deductive methods, which can exploit the symbolic representation of KG for inference new logic-based knowledge. The final part of this thesis presents new research trajectories in the field, proposing neural-symbolic integration as a cornerstone to design an AI which is closer to non-insiders comprehension. Within such a general direction, the thesis proposes three specific challenges for future research—knowledge matching, cross-disciplinary explanations, and interactive explanations.

Acknowledgements

The author wishes to thank his supervisor, Prof. Juan Carlos De Martin, who gave him the opportunity to do the PhD, in a place where critical thinking is continuously stimulated. The author thanks his advisor, Dr. Antonio Vetrò, for his suggestions and help in all the three years of the PhD. Finally, special thanks go to all the colleagues at the Nexa Center for their support and affection. They are (in random order): Selina Fenoglietto, Marco Conoscenti, Giovanni Garifo, Pasquale Pellegrino, Elena Beretta, Francesco Ruggiero, Mattia Plazio, Antonio Santangelo.

To my loving family. To Debora.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Knowledge Graphs (KGs) [79] have emerged as a core abstraction for incorporating human knowledge into intelligent systems. This knowledge is encoded in a graph-based structure whose nodes represent real-world entities, while edges define meaningful and binary relations between these entities. KGs are gaining attention from both the industry [116] and the academia [53], because they provide a flexible way to capture, organize, and query a large amount of multi-relational data.

The structured knowledge that shapes KGs can be composed of simple statements, such as "Socrates is a human", or quantified statements, such as "All humans are mortal". Simple statements represent a collection of *facts* that are arranged as edges within the KGs, while quantified statements require a more advanced and expressive form to represent information. Ontologies define a standard formalism that enables this meaningful representation, specifying the semantics of entities and relations adopted to label nodes and edges of KGs.

Deductive and inductive reasoning approaches allow extending a KG, improving its underlying knowledge. Deductive methods employ the simple statements and the set of rules defined by ontologies to derive additional knowledge. For instance 'Socrates is mortal", which is logically interpretable and understandable, combining "Socrates is a human" and "All humans are mortal". Inductive techniques involve simple and quantified statements to create further knowledge and discover and generalize patterns available in the KG. These patterns can be inferred by the application of statistical learning methods on multi-relational data, which are less interpretable than deductive approaches. However, they are capable of exploiting latent factors in the KG that are not directly extracted as quantities in the data, but whose variations influence every single piece of information we are able to observe.

The most recent implementation of the statistical learning methods includes representation learning techniques, based on deep architectures of Neural Networks (NNs) [63]. These architectures contributed to reaching unprecedented results in the prediction and classification tasks of modern Artificial Intelligence (AI) systems. More specifically, at the time of writing this thesis, NNs natively-built for

graph structures, the so-called Graph Neural Networks (GNNs) [48], are gaining momentum and empowering the cutting-edge research on graph data.

The thesis's primary goal is to investigate the role of NN architectures, particularly the GNNs, to support the publication of KGs. More precisely, this thesis intends to address open research problems (RP) in the KG research field: (**RP1**) the automatic mapping of data source schemas to reference ontologies, considering a semantic modeling perspective. This task represents a key factor for materializing original data as KG statements or virtualizing the access to the source as a KG [149] [160] [176] [21]; (**RP2**) the automatic refinement of existing KGs by inferencing soft, but consistent knowledge in terms of new edges (or links) [119] [138]. Such new edges are hard to encode into deductive and logic-based reasoning, but they are beneficial to develop tools on top of KGs, e.g., recommendation systems. Furthermore, this thesis reports the results within two different application domains: public procurement and academic publications.

1.1 Research Problems and Questions

RP1 - Mapping data source schemas to reference ontologies is a tedious task requiring a significant manual effort and domain knowledge expertise, due to the potential variety of data available on the Web and private data repositories. For this reason, automatic techniques are fundamental to scale the semantic mapping process. This thesis proposes a semantic model perspective on the mapping problem, which requires two steps: (i) the Semantic Type Detection (STD) or semantic labeling, whose goal is to annotate the attributes of the source; (ii) the Semantic Relation Inference (SRI), whose goal is to reconstruct the intended meaning of the data source, predicting the connections between these annotated attributes. The thesis focuses on the automation of the SRI task, and the first research question addressed in this work is the following:

- **RQ1** - Which is the contribution of NN to improve the accuracy of automatically-inferred semantic relations between the source attributes?

RP2 - Predicting new statements in the form of new links within an existing KG is also a well-known problem in literature [119] [138]. Most of the current methods for link prediction employ scoring functions to measure the plausibility of the KG statements. The goal of these methods is to learn the latent representation of entities and relations - also known as *embeddings* - to ensure that the true statements obtain a high score, while false statements obtain a low score. However, the inference of the correct statements or, from another perspective, the reconstruction of the correct edges in the KG, does not take full advantage of the graph structure, which is not directly encoded in the scoring function. Nevertheless, recent developments on GNN architectures are focusing on different approaches to produce the

latent representation of nodes, which embed the local graph structure, including the neighborhood features. Therefore, the second research question addressed in this thesis is the following:

- **RQ2** - Which is the impact of GNNs in the prediction of new edges within a KG, in the context of recommendation systems?

1.2 Research Contributions

For answering **RQ1**, the thesis illustrates novel contributions to the SRI task with NN architectures. An initial study [59] is conducted applying a simple, but efficient neural language model, such as Word2Vec [108], on SPARQL queries performed on different KGs. The goal of this study is to learn the latent representations of SPARQL variables, which are included in specific triple patterns within the query. Variables with a similar latent representation are then labeled with the most common relations available in the set of the triple patterns. Then, the syntactic closeness between such labeled variables and the attributes is exploited to create a mapping between these elements. Consequently, the correct semantic relations between the data source attributes is assigned. The main limitation is that such approach does not take full advantage of the graph structure for the learning process: SPARQL queries include a limited number of graph patterns and Word2Vec treats these patterns as plain text. Considering these limits, a deeper investigation is conducted, developing a tool called SeMi (SEmantic Modeling machIne) [57], which employs a novel method based on GNNs combined with a scoring function, trained on available multi-relational data repositories. The goal of this approach is to produce a latent representation of the entities and the relations between these entities, from the graph structure of the multi-relation data adopted as training set. The results of this investigation show that the adoption of these latent representations increases the accuracy of the SRI within a data source, compared to manually-selected features [157]. SeMi has been adopted in a real scenario to support the building of a novel KG in the public procurement domain. This KG allows to overcome the existing fragmentation of public procurement data and it creates fruitful conditions for deeper data quality analysis. The achieved results [58] show that integrating contracts data into a KG enables to detect consistency issues within the information released by the Italian public administrations. The most relevant consistency issues are: (i) business entities with more than one business name; (ii) unique ids that identify more than one contract; (iii) incoherent payments among different versions of an ongoing contract.

For answering **RQ2**, the thesis reports an approach based on GNNs, for predicting new edges within a novel KG developed in the field of academic publications. In this context, the thesis presents Geranium, a semantic platform to collect and organize the scientific knowledge of the Politecnico di Torino (Polito). The research

achievements obtained with Geranium are the following: (i) an academic KG that semantically connects information on researchers and publications of Polito; (ii) a semantic search engine that aggregates such information and enables advanced features for the content exploration; (iii) a recommendation system which exploits a link prediction mechanism to suggest, for instance, novel collaboration opportunities between researchers of different disciplines, who worked on the same topics.

1.3 New Research Trajectories

Inductive techniques exploiting neural architectures do not provide human-understandable insights on how a specific result was achieved. Furthermore, the application domains such as those analyzed in this thesis — public procurement and academic publications — are contexts where the impact of NNs is relevant: the interpretability of the results is not only a desirable property, but it is a fundamental requirement for the involved stakeholders. Nevertheless, most of the available approaches to implement an eXplainable Artificial Intelligence (XAI) [141] focus on technical solutions usable only by experts able to understand and manipulate the computational architectures of NNs. A complementary approach could incorporate deductive methods, which can exploit the symbolic representation of KG for inference new logic-based knowledge. The final part of this thesis presents new research trajectories in the field, proposing neural-symbolic integration as a cornerstone to design an AI which is closer to non-insiders comprehension [56]. Within such a general direction, the thesis proposes three specific challenges for future research—knowledge matching, cross-disciplinary explanations, and interactive explanations.

1.4 Structure of the Thesis

The remainder of this work is organized as follows. Chapter 2 lays the foundations of the concepts addressed in the whole thesis. It includes the definitions of KGs and ontologies, it discusses the methodologies for enriching and refining KGs, and it explains the main principles behind NN architectures applied to KGs. Chapter 3 provides an overview of the recent work related to these open research problems. Chapter 4 describes context details on the application scenarios related to the public procurement and the academic publication domains. Chapter 5 depicts a novel approach for predicting semantic relations within a data source, based on the training of a neural language model with SPARQL queries. Chapter 6 presents an innovative method to automatically compute semantic relations between annotated attributes, which are predicted by exploiting GNNs. Moreover, this chapter discusses the generation of a novel KG in the public procurement scenario. Chapter 7 reports details on the adoption of the GNNs for the automatic

inference of new knowledge within a novel academic KG. Chapter 8 presents new research trajectories on the integration of deductive and inductive mechanisms for a more comprehensible AI to non-insiders. The thesis ends with Chapter 9, with a synthesis of the research contribution and future work.

1.5 Publications

- (In Printing) Futia, G., Garifo, G., Vetrò, A., & De Martin, J. C. (2020), Modeling the semantics of data sources with graph neural networks. Bridge Between Perception and Reasoning: Graph Neural Networks and Beyond, ICML 2020 Workshop.
- Futia, G., Vetrò, A., & De Martin, J. C. (2020). SeMi: A SEmantic Modeling machIne to build Knowledge Graphs with graph neural networks. SoftwareX, 12, 100516.
- Futia, G., & Vetrò, A. (2020). On the Integration of Knowledge Graphs into Deep Learning Models for a More Comprehensible AI—Three Challenges for Future Research. Information, 11(2), 122.
- Futia, G., Vetro, A., Melandri, A., & De Martin, J. C. (2018). Training Neural Language Models with SPARQL queries for Semi-Automatic Semantic Mapping. Procedia Computer Science, 137, 187-198.
- Futia, G., Melandri, A., Vetrò, A., Morando, F., & De Martin, J. C. (2017, May). Removing barriers to transparency: A case study on the use of semantic technologies to tackle procurement data inconsistency. In European Semantic Web Conference (pp. 623-637). Springer, Cham.
- Futia, G., Morando, F., Melandri, A., Canova, L., & Ruggiero, F. (2015). ContrattiPubblici. org, a semantic knowledge graph on public procurement information. In AI Approaches to the Complexity of Legal Systems (pp. 380-393). Springer, Cham.

Chapter 2

Background

This Section provides the conceptual background required for understanding the rest of this thesis. It introduces the notions of Knowledge Graphs (KGs) and ontologies, it discusses the methodologies for enriching and refining KGs, and it describes KGs inference mechanisms enabled by inductive approaches.

2.1 KGs and Ontologies

KGs are labeled and directed multigraphs that encode information in the form of entities and relations relevant to a specific domain or organization. KGs are effective tools for capturing and organizing a large amount of structured and multi-relational data that can be explored employing query mechanisms. Considering these features, KGs are becoming the backbone of Web and legacy information systems in different research fields and industrial applications. The capability of KGs to manage information effectively is based on a twofold perspective: (i) the graph-based perspective allows us to perform graph and inductive learning algorithms on KGs; (ii) the semantic-based perspective provides a formal framework for the interpretation of the data, which is essential to perform deductive learning.

This twofold perspective is reflected in the notation adopted in this thesis, which defines a KG as follows: $G = \{E, R, T\}$, where G is a labeled and directed multigraph and E, R, T are the set of nodes, edges, and triples respectively. Each triple is formalized as $(u, e, v) \in T$, where $u \in E$ is the head node, $v \in E$ is the tail node, and $e \in T$ is the edge connecting u and v. In the semantic regime, a triple is formalized as a fact in which (u, e, v) corresponds to (s, r, o), where s and o are two entities, the subject and the object of the fact respectively, while r is the relation that connects s and o. To clarify this aspect, please consider the following example statement: "Socrates influenced Plato." From the graph perspective, the triple includes Socrates and Plato as two nodes of the graph, while influence is an

edge between these two nodes. From the semantic perspective, the simple statement includes Socrates and Plato as two entities, while influence is the relation between these two entities. Entities and relations can also be defined as *terms* of the statement.

2.1.1 Defining Simple Statements: the RDF Data Model

The Resource Description Framework (RDF) [94] is a core technology of the Semantic Web (SW) [20] for implementing the twofold representation of KGs. The purpose of SW is to realize a Web-scale data infrastructure readable and understandable by machines. To accomplish this vision, which is only partially implemented, Web technologies such as Uniform Resource Identifiers (URIs) [19] and the HyperText Transfer Protocol (HTTP) [54] protocol are adopted. RDF reflects a simple graph-based data model exploiting Web technologies and provides a formal notion of meaning (*semantics*) that set up the basis for founded deductions. An RDF graph can be serialized in different formats, including N-Triples [16], N3 [18], Turtle [16], JSON [150], and XML [15]. The example "Socrates influenced Plato" can be translated into RDF and serialized in N-Triples as follows:

```
<http://dbpedia.org/resource/Socrates>
<http://dbpedia.org/ontology/influenced>
<http://dbpedia.org/resource/Plato> .
```

Other types of serialization, such as Turtle, enable a more compact declaration of RDF facts employing the so-called prefixes. The example can be rewritten as follows[1]:

```
@prefix dbo: <http://dbpedia.org/ontology/> .
@prefix dbr: <http://dbpedia.org/resource/> .
dbr:Socrates dbo:influenced dbr:Plato .
```

From the SW point of view, the terms dbr:Socrates, dbo:influenced, and dbr:Plato are also known as *resources*, and a relation between two different resources - in this case dbo:influenced - is also known as *predicate* or *property*. This example directly comes from an existing *open* KG known as DBpedia, whose goal is to provide the information included in the Wikipedia infoboxes in the form of a collection of RDF triples. This collection can also be denoted as *RDF graph*.

The running example shows that URIs can be adopted as unique ids for both entities s to o and r. RDF allows us to represent relations between entities, but also specific information related to a single entity. In the RDF statement

[1]RDF statements reported in this thesis adopt N-Triples and Turtles interchangeably, sometimes omitting the prefixes when not necessary for the statement's intelligibility.

```
dbr:Socrates dbo:birthDate -469-0-0^^xsd:date .
```

the object is not an entity but a literal value, with a specific datatype defined using the XML Schema Datatypes (XSD) [161]. From a graph perspective, a literal value is considered a leaf node of the KG. Starting from the RDF representation, quantified statements are added to denote the semantics of entities and relations.

2.1.2 Defining Quantified Statements: RDFS and OWL

RDF Schema (RDFS) [26] and OWL (Web Ontology Language) [104] are two different language models that enable the construction of quantified statements in the form of RDF graphs. RDFS is one of the most prominent standards for defining a semantic schema for RDF graphs. OWL is the most popular ontology language used in practical cases, incorporating and extending RDFs. RDFS and OWL specify large vocabularies to denote peculiar conditions for using RDF nodes and edges, introducing coherency and specific datatypes to support logic reasoning. Moreover, they indicate how terms are interrelated and impose a structure to define the semantic interpretation's constraints.

RDFS

RDFS provides mechanisms to define classes for aggregating entities with similar features. In order to define classes, RDFS provides two different terms:

- the rdf:type property defines the "type" of a subject node; the object of this property must be a class. In RDF, rdf:type can be also replaced with a;
- the rdfs:Class is used to aggregate similar entities.

In the running example, the following statements can be defined to extend the information related to Socrates:

```
ex:Human rdf:type rdfs:Class .
dbr:Socrates rdf:type ex:Human .
```

RDFS allows also to define class hierarchies with the rdfs:subClassOf property. Therefore, the collection of facts can be expanded as follows:

```
ex:Human rdf:type rdfs:Class .
dbr:Socrates rdf:type ex:Human .
ex:Mortal rdf:type rdfs:Class .
ex:Human rdfs:subClassOf ex:Mortal .
```

RDFS also defines specific terms to extend the semantics of properties. These terms include:

- rdf:Property: it is the rdf:type of any terms used as predicate.;
- rdfs:domain and rdfs:range: they define the domain and the range of the property. Domain and range are categorized as rdfs:Class;
- rdfs:subPropertyOf: it specifies hierarchies of properties.

Therefore, the information related to the dbo:influenced property can be expressed as follows:

```
dbo:influenced rdf:type rdf:Property ;
    rdfs:range ex:Human ;
    rdfs:domain ex:Human .
```

Other classes and properties defined by RDFS are out of this thesis's scope, but further information is available in the RDFS W3C Recommendations [26].

OWL

OWL is a language model for describing ontologies. OWL documents, known as ontologies, incorporate the terms defined by RDFS and add further structures to address its limits. For instance, OWL makes explicit the relations of identity or unlikeness between different entities. It supports more expressive class definitions, including union, complement, disjointness, specifying cardinality restrictions. Moreover, it includes more expressive property definitions, enabling the distinction between object and datatype properties. OWL also allows us to define transitive, functional, symmetric, and inverse properties, indicating value restrictions. The statements which describe the relation between classes and properties are also known as *axioms*. These statements are categorized as *true* in the reference domain and are fundamental to enable deductive reasoning.

The running example can be extended with OWL as follows:

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix ex: <http://example.com/> .
@prefix dbr: <http://dbpedia.org/resource/> .
@prefix dbo: <http://dbpedia.org/ontology/> .
@prefic wkd: <https://www.wikidata.org/wiki/> .

# RDF
dbr:Socrates dbo:influenced dbr:Plato .
dbr:Socrates dbo:birthDate -469-0-0^^xsd:date .
```

```
# RDFS
dbr:Socrates rdf:type ex:Human .
ex:Human rdf:type rdfs:Class ;
    rdfs:subClassOf ex:Mortal .

dbo:influenced rdf:type rdf:Property ;
    rdfs:range ex:Human ;
    rdfs:domain ex:Human .

# OWL
dbr:Socrates owl:sameAs wkd:Q913 .
dbo:influenced rdf:type owl:ObjectProperty .
dbo:birthDate rdf:type owl:DatatypeProperty .
rdfs:subClassOf rdf:type owl:TransitiveProperty .
```

This example establishes quantified statements such as "All humans are mortal." Consequently, it also allows us to infer a new fact through deductive reasoning: "Socrates is mortal" derived from the fact that ex:Human is a subclass of ex:Mortal. In the same example, OWL statements allows us to declare that dbo:influenced and dbo:birthDate are two different types of properties, an object property and a data property respectively. Moreover, using OWL it is possible to define the axiom according to which rdfs:subClassOf is a transitive property. Other OWL terms and axioms are available within the OWL W3C recommendations [104].

2.1.3 Querying KGs

A practical language known as SPARQL (SPARQL Protocol and RDF Query Language) [69] has been developed in the SW context to perform queries on KGs. This structured query language's core is based on the graph pattern, which follows the same RDF graph model. In addition, SPARQL introduces the use of variables as valid terms. Therefore, graph patterns are divided into constants, represented by resources such dbr:Socrates, and variables such as ?philosopher, which are identified using question marks. During the querying process, the graph pattern is evaluated against the RDF graph. This process generates a mapping between the graph patterns' variables and the resources in the RDF graph. These variables are then replaced with the RDF resources that satisfy the graph patterns included in the query. Here is available an example of SPARQL query:

```
@prefix dbr: <http://dbpedia.org/resource/> .
@prefix dbo: <http://dbpedia.org/ontology/> .
```

```
SELECT ?philosopher
WHERE {
    dbr:Socrates dbo:influenced ?philosopher .
}
```

Considering our running example, the ?philosopher variable is replaced with the resource dbr:Plato. Advanced SPARQL features, including relational operator and path expressions, are out of this thesis's scope and are available in the SPARQL W3C recommendations [69].

2.2 Methodologies for Enriching and Refining KGs

This Section describes the main characteristics of two methodologies, and the related open problems, in the field of KGs. The first methodology consists of mapping from structured sources to KGs, which is fundamental for enriching KGs by integrating data provided by heterogeneous sources. The second methodology is related to the refinement of a KG, based on already-existing facts. The goal is to fill the missing edges that were not encoded during the generation or the enrichment stages.

2.2.1 Mapping from Structured Sources to KGs

Publishing data into KGs is a complex process because it requires extracting and integrating information from heterogeneous sources. The goal of integrating these sources is harmonizing their data and leading to a coherent perspective overall information. Heterogeneous sources range from unstructured data, such as plain text [60] [61] [27] [101], to structured schemes, including table formats such as CSVs and relational databases, and tree-structured data, such as JSONs and XMLs [72] [34] [31]. The thesis's research contributions are specifically related to the publishing of data into KGs from structured data sources. In fact, these types of sources play a fundamental role in the data ecosystem because much of the legacy information within organizations and on the Web is available as structured data. Unlike other sources, such as plain texts, structured information can be mapped to KGs through a *semantic integration* process. The common strategy adopted by the SW community to apply this process is adopting reference ontologies as global schemas. Then, mappings are constructed to describe the relationships between the global schema and the local schema of the target data source. From a data integration perspective, this approach is classified as Global-As-View (GAV)[43]. According to this perspective, the data integration performance is based on the consistency and expressiveness of the ontology adopted as a global schema. In order to clarify the mapping process in a real scenario, this subsection introduces a novel running example in the public procurement domain. Public procurement

refers to the process by which public authorities and administrations purchase goods or services from companies. In this process, the public authority announces a call for tenders, companies participate in this call with a specific tender, and the public authority shall award one of these tenders.

Suppose we have a target data source ds, which includes a set of attributes $ds\{a_1, a_2, a_3, ...\}$. In public procurement, this target data source is a JSON file representing the information on a specific public contract (see Listing 1). The JSON (JavaScript Object Notation) is a language-independent data interchange format. It employs human-readable text to store and transfer data objects characterized by attribute–value pairs and any serializable datatype, including arrays.

```
{
  "contract_id": "Z4ADEA9DE4",
  "contract_object": "MANUTENZIONE ORDINARIA MEZZI DI TRASPORTO",
  "proponent_struct": {
    "business_id": "80004990927",
    "business_name": "Ministero dell'Interno"
   },
   "participants": [
     {
       "business_id": "08106710158",
       "business_name": "CAR WASH CARALIS"
     }
   ]
}
```

Listing 1: JSON file including data related to a public contract

This JSON describes the following data:

- *contract_id* includes the identifier of the required service (“Z4ADEA9DE4");
- *contract_object* includes the description of the service (“MANUTENZIONE ORDINARIA MEZZI DI TRASPORTO");
- *business_id* and *business_name*, nested in the *proponent_struct* include the identifier (“80004990927") and the name (“Ministero dell’Interno") of the public body, which proposes the tender, respectively;
- *business_id* and *business_name*, nested in the *participants* include the identifier (“Z4ADEA9DE4") and the name (“CAR WASH CARALIS").

The mapping process requires a reference ontology O as global schema. The Czech OpenData.cz initiative releases one of the most common ontologies adopted in public procurement and it is available on GitHub[2]. The running example considers only a subset of this ontology's axioms, while an extensive description is available in Chapter 4. The axioms involved in the mapping process can be represented in Turtle format as follows:

```
@prefix dcterms: <http://purl.org/dc/terms/> .
@prefix gr: <http://purl.org/goodrelations/v1#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix pc: <http://purl.org/procurement/public-contracts#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

#######
####### Classes
#######

pc:Contract a owl:Class .
gr:Offering a owl:Class .
pc:Tender a owl:Class ;
    rdfs:subClassOf gr:Offering .

#######
####### Relations (or Object Properties)
#######

pc:contractingAuthority a owl:FunctionalProperty, owl:ObjectProperty ;
    rdfs:domain pc:Contract ;
    rdfs:range gr:BusinessEntity .

pc:tender a owl:ObjectProperty ;
    rdfs:domain pc:Contract ;
    rdfs:range pc:Tender .

pc:awardedTender a owl:FunctionalProperty, owl:ObjectProperty ;
    rdfs:subPropertyOf pc:tender .

pc:bidder a owl:ObjectProperty ;
```

[2]The Public Contracts Ontology: https://github.com/opendatacz/public-contracts-ontology

```
    rdfs:domain pc:Tender ;
    rdfs:range gr:BusinessEntity .

#######
####### Datatype properties
#######

dcterms:identifier a owl:DatatypeProperty ;
    rdfs:domain pc:Contract ;
    rdfs:domain gr:BusinessEntity ;
    rdfs:range rdfs:Literal .

rdfs:label a owl:DatatypeProperty ;
    rdfs:domain pc:Contract ;
    rdfs:domain gr:BusinessEntity ;
    rdfs:range rdfs:Literal .

rdfs:description a owl:DatatypeProperty ;
    rdfs:domain pc:Contract ;
    rdfs:range rdfs:Literal .
```

The entire mapping process includes two main steps. The first step is creating a map between the local schema of the target data source and the reference ontology. The second step is materializing the source's data as KG statements or virtualizing the access to the source, defining a graph-based view over the legacy information. Materialized statements can be directly published into KGs, while a graph-based and virtualized access allows us to retrieve and explore data of the target data source like it is a KG. The widest-adopted approaches for the mapping step are based on the so-called *custom mappings*. These approaches exploit customizable documents written with declarative languages to perform the map generation step. Declarative languages exploit the SW formalism to describe the relationships between the local and the global schemas. The most prominent language adopted by the research community is R2RML [37], which expresses customized mappings written in RDF between relational databases to KGs. An extension of this language, known as RML [40], is a more generic mapping language, whose applicability is extended to other types of tables, such as CSV files, and tree-structured schemes. Other types of languages, such as TARQL [35] and JARQL [143], adopt the SPARQL syntax to create mappings for specific formats, such as CSV and JSON files respectively. An example of JARQL describing the mapping between the *ds* and *O* is available below:

```
@prefix dcterms: <http://purl.org/dc/terms/> .
```

```
@prefix pc: <http://purl.org/procurement/public-contracts#> .
@prefix gr: <http://purl.org/goodrelations/v1#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .


CONSTRUCT {
    ?BusinessEntity0 dcterms:identifier ?proponent_struct__business_id ;
        rdf:type gr:BusinessEntity.
    ?BusinessEntity1 dcterms:identifier ?participants__business_id ;
        rdf:type gr:BusinessEntity .
    ?Tender0 pc:bidder ?BusinessEntity1 .
    ?Contract0 dcterms:identifier ?contract_id ;
        rdf:type pc:Contract ;
        pc:contractingAuthority ?BusinessEntity0 ;
        pc:tender ?Tender0 .
}

WHERE {
    ?root a jarql:Root.
    OPTIONAL { ?root jarql:contract_id ?contract_id . }
    OPTIONAL { ?root jarql:proponent_struct ?proponent_struct . }
    OPTIONAL { ?proponent_struct jarql:proponent_struct__business_id
    ?proponent_struct__business_id . }
    OPTIONAL { ?root jarql:participants ?participants . }
    OPTIONAL { ?participants jarql:participants__business_id
    ?participants__business_id . }
    BIND (URI(CONCAT('http://purl.org/procurement/public-contracts/contract/',
    ?contract_id)) as ?Contract0)
    BIND (URI(CONCAT('http://purl.org/goodrelations/v1/businessentity/',
    ?proponent_struct__business_id)) as ?BusinessEntity0)
    BIND (URI(CONCAT('http://purl.org/goodrelations/v1/businessentity/',
    ?participants__business_id)) as ?BusinessEntity1)
    BIND (URI(CONCAT('http://purl.org/procurement/public-contracts/tender/'
    ?contract_id + + '_' participants__business_id)) as ?Tender0)
}
```

This JARQl file includes 3 main parts, the first one is included in the CONSTRUCT section, while the others are included in the WHERE section. The CONSTRUCT section describes the graph patterns that encode the semantic types, such as "?BusinessEntity0 dcterms:identifier ?proponent_struct__business_id" and the semantic relations, such as "?Contract0 pc:contractingAuthority ?BusinessEntity0".

The first segment of the WHERE section, which includes the OPTIONAL operators, describes the way to parse the JSON for extracting the data required to create the KG facts. For instance, the pattern "?proponent_struct jarql:proponent_struct___business_id ?proponent_struct___business_id" indicates that the variable ?proponent_struct___business_id has to be replaced with the proponent_struct___business_id attribute of the JSON. The second segment of the WHERE section, which includes different BIND operators, declares how to generate the entity URIs for the data extracted from the JSON. The line BIND (URI(CONCAT('http://purl.org/procurement/public-contracts/contract/', ?contract_id)) as ?Contract0) indicates the URIs contract entities are built combining the http://purl.org/procurement/public-contracts/contract/ URI and the value extracted from the values that replace the ?contract_id variable.

Language-driven engines have to goal to materialize or virtualize KG statements, following the instructions of the mapping documents written using declarative languages. These engines perform two different tasks: the first task is to link the the target data source fields to a class or a property defined by the reference ontology. Once this link has been created, these engines materialize or virtualize the URIs and the KG statements, retrieving the legacy information included in the data source. The JARQL file describing the link between the set of attributes $ds\{a_1, a_2, a_3, ...\}$ and the global schema represented by O allows to materialize the following statements:

```
@prefix dcterms: <http://purl.org/dc/terms/> .
@prefix pc: <http://purl.org/procurement/public-contracts#> .
@prefix gr: <http://purl.org/goodrelations/v1#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix contract: <http://purl.org/procurement/public-contracts/contract/> .
@prefix be: <http://purl.org/goodrelations/v1/businessentity/> .
@prefix tender: <http://purl.org/procurement/public-contracts/tender/> .

# ?BusinessEntity0 dcterms:identifier ?proponent_struct__business_id ;
#      rdf:type gr:BusinessEntity .

be:08106710158 dcterms:identifier 08106710158 ;
    rdf:type gr:BusinessEntity .

# ?BusinessEntity1 dcterms:identifier ?participants__business_id ;
#        rdf:type gr:BusinessEntity .

be:08106710158 dcterms:identifier 08106710158 ;
    rdf:type gr:BusinessEntity .

# ?Tender0 pc:bidder ?BusinessEntity1 .
```

```
tender:Z4ADEA9DE4-80004990927 pc:bidder be:08106710158 .

# ?Contract0 dcterms:identifier ?contract_id ;
#     rdf:type pc:Contract ;
#     pc:contractingAuthority ?BusinessEntity0 ;
#     pc:tender ?Tender0 .

contract:Z4ADEA9DE4 dcterms:identifier Z4ADEA9DE4 ;
    rdf:type pc:Contract ;
    pc:contractingAuthority 80004990927 ;
    pc:tender tender:Z4ADEA9DE4-80004990927 .
```

In order to clarify the transformation process enabled by the JARQL file, this example reports as comments the graph patterns included in its CONSTRUCT section. One of the main reasons for using JARQL as declarative language is its capacity to create URIs, combining data located at different levels of the JSON tree structure. In the running example, the Tender's URI is built combining the id of the contract (Z4ADEA9DE4), located at the JSON's root level, and the id of the business entity (80004990927), located in a nested structure of the JSON.

Mapping Automation: the Semantic Model Approach

Mapping data source schemas to reference ontologies is a tedious task requiring a significant manual effort and domain knowledge expertise. For this reason, automatic techniques are fundamental to scale the mapping process. The research contributions described in this thesis to automatize the mapping adopt a graph-based approach to the mapping problem. According to this approach, proposed for the first time by Knoblock et al. [89], the result of the mapping can be seen as a graph, known as *semantic model*, which is able to express the links between the local schema, represented by the attributes of the target data source, and the global schema, represented by the reference ontologies. A semantic model is a powerful tool for representing the mapping for two main reasons. In the first place, it frames the relations between ontology classes as paths in the graph. Secondly, it enables the computation of graph algorithms to detect the correct mapping.

The implementation of the semantic model approach requires that ontology axioms are represented in a graph structure. The ontology O can be seen as a directed, typed, labeled, and multi-relational graph depicted in Figure 2.1. The graph nodes represent the classes defined by the ontology. The edges represent the different types of property in terms of object properties, depicted with black arrows, data properties, depicted with the grey-dashed arrows, and subclass relations, depicted with grey arrows.

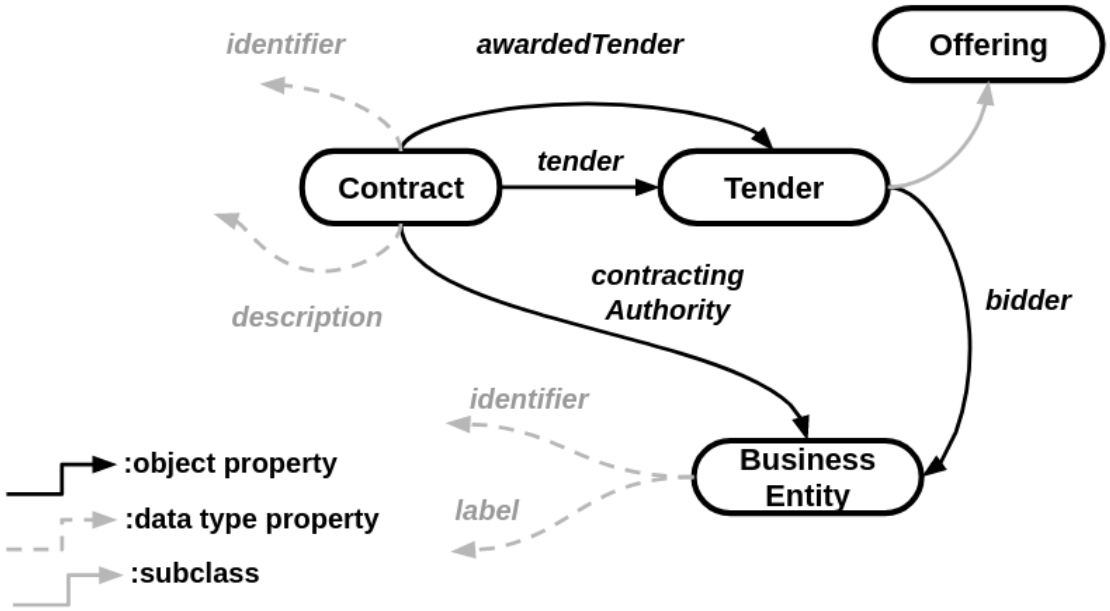

Figure 2.1: Ontology graph describing the public procurement domain.

The mapping can be framed into the semantic model $sm(ds)$. The semantic model is a directed and labeled graph, whose leaf nodes represent the attributes of ds, while other parent nodes and edges derive from the properties defined in O. In the running example, the correct semantic model is depicted in Figure 2.2.

The graph depicted in Figure 2.2 corresponds to CONSTRUCT section of the JARQL file introduced in Subsection 2.2.1[3]. One of the main limits of the semantic model provided by Knoblock et al. is that it does not provide features for building URIs. The URI generation is not a research focus of this thesis. Nevertheless, one of the approaches presented in this thesis (see Chapter 6) addresses this issue implementing specific solutions.

The automatic generation of the semantic model is based on two main steps. The first step is the Semantic Type Detection (STD) or semantic labeling, where each attribute of ds is annotated with a pair of an ontology class and a datatype property: $sl_1(a_1) = \langle c_{a_1}, p_{a_1} \rangle$. From a graph-based perspective related to the running example, the attribute node "contract_id" (a_1) is connected to the class node "pc:Contract" (c_{a_1}) through the datatype property "dcterms:identifier" (p_{a_1}) (Figure 2.3). The second step is the Semantic Relation Inference (SRI), whose goal is to identify the connections between the annotated attributes within the target data source. In the simplest case, this relation is represented by an object property,

[3]The ontology namespaces have been removed from the graphical representation of the ontology graph to improve clarity

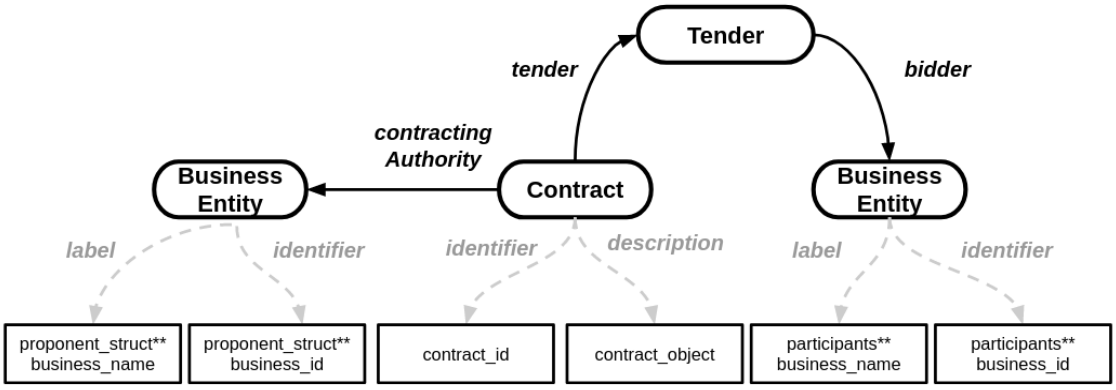

Figure 2.2: Sample of semantic model describing the semantics of the JSON data.

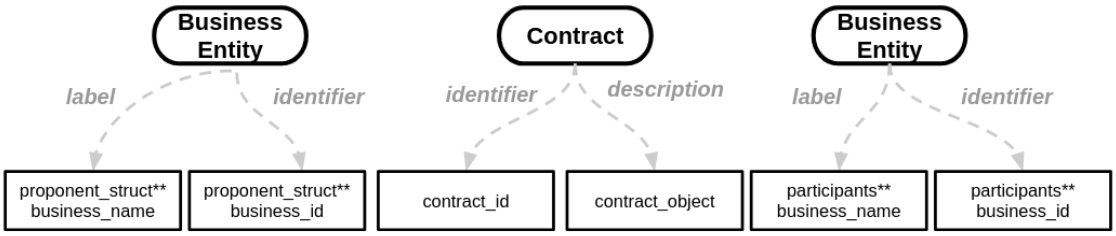

Figure 2.3: Semantic types annotating the attributes of the JSON structure.

$sr_1(sl_1, sl_2) = c_{a_1} \xrightarrow{p_{o1}} c_{a_2}$, and the length of the relation (or path) is equal to 1. In the running example, the classes “pc:Contract" and “gr:BusinessEntity" that annotate the attributes “contract_id" and “proponent_struct**business_id" are connected with the property “pc:contractingAuthority" (Figure 2.4). In more complex situations, the path includes different ontology classes and properties, $sr_1(sl_1, sl_2) = c_{a_1} \xrightarrow{p_{o2}} c_1 \xrightarrow{p_{o3}} c_{a_2}$, and its length is longer than 1. In the public procurement domain, this type of semantic relation is includes the class nodes “pc:Contract", “pc:Tender", and “gr:BusinessEntity". “pc:Contract" and “pc:Tender" are connected by the object property “pc:tender", while “pc:Tender" and “gr:BusinessEntity" are connected by the object property “pc:bidder" (Figure 2.5). This example demonstrates that inferencing the correct semantic relation is a complex task. In fact, many plausible semantic relations of different lengths can be plausibly correct for building the semantic model of the target source. Figure 2.6 shows an example of an incorrect semantic model for the target source.

As introduced in Chapter 1, this thesis intends to address the research problem of inferencing the correct semantic relations between annotated attributes and proposes inductive techniques based on representation learning (see Chapters 5 and 6). The main principles behind these techniques are discussed in Section 2.3.

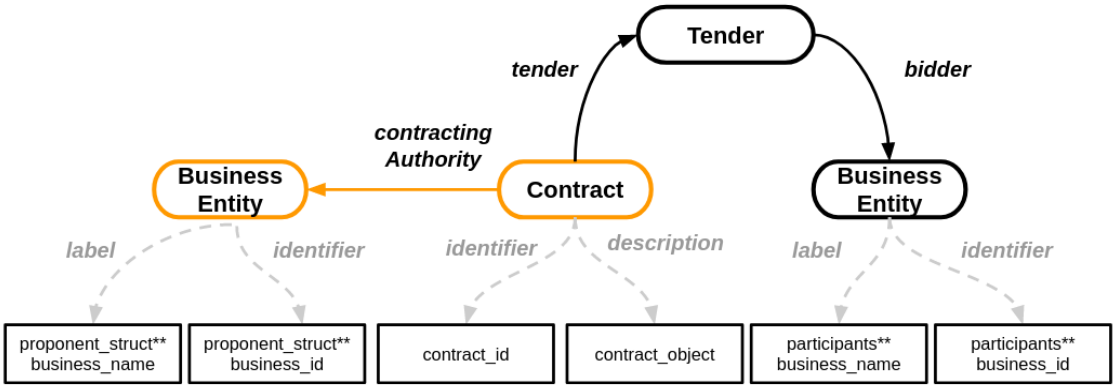

Figure 2.4: Semantic relation with length equals to 1.

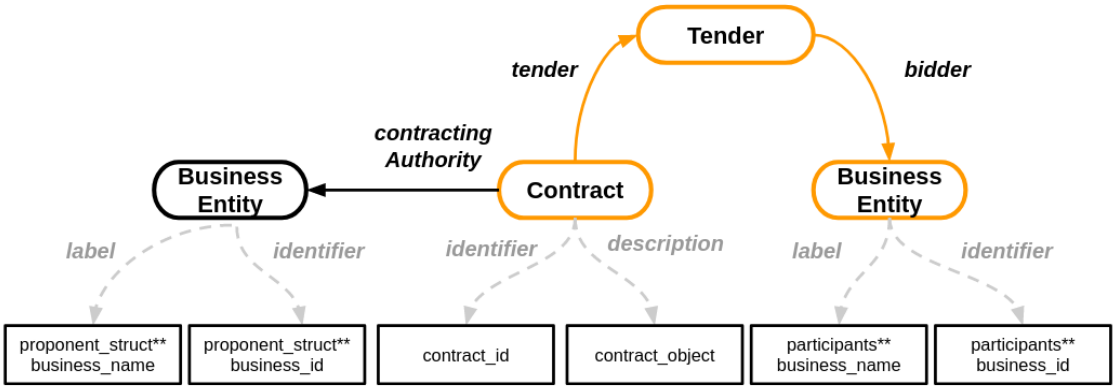

Figure 2.5: Semantic relation with length equals to 1.

2.2.2 Refinement of KGs

KGs are able to continuously aggregating new information, incorporating data through the mapping procedure. Despite this aggregation process, KGs are inevitably characterized by incompleteness. The refining (or completion) task in a KG aims at filling the missing edges that were not encoded during the generation process. The task of predicting new edges (also known as *link prediction*) can be applied to 3 different classes of link: (i) *type links*, involving edges that indicate the type of an entity; (ii) *identity links*, involving edges corresponding to the property owl:sameAs, which indicates nodes referring to the same entity; (iii) *general links* involving arbitrary labels for the edges. For the thesis's purpose, which intends to focus on recommendation purposes, general links are the preferred to be predicted. As mentioned in Section 2.1, deductive reasoning allows us to combine simple and quantified statements for inferencing new facts. Such new facts are logically understandable and contribute to the KG completion in a semantic regime. However,

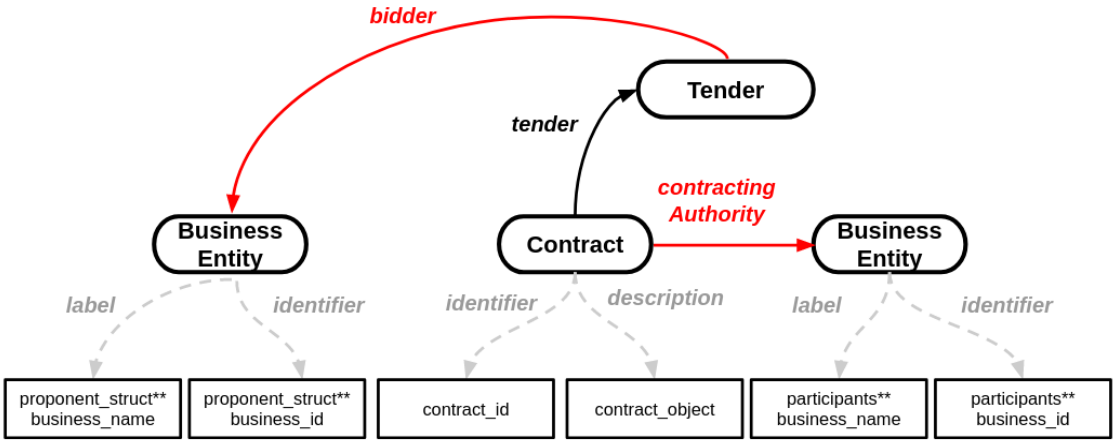

Figure 2.6: Incorrect semantic model of the target source.

deductive inference does not allow us to predict new facts that can be useful for application-based systems, such as recommendation tools. Inductive techniques can be adopted to define the likelihood of correctness of a new (semantic-valid) fact. To understand the impact of inductive approaches for predicting general links, please consider the following example in academic publications. This running example is built starting from real data available on the publications repository of the Politecnico di Torino, which is called IRIS[4]. The RDF statements involved in the example are created in the Geranium project context, whose details are discussed in Chapter 7.

```
@prefix ger: <http://geranium-project.org/> .
@prefix aut: <http://geranium-project.org/authors/> .
@prefix pub: <http://geranium-project.org/publications/> .
@prefix jou: <http://geranium-project.org/journals/> .
@prefix key: <http://geranium-project.org/keywords/> .
@prefix onto: <http://geranium-project.org/ontology/> .
@prefix dcterms: <http://purl.org/dc/terms/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

aut:rp31513 a ger:Author ;
    rdfs:label "Daniela De Luca" .

pub:11583/2680251 a ger:Publication ;
```

[4] https://iris.polito.it/

```
    dcterms:title "District data management,
    modelling and visualization via interoperability" ;
    dcterms:subject key:DataManagement ;
    dcterms:contributor aut:rp31513 .

aut:rp12842 a ger:Author ;
    rdfs:label "Antonio Vetrò" .

pub:11583/2381987 a ger:Publication ;
    dcterms:title "Linked Data approach for
    selection process automation in Systematic Reviews" ;
    dcterms:subject key:DataManagement ;
    dcterms:contributor aut:rp12842 .
```

These statements describe information related to two different publications: pub:11583/2680251 and pub:11583/2381987. They also declare the corresponding contributors, which are Daniela De Luca and Antonio Vetrò respectively. In the IRIS repository, these two researchers have never been co-authors of the same papers. However, a recommendation system built on top of these data, could suggest a collaboration opportunity between Antonio Vetrò and Daniela De Luca, because their articles address the same research topics (key:DataManagement). Predicted edges can connect, for instance, a researcher with a publication that does not belong to him. However, a high score assigned to this prediction can be interpreted according to a recommendation perspective. For instance, the recommendation system can suggest the topics related to this publication, which may still be unexplored by the researcher. Moreover, the publication's authors could be suggested as collaboration opportunities if the researcher has never worked with them.

can be predicted by inductive techniques. Section 2.3 introduces the application of these techniques focused on the representation learning.

2.3 Inference in KGs with Representation Learning

According to deductive learning, the knowledge acquired by machines has to be hard-coded in the system exploiting formal languages, such as ontologies. Consequently, the system can reason on the simple and the quantified statements expressed in these formal languages through logical inference rules. From an opposite perspective, the main principle behind inductive methods indicates that machines are able to derive their own knowledge on the data, discovering and generalizing patterns within a set of input observations. Machines prove this capability, known

as *machine learning*, by means of statistical learning approaches. The performance of machine learning algorithms strictly depends on the representation of the data, which is traditionally achieved by defining a hand-designed set of features. However, in many learning tasks such as computer vision, these features' manual detection is not an easy and effective process.

Representation learning [17] is an approach that improves the machine learning performance in these types of situation, deriving the set of features for representing data without human intervention. Namely, a representation learning algorithm's goal is to automatically infer *latent* - or hidden - features of the data. These features are represented as dense and low-dimensional vectors, known as *embeddings*, which are not directly extracted as quantities in the data, but whose variations influence every single piece of information we are able to observe. The key algorithm for the embeddings learning is the *back-propagation* [135]. The goal of the back-propagation is to efficiently compute the gradient of the loss function with respect to the weights, or parameters, of a model. Through this gradient computation, it is possible to update the weights and obtain the embeddings to minimize the loss value. As a consequence, the learning process can be reduced to an optimization problem, finding a function that produces the minimal loss. Such methodological advances, in combination with the increasing of computational resources and the availability of large datasets, make the modern representation learning techniques very powerful in being trained without supervision, extracting patterns and regularities from the input data.

2.3.1 Knowledge Graph Embeddings

Knowledge Graph Embeddings (KGEs) are the result of specific representation learning models applied to KGs. The goal of KGE models is to embed the KG components - entities and relations - into a continuous and low-dimensional vector space. The latent factors projected into KGEs have an essential role in analyzing and mining additional *soft*-knowledge in KGs. In fact, for each entity pair $s, o \in E$ and any relation $r \in R$, it is possible to determine if a statement (s, r, o) is true according to the embeddings learnt by KGE techniques. In this thesis, the latent feature representations of s, r, o are defined with their related bold face $\boldsymbol{s}$, $\boldsymbol{r}$, $\boldsymbol{o}$. The KGE approaches define a *scoring function* $f_{sco}(\boldsymbol{s}, \boldsymbol{r}, \boldsymbol{o})$ for each KG statement (s, r, o). In general KGs comprises only true statements, while non-existing statements can be consider either missing or false. For these reasons, a closed-world assumption is deemed to address this ambiguity in categorizing non-existing statements (see Section 2.1 for further details). As a consequence, the scoring function $f_{sco}(\boldsymbol{s}, \boldsymbol{r}, \boldsymbol{o})$ returns a higher score for existing statements and a lower score vice-versa. For the scope of this thesis, we consider KGE models which formalize the training goal as a binary classification problem. Therefore, the entity and the relation embeddings are learnt by minimizing the cross-entropy loss:

$$L = -\frac{1}{(1+neg)|\hat{T}|} \sum_{\hat{y}\epsilon T,T'} y \log \hat{y} + (1-y) \log(1-\hat{y}) \tag{2.1}$$

, where

- neg is the number of negative samples obtained corrupting each true fact included in the training set;
- $\hat{T}$ is the subset of all existing facts that are included in the training set;
- $T' = E \times R \times E \times -T$ includes the set of non-existing facts obtained corrupting entities and relations available in the existing ones;
- $\hat{y} = \sigma(f_{sco}(\boldsymbol{s}, \boldsymbol{r}, \boldsymbol{o}))$, where σ is the logistic sigmoid function, which map any values into a real number between 0 and 1 (a necessary step for the binary classification purpose).

In the most recent literature review on KGs [79], three different types of KGE techniques have been proposed: (i) *linear* or *translational* models [24] [], which formulate the relations between two entities as a linear transformation, projecting $\boldsymbol{s}$ into $\boldsymbol{o}$; (ii) *bilinear* or *factorization* models, where $\boldsymbol{r}$ is encoded within a matrix and combine $\boldsymbol{s}$ and $\boldsymbol{o}$ by a multiplication. This multiplication can be seen, from another perspective, as a factorization method to decompose the relational data for the representation learning purpose; (iii) *neural* models, which are more flexible than the previous techniques, due to the multiple parameters enclosed in their network structure. The following subsections describe specialized neural architectures for the embedding generation, which are able to exploit sequence structures, such as Neural Language Models (NLMs), and graph structures, such as Graph Neural Networks (GNNs).

2.3.2 Neural Language Models and Word2Vec

Neural Language Models (NLM) are popular methods for learning *word embeddings* (WE). These real-valued representations are able to convey a distributed semantics of the words available in a text corpus used for training the model. The seminal paper of Mikolov et al. [108] proposes a simple, but efficient NLM known as Word2Vec, which is implemented in two specific variants: Continuous Bag-Of-Words (CBOW) and Skip-Gram 2.7.

The CBOW model's goal is to predict a word given its context. The latter is defined as the window of words to the left and the right of a target word in a sentence. In CBOW, the network has the following structure: (i) the input layer is characterized by the surrounding words of the target, whose embedding representations are retrieved from the input weight matrix and projected in the

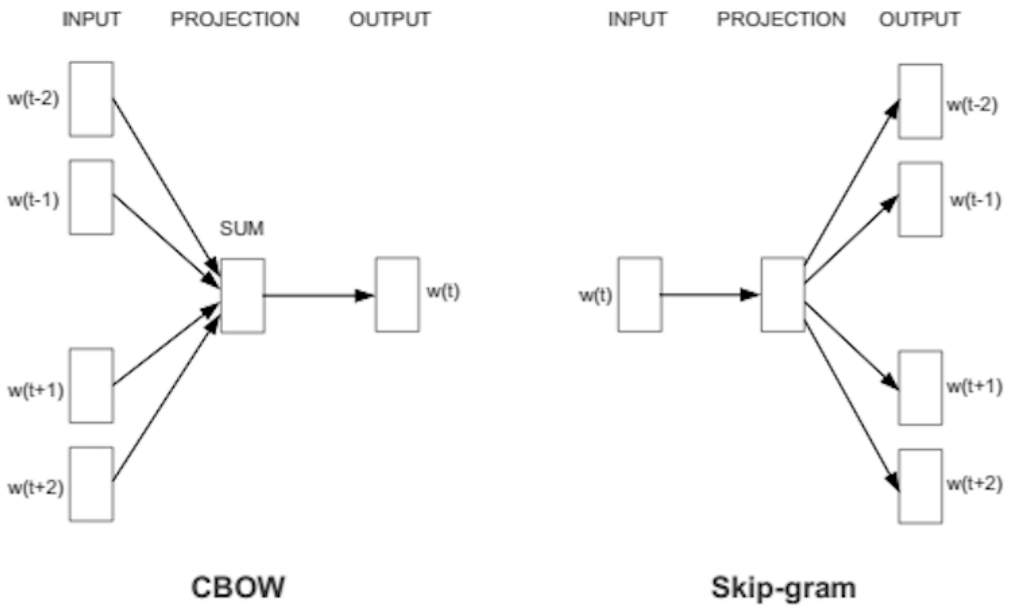

Figure 2.7: Architecture of Word2Vec models: CBOW and Skip-Gram

projection layer; (ii) using the output weight matrix between the projection layer and the output layer, a score for each word in the vocabulary is computed. This score value is the likelihood of the word being a target word.

More formally, given a sequence of words w_1, w_2, ..., w_T, the objective of CBOW is to maximize the log probability of the target word:

$$1/T \sum_{t_1}^{T} \sum_{-m \leq j \leqslant m, j \neq 0} \log p(w_t | w_{t+j}) \tag{2.2}$$

where m is the size of the window context used for the training, which is a function of target word w_t .

To compute the probability $p(w_t|w_{t+j})$, the CBOW formulation employs the softmax function as follows:

$$p(w_O|w_I) = \frac{\exp((v'_{w_O})^\top v_{w_I})}{\sum_{w=1}^{W} \exp((v'_w)^\top v_{w_I})} \tag{2.3}$$

where $w_O = w_t$, $w_I = w_{t+j}$ in j-th network propagation related to a specific context window, v_w and v'_w are the vector representations of w and W is the number of words in the vocabulary.

As in the case of KGE, WE are learned training the network in a cross-entropy loss regime, because the softmax represents a non-linear variant of multinomial logistic regression.

$$L = -\frac{1}{T}\sum_{t=1}^{T}\sum_{-m\leq j\leqslant m, j\neq 0} \log p(w_t|w_{t+j}) \tag{2.4}$$

The Skip-Gram model's goal is the opposite of CBOW because it predicts the context words from the target word. The network has the following structure: (i) the input layer is constituted by the target word, whose vector representation is retrieved from the input weight matrix and projected in the projection layer; (ii) using the output weight matrix between the projection layer and the output layer characterized by the surrounding words of the target, a score for each word in the vocabulary is computed. This score value is the probability of the word being a context word.

More formally, given a sequence of words w_1, w_2, ..., w_T, the objective of Skip-Gram is to maximize the log probability of the target word:

$$1/T\sum_{t_1}^{T}\sum_{-m\leq j\leqslant m, j\neq 0} \log p(w_{t+j}|w_t) \tag{2.5}$$

To compute the probability, the Skip-Gram formulation adopts the softmax function of CBOW declared in Equation 2.3 and the only difference is that $w_O = w_{t+j}$, while $w_I = w_t$. In a similar way, in the loss function indicated in Equation 2.4 the probability definition has to be replaced with $\log p(w_t|w_{t+j}$.

Specific methods in the context of KGs have been inspired by Word2Vec to learn node (or entity) embeddings. RDF2Vec [130] is one of the most prominent approaches in this context. In its original version, RDF2Vec exploits random walks within KGs in order to create sequences of KG entities. Then, these sequences are used as input for the Word2Vec algorithm. RDF2Vec proved to be very powerful in generating entity embeddings for different purposes, from data mining tasks to recommender systems [131].

Considering the promising results in employing NLM for entity embeddings, this thesis presents an initial study that presents an approach where Word2Vec has been trained with SPARQL queries, whose graph patterns are adapted as input sequences of the model. This approach assigns a vector representation to the SPARQL variables, it aggregates such variables in clusters, which are labeled with the most common relation extracted from the related graph patterns. It then employs these semantic relations to connect the attributes of the data source, exploiting their syntactic closeness to the SPARQL variables. Further details on the implementation and the results are available in Chapter 5.

2.3.3 Graph Neural Networks

KGE techniques and neural language models encode the interactions between entities and relations through models that are not natively built for encoding graph

structures. However, a novel family of neural architectures has been proposed to address this limitation. In this context, Graph Neural Networks (GNNs) are becoming the key framework for learning the latent representation of graph-structured data.

The operating principle behind GNNs is based on the analogy between NNs and graphs. In fact, neural architectures are formalized as weighted and directed graphs, whose nodes define the computational units - or artificial neurons - and edges are the weighted connections between these units. Nevertheless, the topology of traditional NNs is a priori structured for a specific purpose. For instance, the feed-forward architectures provide a fully-connected and homogeneous sequence of layers: each node of the previous layer is linked by a weighted edge to all the nodes in the next layer. On the contrary, the graph data structure is heterogeneous because the topology is directly driven by the edges defined between the nodes. GNNs are based on neural architectures designed following graph data topology, where the weighted connections of the NN match the edges available in the graph structure.

The most-adopted class of GNNs is known as Graph Convolutional Networks (GCNs). The goal of the GCNs is to update the representation of nodes from one layer to the other on the basis of the following transformation: $h_i^{l+1} = f_{agg}(h_i^l, \left\{h_j^l\right\}_{j \in N_i})$, in which:

- $h^l \in \mathbb{R}^{n \times d}$ is the hidden representation of the nodes in the l-th layer;
- $h^{l+1} \in \mathbb{R}^{n \times d}$ is the hidden representation of the nodes in the $l+1$-th layer;
- $\left\{h_j^l\right\}_{j \in N_i}$ includes all the hidden representations of the neighbors N_i of the node i in the l-th layer.
- f_{agg} is an aggregation function, which establishes how to accumulate the representations of N_i into the node i.

This type of transformation has proven to be very powerful in aggregating and encoding features of the nodes in the graph and led to a significant advance in graph-based tasks, such as node classification [88]. On the basis of this powerful model, the forward propagation step can be defined as a *recursive neighborhood diffusion* and formalized according to the following Equation:

$$h_i^{l+1} = \sigma\left(h_i^l \, W_1^l \; + \sum_{j \in N_i} h_j^l \, W_2^l\right), \tag{2.6}$$

where σ is a non-linear activation function such as $ReLU$ and $W_1^l, W_2^l \in \mathbb{R}^{d \times d}$ are two weight (or parameter) matrices. The key idea behind Equation 2.6 is that the representation of a node in the graph is updated with the features of its

neighbors, through the convolution operation computed with shared parameters (or weights) across the local structure. This computation is performed in parallel for all nodes at each network update and, by stacking up several layers, it is possible to capture and encode the relations between nodes across multiple hops in the graph. A graphical representation of this propagation model is reported in Figure 2.8. Stacking a single GCN layer, the representation of the central node depicted in orange is updated with the features of 1-hop neighbor nodes depicted in blue (see the left area of Figure 2.8). Stacking an additional layer, its representation is further updated with the the features of 2-hop neighbor nodes depicted in green (see the central area of Figure 2.8). As shown in the right area of Figure 2.8, the updated features of the central node aggregate the representation of its local graph structure.

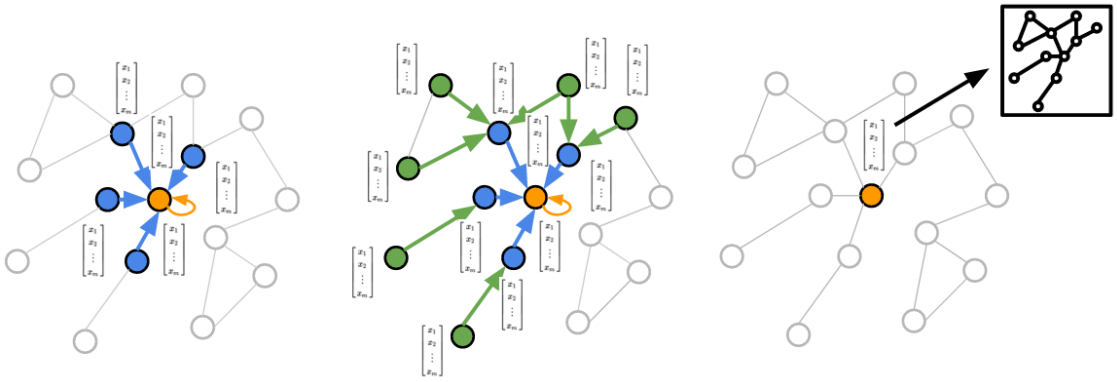

Figure 2.8: Propagation model in GCNs

In the *isotropic* formulation of GCNs, the nodes receive an equal contribution from each neighbor, obtaining the same weight values, in this case W_2^l, along each "direction" defined by each edge. The most popular isotropic GCNs is the Vanilla GCNs [88], in which the node representations are updated via the isotropic average computation over the features of neighbor nodes.

$$h_i^{l+1} = ReLU\left(h_i^l\, W_1^l \;+\; \frac{1}{deg_i} \sum_{j \in N_i} h_j^l\, W_2^l\right), \tag{2.7}$$

where deg_i is the in-degree of node i. If necessary, the features of the central node h_i can be included in the update operation by means of self-loops.

2.3.4 Graph Auto-Encoder Framework

As mentioned in subsection 2.3.1, KGE techniques are not able to encode the graph structure: the embeddings representing entities and relations are directly optimized during the training process. On the other hand, GNN models are

natively-built to encode the local neighborhood structure into the node (or entity) representation. Considering their respective characteristics, GNN models and KGE techniques can be incorporated into an end-to-end architecture, also known as *Graph Auto-Encoder* (GAE). Goodfellow et al. [63] describe the auto-encoders as a quintessential example of the representation learning algorithms. An auto-encoder combines two different components: (i) an *encoder*, whose goal is to transform the input data into a different representation, typically for feature learning and dimensionality reduction tasks; (ii) a *decoder*, whose objective is to employ this new representation to reconstruct the original data. In the domain of this thesis, the GNN models play the role of the encoder, which produces an enriched representation of all entities within the KG, accumulating the neighborhood features; the KGE techniques play the role of the decoder, exploiting this enriched representation to reconstruct the edges within the KG. Based on the notation adopted in this thesis, the real-valued vector of the subject entity corresponds to the representation of the i-th node: $\boldsymbol{s} = h_i^L$, where h_i^L is the latent representation at the last-stacked layer of the GNN. The real-valued vector of the object entities corresponds to the latent representation of N_i are: $\boldsymbol{o} = h_j^L$. The scoring function introduced in Subsection 2.3.1 can be update as follows: $f_{sco}\left(h_i^L, \boldsymbol{r}, h_j^L\right)$. The cross entropy loss 2.1 is computed per edge and allows us to obtain valuable embeddings for the link prediction task. A schematic representation of the GAE is represented in Figure 2.9.

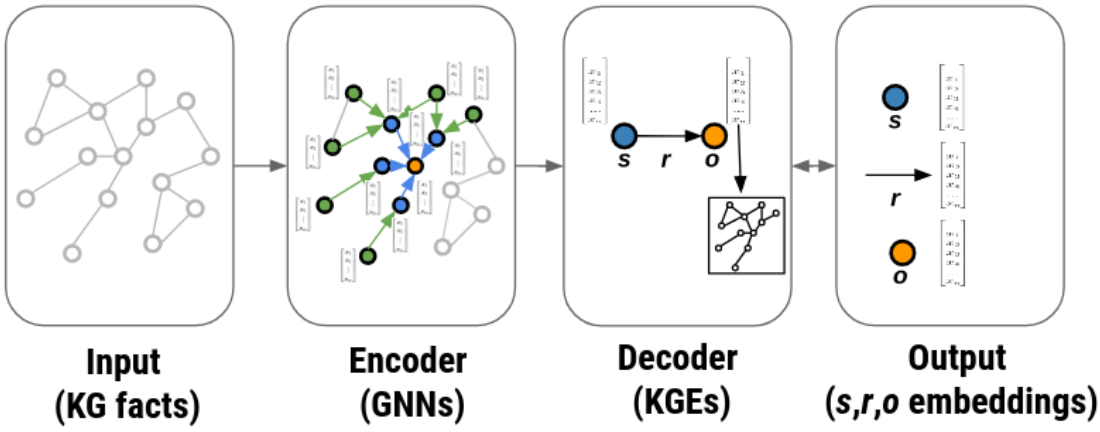

Figure 2.9: Graph Auto-encoder Architecture

The effectiveness of the GAE architecture has been fully demonstrated by Schlichtkrull et. al [144]. Thus, the thesis proposes a novel mapping approach in the procurement domain based on the GAE framework (see Chapter 6). This approach supports the generation of semantic models of data sources, particularly in the automatic inferences of semantic relations between the source's annotated

attributes. Moreover, the same framework is adopted in a more traditional application context, such as link prediction. However, the GAE is employed in a completely new domain, such as the academic publications, for building a recommendation system useful in a real scenario (see Chapter 7).

Chapter 3

Related Work

This Chapter provides a comprehensive overview of the research contributions related to the open problems described in Section 2.2. The discussed open problems are related to the publication of high-quality KGs and are focused on: (i) the automatic mapping of data source schemas to reference ontologies for the KG generation or enrichment from structured data; (ii) the automatic KG completion, through the prediction of new statements based on the existing facts.

3.1 Mapping of Structured Sources to KGs

The automatic mapping of data sources to KGs is challenging for different reasons, including the lack of metadata, incompleteness, and noisy data. In addition, different types of source, such as CSV, XML, JSON files, and relational databases, have different features that can be taken into consideration during the mapping process. Therefore, this Section describes the full landscape of approaches related to the mapping problem, including (i) the mapping from structured data to KGs; (ii) the mapping from relational databases to KGs; (iii) the subset of approaches that frame the mapping problem in a semantic model perspective, considering both structured sources and relational databases. This landscape analysis shows that the adoption of the inductive learning method based on NNs is not yet fully explored and developed. This circumstance spotlights the novelty of the thesis contributions, which adopts Neural Language Models (NLMs) and Graph Neural Networks (GNNs) to address this open problem.

Despite the significant amount of work to address the mapping problem from structured data to KGs, there is a lack of a common framework to perform a systematic evaluation of different approaches. For this reason, very recent initiatives (2019) such as the "SemTab: Semantic Web Challenge on Tabular Data to

Knowledge Graph Matching[1]" aims at filling this gap, providing an environment to enable this systematic comparison between the state-of-the-art systems. However, the thesis's evaluation process covers a full-grown context, such as the mapping from relational databases to KGs. This research field provides sophisticated tools, including RODI [122], which provide a common framework to compare different systems. Moreover, considering the specificity of the semantic model perspective on the mapping problem, an in-depth analysis is conducted comparing SeMi (see Chapter 6) with other tools producing semantic models as output. As mentioned in Subsection 2.2.1, semantic models explicitly shape complex relations, which characterize real scenarios such as the public procurement domain discussed in Section 4.1.

3.1.1 From Structured Data to KGs

The simplest formulation of the mapping consists in annotating the elements of a structured source, such as a CSV, an XML, or a JSON file with semantic tags. These semantic tags include KG and ontology components, such as classes, properties, and entities. In the context of tabular data that include CSV files, the most relevant tasks identified by the proposed Semantic Web challenges[3] require to: (i) assign a semantic type (e.g., an ontology class) to a column; (ii) assign an ontology property to the relation between two columns; (iii) matching a table cell to a KG entity. In the case of tree-structured data that include XML and JSON files, the relevant tasks are analogous to those of the tabular data. However, the detection of the correct KG class or the KG property requires to traverse different tree structures and to identify relations between elements at different levels of the tree.

Currently, much of the available work is focused on tabular data. Most of the approaches exploit the background knowledge for driving the annotation of columns and the inference of the properties between these columns. A seminal approach proposed by Mulwad et al. [112] describes an extensible, domain-independent approach to encode a table as RDF data. Their framework incorporates the semantic knowledge of a background KG in the message passing algorithm adopted for the joint inference in a probabilistic graphical model. Syed et al. [153] describe a technique to automatically-infer the semantic description of a table, employing both table headers and the values stored in the table cells. Similarly to [112], this method exploits Web-based knowledge bases, such as Wikitology [154]. The method introduced by Venetis et. al [165] exploits a wide coverage, but noisy database of class labels and properties automatically extracted from the Web. This

[1]Further information available at: [2]

[3]Further information available at: http://www.cs.ox.ac.uk/isg/challenges/sem-tab/

database is exploited by a reasoning-based system, which is able to identify sufficient evidence for the column annotation, or the property detection, of unseen tables. The technique presented by Kruit et. al [92] exploits a background KG to infer all possible label assignments exploiting a probabilistic graphic model. Such model assigns a score based on the label similarities and then updates this score to maximize the entity coherence across different rows by means of a loopy belief propagation algorithm. The goal of specific approaches is to improve scalability issues. For instance, Takeoka et al. [158] present a computationally-efficient method that employs a probabilistic model for the table annotation, combined with multi-label classifiers. This approach supports the annotation of different types of data, including numerical values. Tableminer+ [181] is an efficient system whose annotation task is supported by various types of contextual information as features for the detection process.On the one hand, it exploits available features both inside and outside tables, and on the other hand it adopts an incremental and iterative process to reach the optimal annotation for the entire table. Ramnandan et al. [128] adopt a different perspective for semantic labeling, that consider an holistic view of the data values corresponding to a semantic label. The goal of this approach is to capture the features of data instances that are related to a semantic type as a whole. Their classifier assigns a semantic label to each attribute of the target data source: the algorithm predicts candidate semantic types by computing the cosine similarity [4] between the TF-IDF[5] vectors of the labeled values in the training data, and the unlabeled values coming from an attribute of a new target source. In SeMi (Chapter 4) the method proposed by Ramnandan et al. [128] is implemented to detect semantic types. This approach has the following advantages: (i) *efficiency and scalability*: the method is about 250 times faster than methods that use other algorithms such Conditional Random Fields; (ii) *accuracy in different fields*: the approach improves accuracy of competing methods on a plethora of diverse sources; (iii) *generality*: the method is agnostic in terms of ontology and schema for the semantic labeling purpose. At the end of the semantic labeling process, SeMi suggests a ranking of semantic labels for each attribute of the source, to be validated by the user or by the domain expert. Promising results come from methods that intend to avoid the adoption of background knowledge to support the mapping process. One of the most recent work in this direction is MantisTable developed by Cremaschi et. al [34], that provides a fully-automatic semantic table interpretation. MantisTable is based on an unsupervised approach that allows to annotate tables, including the labeling of columns and the relations between them, possibly without exploiting

[4]Cosine similarity is a measure of similarity between two vectors, obtained computing the cosine of the angle between them

[5]The term frequency-inverse document frequency (TF-IDF) reflects how important a word is to a document from a collection or corpus

the header row or other external data. Considering the large amount of available knowledge included within Linked Open Data and in other semantically-structured form, the contributions of these thesis exploit semantic information as background knowledge, including SPARQL queries and existing KGs.

The mapping of tree-structured data to KGs is traditionally performed by general-purpose tools, which are also able to transform tabular data into KGs. One of the most prominent tools is Open Refine[166] that allows to clean, transform, and enrich messy data. It enables a semi-automatic reconciliation step against a SPARQL endpoint or any database exposing a Web service, which meets the specifications related to the Reconciliation Service API. The tool provides multiple entity candidates that can be manually refined by the user. Moreover, an ontology class restriction can be applied at the same level of the tree structure to narrow the the number of plausible matches. Other Open Refine functionalities can be enabled installing different extensions. Among general-purpose systems is worth to mention Karma, which enables users to integrate data from a variety of data sources, including tabular and tree-structured data. Karma is able to learn the annotation of the attribute source to ontology classes and then exploit ontologies to propose a semantic model that connect together these classes. This semantic model is then exploited to generate RDF data ready to be published or to be stored in a database. Further details on the approaches integrated in Karma for the semantic model generation are discussed in Subsection 3.1.3.

Recent works begin to explore embedding-based approaches for the mapping purpose. Efthymiou et. al [49] developed and analyzed three different unsupervised methods. The first method exploits the context included in Web tables, to support a lookup-based method to identify the corresponding entity within a KG. The second technique uses Word2Vec embeddings derived from the entity context in a KG to discover Web table entities. Finally, the third method is based on an ontology matching procedure, which combines schema and instance information of entities from both the Web table and the KG. Other research work fully employ neural architectures. Ruemmele et al. [133] developed three different learning-based approaches for the detection of semantic types: a classification model based on a manual feature engineering process and two deep learning models that exploit the Convolutional Neural Network and the Multi-Layer Perceptron architecture respectively. ColNet [31] is a neural network based framework focused on type annotation and entity column detection. ColNet integrates deductive reasoning and machine learning, in order to embed the semantics of a column in the vector space and predict the semantic type with a set of candidate KG classes. These approaches shows that the research work exploring NNs is currently focused on the column annotation with semantic types, while less attention is given to the semantic connections between these annotated classes. The contributions provided by this thesis intends to mainly address the issue of the semantic relation inference.

3.1.2 From Relational Databases to KGs

The generation or the enriching of KGs with data from relational databases (RDB) is a well-known problem in literature and an active field of research. This transformation process, also known as RDB2RDF, intends to make data hosted in relation databases accessible to the Semantic Web. The goal is to convert relational data into RDF or virtually exposing relational data, so they can be queried through the SPARQL language.

Structured data and relational databases present different features. Relational schemas are optimized for specific workflows (and workloads) and the data are distributed among different tables to reach this purposes. Structured data instead tend to be self-consistent and provide to the user the access to unique file containing the data. In some cases, relational databases are optimized for update-intensive workloads and the information is spread over different tables using many-to-many (n:m) relationships. In other cases, the goal is to optimize read-intensive workloads and the data covering different entities tend to be aggregated in the same table of the relational database. Foreign keys, which do not exist in structured data such as CSV or JSON files, act as cross references between tables, pointing to the their primary keys.

The most recent works for the RDB2RDF process include framework for mapping relational database system to RDF, in order to provide advanced semantic query capabilities [118]. Considering the computational perspective, some works are focused on the scalability issues of the mapping process. For instance, the work proposed by Kamal et. al [1] proposes a semi-automatic technique, which obtains valuable performances in mapping data from multiple relational databases. Other works exploit different intermediate artifacts and systems to perform the RDB2RDF. Malik et al. [102] propose a method that employs the XML standard as a transitional language between relational databases and KGs. Yu et. al [179] instead use R2RML, SPARQL, and the Jena framework to conduct a data fusion operation into RDF graphs.

Among the most prominent tools in the field, it is worth to mention BootOX [84] and IncMap [124]. BootOX does not exploits declarative languages, but is based on a direct mapping step[6]: every table in the database is mapped into a class of the ontology, data attributes are mapped on data properties, and foreign keys to object properties. IncMap instead runs in two phases: firstly, it uses lexical and structural matching and, secondly, it represents with a meta-graph the ontology and the schema of the input dataset in order to preserve their structure. It is important to remark MIRROR [105] and D2RQ [23]. Both tools do not necessarily

[6]For more details, see: "A Direct Mapping of Relational Data to RDF", W3C Recommendation 27 September 2012. More information available at: https://www.w3.org/TR/rdb-direct-mapping/.

exploit an existing domain ontology, but they can generate an ontology on-the-fly based on the input data schema. In details, MIRROR produces R2RML mappings exploiting the morph-RDB[7] engine. D2RQ, instead, uses its own native language to define the mappings.

These prominent tools (BootOX [84], IncMap [124], MIRROR [105], and D2RQ [23]) have been included in a benchmark suite known as RODI ([122]). RODI compares the results of the SQL queries performed on relational databases against the results of SPARQL queries performed on KGs generated by the tools. The reasoning behind RODI is twofold: (i) R2RML encourages developers to comply with a standard, however many tools such as D2RQ adopt an internal mapping language; (ii) the mapping process is useful and effective for the specific task at hand. For these reasons, the mapping accuracy can be evaluated considering the results of a query workload posed against the generated KGs. Taking into account the wide range of tools that can be systematically evaluated with RODI, the contribution presented in Chapter 6 of this thesis is included in this benchmark and evaluated against different systems.

3.1.3 Semantic Modeling Systems

The formulation of the mapping as a semantic modeling problem extends the requirements defined by the challenges mentioned in the case of structured data (see Subsection). Semantic types allow to annotate the column of a table or the attribute of a JSON file, combining an ontology class and an ontology data property. However, a full semantic interpretation of the data source requires to understand how the data source attributes are tied. In their influential works focused on the semantic modeling perspective [155] [156] [157] Taheriyan et al. indicate that research efforts in semantic modeling focused so far mainly on the Semantic Type Detection (STD), while less attention has been given to the automatic Semantic Relation Inference (SRI). This trend is also observed in approaches which employ NNs for the mapping purpose (see Subsection). The motivation has to be found in the complexity of the second step: in fact, even when semantic labels are manually annotated, inferring the relations through an automatic mechanism is not trivial and it is still an open issue in research. In addition, in more complex (but not unusual) situations, which are not fully covered by traditional approaches, semantic types can be connected through multiple paths that include different sequences of ontology classes and object properties.

Most of the approaches focused on semantic models exploit a background knowledge to support the mapping process. In many cases, they employ existing semantic

[7]The GitHub repository of the engine is available at: https://github.com/oeg-upm/morph-rdb

models for similar sources, to learn the semantics of the target source. In this context, the work of Vu et al. [168] employs probabilistic graphical models to identify the most plausible semantic model of a data source within a combinatorial space. Among the advantages offered by this approach, the authors mention the robustness against noisy information and a straightforward method for taking advantage of relations within the data. Taheriyan et al. [156] propose a system that exploits existing semantic models and an ontology to build a weighted graph that includes all plausible semantic models for the target source. Then, on the basis of the assigned weight, the system computes a ranked list of candidate semantic models. The main limitation of both approaches is that accuracy is hugely dependent on the availability of semantic models. However, in many domains existing semantic models are not available and manually create them is a very expansive process. Among other approaches proposed in the literature to address the semantic modeling problem, and in particular the relation inference, a promising one is the exploitation of Linked Data (LD) repositories as background knowledge. As demonstrated by the work of Taheriyan et al. [157], the results of this learning process are helpful to select a path representing the correct semantic interpretation of the target source. The main contribution of this thesis to address the mapping problem (Chapter 6) takes inspiration from this work to integrate a novel mechanism for inferring semantic relations using background LD. The most important difference between the thesis contribution and the work of Taheriyan et al. [157] is that the latter adopts a manual extraction of features (e.g., complex graph patterns to represent semantic relations of different lengths), while the approach explained within the thesis automatically learns latent features for entities and properties, encoding them in a vector space. These features are learnt by a Graph Auto-encoder, exploiting the local neighborhood structures within the LD graph.

3.2 Refinement of KGs

The result of building a KG is never be completely accurate. One the one hand, it is not able to reach full coverage and include all possible statements related to a specific entity. One the other hand, some errors risk being introduced during the construction process, preventing the KG from being fully correct. For these reasons, the KG refinement is an active and discussed research field. Paulheim [119] describes three different orthogonal dimensions for classifying KGs refinement approaches. These methods can be distinguished considering their *overall goal*, i.e., correction against completion of the KG; the *refinement target*, i.e., entity types, relations between two different entities, or between an entity and a liter value; the data exploited by the method, which include the KG itself, or other external sources. The link prediction mechanism of the Geranium platform (Chapter 7)

in the academic publications domain can be classified according to these orthogonal dimensions. It is focused on the KG completion task: its goal is to suggest new statements, representing interesting opportunities of collaborations between researchers. As a direct consequence, the target of the completion task is connected to the relation between two entities. Finally, the data exploited by the contribution involve the KG itself, without considering further sources. The refinement employing external sources can be partially framed into the mapping problem of structured sources that is already discussed in Section 3.1. This Section provides an overview of the tools that can be categorized according to the dimensions which characterize the thesis' contribution. A specific focus is dedicated to KG embeddings, which become the state of the art for completions tasks, predicting new links within a KG.

3.2.1 Completion Approaches with KG Embeddings

The completion task aims at increasing the coverage of a KG. In the most recent literature review on KGs [79], three different types of KGE techniques have been proposed to reach this purpose: (i) *linear* or *translational* models, which formulate the relations between two entities as a linear transformation, projecting $\boldsymbol{s}$ into $\boldsymbol{o}$; (ii) *bilinear* or *factorization* models, where $\boldsymbol{r}$ is encoded within a matrix and combine $\boldsymbol{s}$ and $\boldsymbol{o}$ by a multiplication. This multiplication can be seen, from another perspective, as a factorization method to decompose the relational data for the representation learning purpose; (iii) *neural* models, which are more flexible than the previous techniques, due to the multiple parameters enclosed in their network structure.

Translational models encode statements as transformation from subject nodes to object nodes by means of the labeled edge. The most simple formulation is represented by TransE [24], which learns the embeddings of $\boldsymbol{s}$, $\boldsymbol{r}$, and $\boldsymbol{o}$, making sure that $\boldsymbol{s} + \boldsymbol{r}$ is close as possible to $\boldsymbol{o}$. On the contrary, if the edge is negative example, TransE tends to consider $\boldsymbol{o}$ away from $\boldsymbol{s} + \boldsymbol{r}$. Many variants of TransE have been proposed to improve this simple formulation. TransH [173] intends to distinguish different relations employing different hyperplanes. As a result, $\boldsymbol{s}$ is initially projected onto the hyperplan of $\boldsymbol{r}$ before considering the translation to $\boldsymbol{o}$. A generalization of this approach is provided by TransR [99], in which $\boldsymbol{s}$ and $\boldsymbol{o}$ are both projected into a general vector space related to $\boldsymbol{r}$. TransD [83] associates entities and relations with secondary vectors, which are then exploited to project the entity into a relation-defined vector space.

The main idea that unifies all factorization methods is based on the fact that a tensor can be *decomposed* into tensors of lower orders, also known as *factors*. In this perspective, these factors are able to capture latent features underlying the information encoded in the original tensor. Therefore, the original tensor can be approximately recomposed through a sequence of basic operations, including

bilinear maps. In the context of KGs, such factors are represented by entity and relational embeddings. The seminal method to compute these embeddings with a factorization approach is represented by DistMult [178]. In this specific case each entity and relation is associated with a vector and the related scoring function computes the following operation: $\boldsymbol{s} * \boldsymbol{r} * \boldsymbol{o}$. The goal of the learning process is to maximize the probability of existing edges and to minimize the probability of non-existing edges. Other approaches such as RESCAL [115] replaces the relation vector with a matrix in order to exploit a multiple dimension to encode the direction of the edge. HolE [114] introduces the so-called *circular correlation operator* in order to combine entities and relation vectors. As happen in the case of RESCAL, this operator is able to capture the direction of the relation. In regards to ComplEx [163], entity and relation vectors are represented using complex numbers. As a consequence, the number of parameters to learn is maintained low and the model is able to compute the direction of the edge.

Neural models are able to learn embeddings adopting non-linear scoring methods to compute the probability of a statement. Neural Tensor Networks (NTNs) [147] is one of the initial contributions based on neural networks. The goal in this case is to compute the outer product between $\boldsymbol{s}$ and $\boldsymbol{o}$ and combine the result with a network layer representing $\boldsymbol{r}$, in order to compute the plausibility score. Most recent approaches involve the adoption of convolutional filters within the models. ConvE [39] intends to produce a matrix, concatenating the vector representations of $\boldsymbol{s}$ and $\boldsymbol{r}$. This matrix is provided as input of multiple convolutional layers of 2 dimensions, which are able to return a feature map tensor. Such tensor is transformed employing a linear transformation and the final score is achieved performing a dot product between the resulting vector and $\boldsymbol{o}$.

Chapter 4

Application Scenarios

This Chapter describes the main features of two main application scenarios, which are characterized by the research problems discussed within this thesis. The first scenario is represented by Open Government Data (OGD) released by Italian Public Administrations (PAs) in the context of public procurement. In this specific situation, each PA is responsible for publishing information related to its public contracts, following specific legal guidelines. However, considering the fragmentation of the systems and the processes adopted by different PAs, it may happen that data sources with different structures are published. As a consequence, tools for the automatic generation of semantic models can be incorporated into a pipeline for the generation of KGs from structured sources. The analysis of this KG and the adoption of its related technologies can enable to detect and partially resolve quality data problems related to inconsistency issues. The second scenario is related to the Open Data (OD) in the context of academic publications released by the Politecnico di Torino (Polito). The available search engine[1] built on top of such information does not allow to explore the implicit, but useful connections between researchers. Building a search engine based on a KG integrating publications data enables advanced search features and allows us to predict new links that can be interesting for recommendation purposes.

4.1 Integration Public Procurement Data into a KG

Public Procurement (PP) information, made available as OGD, leads to tangible benefits to identify government spending for goods and services. Nevertheless,

[1]IRIS: https://iris.polito.it/

making data freely available is a necessary, but not sufficient condition for improving transparency. Fragmentation of OGD due to diverse processes adopted by different administrations and inconsistency within data affect opportunities to obtain valuable information. This Section describes a solution based on KGs and linked data to integrate existing datasets and to enhance information coherence. It presents an application of such principles through a semantic layer built on Italian PP information available as OGD. As result, the approach overcomes the fragmentation of data sources and it increases the consistency of information, enabling new opportunities for analyzing data to fight corruption and for raising competition between companies in the market.

This Section has the following structure. Subsection 4.1.1 reports a domain analysis in the field of public procurement and spending information published according to linked data principles. Subsection 4.1.2 describes the context and the role of PP information and gives an overall view of public data made available by Italian administrations. Subsection 4.1.3 explains current problems in terms of data quality of such data. Subsection 4.1.4 illustrates the approach for processing, transforming, and publishing procurement information as linked data. Subsection 4.1.5 reports results of the analysis on data quality issues. Subsection 4.1.6 presents a discussion on obtained results.

4.1.1 Analysis of the Domain

This Section reports contributions of procurement and spending data transformed and published according to linked data principles. This domain has already been addressed by several research projects, however a comprehensive work on Italian procurement data is not addressed yet. Furthermore, at the best of our knowledge, an analysis on procurement data consistency exploiting semantic technologies to improve transparency has not yet been accomplished.

One of the most important contributions in this domain is the LOD2 project, since it systematically addresses many phases of procurement linked data processing [152]. There are several other notable initiatives: the TWC Data-Gov Corpus [41], Publicspending.gr [106], The Financial Transparency System (FTS) project [103], Linked Spending [78], LOTED [164] and MOLDEAS [6].

In particular, the TWC Data-Gov Corpus gathers linked government data on US financial transactions from the Data.gov project[2]. This project exploits a semantic-based approach in order to incrementally generate and enhance data via crowdsourcing. Publicspending.gr has the objective of interconnecting and visualizing Greek public expenditure with linked data to promote clarity and enhance citizen awareness through easily-consumed visualization diagrams. The FTS project of the

[2]Data.gov project website: https://www.data.gov/

European Commission contains information about grants for EU projects starting from 2007 to 2011, and publishes such data as linked data. Exploring this dataset, users are able to get an overview on EU funding, including data on beneficiaries as well as the amount and type of expenditure. Linked Spending is a project for the conversion to linked data of information published by the OpenSpending.org, an open platform that releases public finance information from governments around the world. The project uses the DataCube vocabulary[3] to model data in order to represent multidimensional statistical observations. LOTED[4] is focused on extracting data from single procurement acts and aggregating it over a SPARQL endpoint. Finally, MOLDEAS presents some methods to expand user queries to retrieve public procurement notices in the e-Procurement sector using linked data.

4.1.2 Study Context

This Subsection presents an overview on PP data, a description of the Italian legislative context according to which procurement data is published by public bodies, and an analysis of the key characteristics of such data.

Public Procurement as Open Data

Open Data on PP, namely the procurement of goods or services on behalf of a public authority, is a specific area of the OGD characterized by big potential for increased openness of government information and incentives for supporting business activities. As reported by the Organisation for Economic Co-operation and Development (OECD), around US$ 9.5 trillion of public money is spent each year by governments procuring goods and services for citizens[5]. Furthermore, PP transparency is a crucial toolset to identify problems that arise from corruption, promoting competition and growth: according to the Transparency International Slovakia initiative[6] "reforms in procurement that included contract publication led to an increase in bids from an average of 2.3 per public tender in 2009 to 3.6 in 2013". In other words, as argued by Svátek [152], PP information is able to unify *public* needs and *commercial* offers: it enables a lively context to increase the interoperability between data models[7], methodologies, and sources independently

[3]DataCube vocabulary information: https://www.w3.org/TR/vocab-data-cube/

[4]LOTED project website: http://www.loted.eu/

[5]More details available in the OECD blog post "Transparency in public procurement, moving away from the abstract": http://oecdinsights.org/2015/03/27/transparency-in-public-procurement-moving-away-from-the-abstract/

[6]For more information, see: http://www.transparency.sk/

[7]The ISA initiative of the European Commission represents a landmark for understanding different levels of data interoperability. More information available at http://ec.europa.eu/

designed within the two sectors.

Despite the tangible benefits that come with the publication of PP information as OGD, making data available does not automatically produce transparency. As underlined by Janssen [82], providing data alone is not sufficient: deep insights into the working of mechanisms to ensure that information can be easily accessible, processed, and interpreted are necessary to create transparency. Such reflections emerge from a conceptual framework called Big and Open Linked Data (BOLD), proposed by Janssen himself, that identify categories, dimensions, and sub-dimensions that influence transparency [82].

An obstacle to a comprehensive implementation of transparency through OGD is related to the fragmentation of existent open government datasets, in particular in the domain of procurement data[8]. KGs and linked data principles can be a *modular* and *scalable* solution to overcome the fragmentation in government data, increasing citizen awareness of government functions and enabling administrations to work more efficiently.

The Italian Legislative Context

The Italian Legislative Decree n. 33/2013 (DL33/2013) of March 14th, 2013[9] re-ordered obligations of disclosure, transparency, and dissemination of information by public administrations. According to specific requirements defined by the decree (Article 9 - DL33/2013), each body is required to create a specific section on its website called "Amministrazione Trasparente" (Transparent Administration). In this section, administrations provide details related to public procurement, with particular emphasis on procedures for the award and execution of public works, services, and supplies (Article 37 - DL33/2013[10]). Such data is published on the

---

isa/documents/isa_annex_ii_eif_en.pdf

[8]Only 1/5 of total public expenditure on goods and services is published with rules complying with the EU Directives, for an estimated value of €420 billion. It means that "the bulk of total public expenditure on goods, services, and works is not organised in accordance with EU procurement legislation". See http://ec.europa.eu/internal_market/publicprocurement/docs/modernising_rules/executive-summary_en.pdf

[9]See: http://www.normattiva.it/uri-res/N2Ls?urn:nir:stato:decreto.legislativo:2013-03-14;33!vig=. Notice that DL33/2013 has been recently amended by D.Lgs. 25 maggio 2016, n. 97 (DL97/2916), with a general tendency toward a more centralized publication of data - see, in particular, Article 9-bis - but no immediate impact on the publication requirements discussed in the paper at hand. See below footnote 10 for additional comments. Last visit on Nov. 2016

[10]Following the aforementioned amendments by DL97/2016, a National Public Contracts Data Base -the BDNCP- is forthcoming, as described in the new Annex B of DL33/2013, but its creation will face all the problems described in the paper at hand - and possibly benefit from the suggested approach

basis of a precise XML Schema Definition[11] (XSD) provided by *ANAC - Autorità Nazionale Anticorruzione* (Italian National Anti-Corruption Authority)[12], which has supervisory duties. After the publication on their websites, administrations transmit the link of the dataset to ANAC via certified mail. ANAC, at this point, performs a preliminary check and releases an index file (in JSON format), containing details related to the availability of data[13].

Source Data

Public bodies can publish and transmit to ANAC two types of XML files. The first type contains the actual data on contracts until the publication date (January 31st of each year). In order to facilitate the consistency of publications and the comparison of information, the structure of the document is defined by a precise XSD Schema[14]. The main structure of the XML file includes a section with the dataset metadata and a section containing multiple contracts, each of whom can be identified by the XML tag "lotto". The metadata section lists some information, including the first publication date and the last update of the dataset, the business name[15] of the contracting authority that spreads the dataset, the url of the dataset, and the license. The section containing data on contracts includes the following information: the identification code of the tender notice or CIG (that stands for Codice Identificativo Gara), the description of the tender, the procedure type for the selection of the beneficiary, the identification code and the business name of bidders (tender participants), the identification code and the business name of the beneficiary, the awarded amount, the paid amount, the dates of commencement and completion of works.

The second type of XML, instead, is an index that collects links to other XML files containing actual public procurement data[16].

---

[11] XSD is a W3C recommendation that specifies how to describe an XML document

[12] See: http://www.anticorruzione.it/

[13] The JSON index is available at https://dati.anticorruzione.it/l190, by clicking on the "Esporta" (Export) button

[14] A representation of the XSD schema is available at http://dati.anticorruzione.it/schema/datasetAppaltiL190.xsd

[15] A pseudonym used by companies to perform their business under a name that may differs from their legal name

[16] A representation of the XSD schema is available at http://dati.anticorruzione.it/schema/datasetIndiceAppaltiL190.xsd

4.1.3 Data Quality Problems

As described in Section 4.1.2, public contracts information is generated and spread on Italian public bodies websites. Due to diverse processes and tools adopted by administrations, the quality of PP data is extremely variable depending on the single case and it is impaired in terms of *accuracy*, *completeness*, and *consistency*[17].

Accuracy is defined as *the degree to which a data value conforms to its actual or specified value.* This metric is divided in syntactical accuracy and semantic accuracy, which are defined in the following way:

- *Syntactical accuracy is defined as the closeness of the data values to a set of values defined in a domain considered syntactically correct.*
- *Semantic accuracy is defined as the closeness of the data values to a set of values defined in a domain considered semantically correct.*

The definition of completeness is dependent on the perspective used:

- Computer system's point of view: *completeness is the extent to which all necessary values have been assigned and stored in the computer system.*
- End-user point of view: *completeness is the extent by which the data consumer's need is met.*

Consistency *refers to the absence of apparent contradictions within data.* Inconsistency can be verified on the same or different entities. In the context of XML data that refers to a schema, integrity constraints are properties that must be satisfied by all instances of a database schema. Although integrity constraints are typically defined on schemas, they can at the same time be checked on a specific instance of the schema that presently represents the extension of the database.

Interdependence between quality metrics

Although there are different metrics to assess the quality of data as shown in the previous Subsection, such metrics are closely interdependent. In the procurement domain, for instance, a contract could present issues like bad comma position in a payment value (accuracy) or the lack of the payment field (completeness). Both errors have a direct impact on the consistency of information: contradictions inevitably occur when it is analyzed the total amount of expenditure resulting by several XML files that report data of an ongoing contract.

For these reasons, although the focus of the study concerns the consistency of the information, Subsection 4.1.5 proposes also a comprehensive analysis of the data quality in terms of accuracy and completeness.

[17]Such data quality metrics are defined by the International Organization for Standardization: ISO/IEC 25012

Focus on Data Consistency

Certain types of consistency problems directly emerge analyzing contracts data collected in a single XML file. Here are reported 3 examples of such inconsistencies:

- contracts in which the beneficiary is more than one;
- contracts in which the amount of money is paid, but no recipient is present in the data;
- contracts in which the sum reported as paid is greater than the sum initially awarded to the beneficiary.

Other types of inconsistencies manifest themselves only after merging data contained in different sources. The following cases are real examples of inconsistencies with Italian PP data:

- business entities with more than one business name;
- CIGs that identify more than one contract;
- incoherent payments among different versions of an ongoing contract.

The aforementioned issues represent a significant barrier to achieve transparency, because the results obtained by querying the dataset are likely to be inconsistent and misleading. For example, consider a citizen trying to access the effective business name of a contracting authority identified by the id "00518460019" (this is VAT number of the "Politecnico di Torino"). If the data quality in terms of semantic accuracy is poor, such id could be associated with wrong business names like "Politecnico di Milano" and/or "Politecnico di Bari". So, when the citizen performs a search using the business name as search key, he obtains an inconsistent result. The same problem happens when he wants to get details of a contract identified by a CIG: in this case, he is likely to get discordant values. Moreover, incoherent values of payments are not deductible from a single XML file, because errors emerge by analyzing the evolution of the contract data published in different years.

4.1.4 Applying the KG and the Linked Data Approach

In this Section all stages of the approach to publish Italian PP according to linked data principles are shown. Each stage is accomplished by means of different software components and resources that are shown in Figure 4.1.

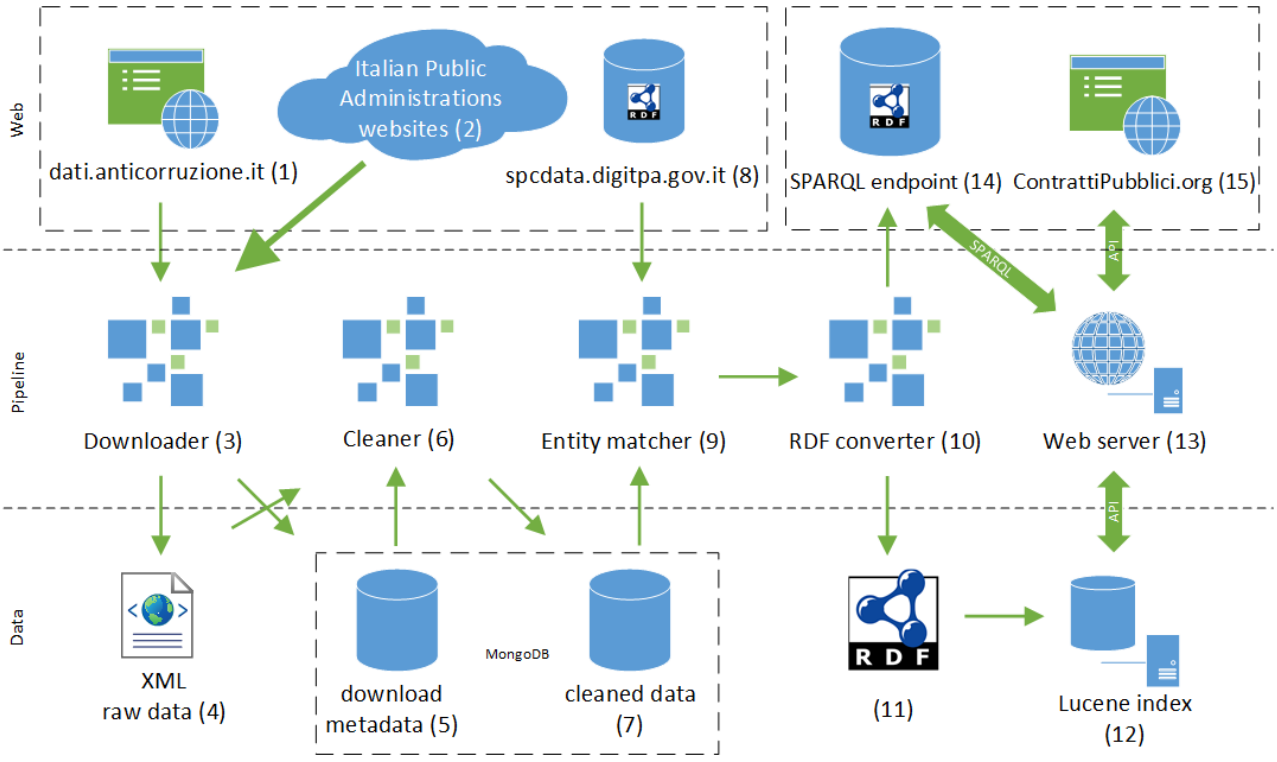

Figure 4.1: Architecture for processing and publishing Italian PP as linked data

Harvesting of XML Files

The task of data harvesting is assigned to the Downloader component (3), that exploits the index file provided by ANAC[18] (1). On the basis of URLs contained in this file, the data fetching process from public bodies' websites (2) is able to manage two different cases. In the first case, the component gets XML files containing PP data and store them locally (4). Download metadata and the local path of each XML are stored in a MongoDB NoSQL database (5), in order to facilitate accessibility to such information and keep track of any duplicate file. In the second case, in which the XML contains links to other XML files, the component is able to cross the links chain[19] and performs the download process shown in the first case. When the component is not able to recognize the expected XML schema (as actual procurement data or as index) the file is stored in a dedicated directory of the file system for a later manual check. This problem occurs when the resource is not published according to an accepted format (e.g., it is a PDF file). In the

[18]The index file provided by ANAC is available at http://dati.anticorruzione.it

[19]In some cases an index points to another index that finally might point to a file, or to another index

worst cases, XML indexes are recursive, since they contain URLs that reference to the XML index itself. For these reasons, some features are implemented in order to manage this critical issue that threatens to undermine the entire pipeline. Moreover, during the download operation, a lot of servers do not reply: more than 10 different HTTP responses have been collected, which reveal how the quality of service over the 15,000 infrastructures of the Italian public administration might not be reliable. Results of the download process are reported in Section 4.1.5.

Cleaning of Procurement Data

The next step to the data harvesting is performed by the Cleaner (6). During this stage procurement information is extracted from XML files and each contract is processed and stored as unique document in an instance of the MongoDB database (7). Analyzing such data the magnitude of data quality in terms of accuracy, completeness, and consistency is evaluated (results of such evaluation are available in Section 4.1.5).

When new errors are detected, the Cleaner component is progressively improved. Every time a specific fix is applied on data regarding to a contract, new metadata are included into the related MongoDB document: the original data is preserved and a specific field called "errors" is compiled with the identified issue. For instance, encountering a bad format for the date value (dd-mm-yyyy), such value is transformed in the correct format according to ISO 8601 (in our case yyyy-mm-dd), preserving the original data and saving in the "errors" field the following string: "bad date format" (Section 4.1.5 reports adopted solutions for the most common procurement data quality issues).

Public Contracts Ontology

In order to publish Italian public procurement according to linked data principles, the Public Contracts Ontology (PCO) is adopted. This ontology is developed in the context of the Czech OpenData.cz initiative[20]. According to its authors, this ontology describes "information which is available in existing systems on the Web" and "which will be usable for matching public contracts with potential suppliers" [42]. Therefore, the goal of the PCO is to offer a generic model for describing public contracts, without providing details of the public procurement domain, that are specific to fields and countries.

In the PCO domain, a call for tenders is submitted for the award of a public procurement contract. Therefore, XML fields and data described in Section 4.1.2 are mapped into entities, classes, and relations provided by the PCO. Figure 4.4

[20]The Public Contracts Ontology is available on GitHub platform at: https://github.com/opendatacz/public-contracts-ontology

shows the data model adopted for publishing Italian PP as linked data. Although there is a significant degree of overlap between the XSD that describes the schema of source data and the PCO, additional elements are introduced to better describe our domain. For instance, the concept of tender was not fully expressed in the data model adopted in XML files, since there are only information about participants, but not details related to offering services and prices. Nevertheless, the tender is one of the most important entity in the PCO to link participants to the public contract. For these reasons, during the conversion to linked data (Section 4.1.4), tender entities are created using as identifier the id code of the participant and the CIG of the contract.

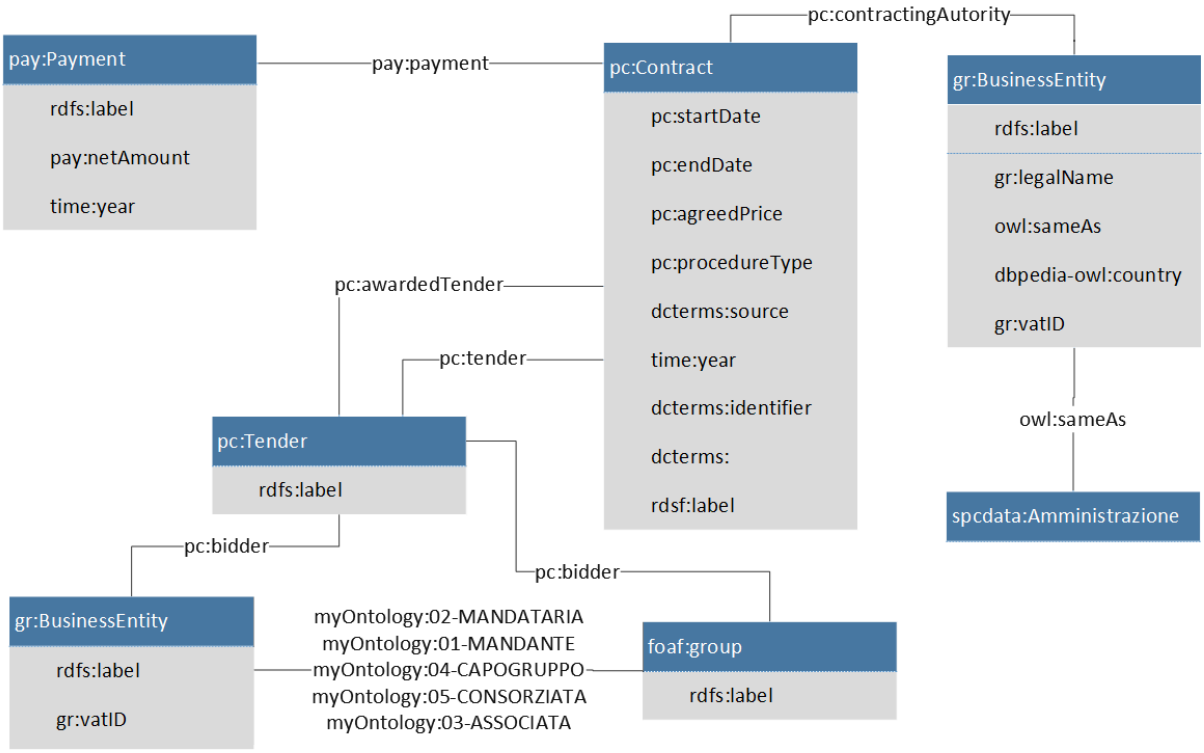

Figure 4.2: Schema that describes Italian PP in the linked data domain

Triplification and Interlinking

After the cleaning stage, contracts data stored in the MongoDB instance are converted into RDF using the N-Triples serialization[21]. The component that performs this task is called RDF converter (10), that maps fields and data values of the contracts into properties, entities, and literals defined by the PCO.

[21]More information available at: https://www.w3.org/TR/n-triples/

The triplification process is a trivial task for contracts extracted from XML files that are compliant with the schema defined by the Italian law (see Subsection 4.1.2). However, not all downloaded XMLs are valid, due to errors in the management and the publication of data. To address this issue, SeMi has been exploited to integrate potential valuable data from sources with invalid schema. A detailed description of the results of SeMi in this application domain are reported in Chapter 6.

Before completing the triplification process, the Entity matcher component (9) performs the so-called *interlinking* stage. For the application, to improve the consistency of the information, public bodies listed in Italian PP are interlinked to public bodies gathered from the SPCData database[22], provided by the *Agenzia per l'Italia Digitale* (8), that contains the index of Italian public administrations. The matching between entities are created using the identification code contained in both datasets. After the interlinking step, the final RDF file (11) is pushed into a Triple Store that exposes data via a SPARQL endpoint (14).

As shown in Figure 4.1, the RDF file is also published in a Lucene[23] index (12) to enable full-text search features and data published within the endpoint can be queried by a Web server (13) to populate a Web interface[24] (15).

4.1.5 Application Results

This Section reports results obtained with stages described previously in order to reduce data fragmentation and to improve data quality, in particular the consistency of information. Moreover, this Section reports preliminary results achieved introducing the GAE, compared to results obtained with the initial steiner tree detection, for the task of semantic relation inference.

Harvesting Results

With the approach described in Section 4.1.4, information coming from more than 300,000 XML files published by 15,000 public bodies are integrated. Table 4.1 reports details of the harvesting phase (see Section 4.1.4) that was accomplished in 4 different periods, starting from May 2015[25]. The download process has been carried out at different times for two reasons. The first reason is that ANAC releases the index file containing references to XML files in February of each year. The second reason is related to problems about servers uptime, which inevitably impacts on the

[22]More information available at: http://spcdata.digitpa.gov.it/index.html

[23]Apache Lucene is a high-performance, full-featured text search engine library written entirely in Java. More information available at: http://lucene.apache.org/core/

[24]The ContrattiPubblici.org project developed by Synapta Srl aims at demonstrating the opportunities for transparency and business of public procurement spread according to linked data

[25]Some information is missing because in 2015 the number of requested URLs were not stored

availability of XML files. In order to tackle the last problem, on each harvesting cycle the download of all XMLs is performed, generating the duplication of files, that are managed during the data cleaning process.

Table 4.1: Number of downloaded XML files of procurement data in different periods of time

	May 2015	**Nov 2015**	**Feb 2016**	**Nov 2016**
URL requested	-	-	207.674	271.664
downloaded files	205.415	184.738	201.451	252.246
valid XMLs	199.341	180.609	197.338	247.881

Quality Problems Addressed

During the cleaning phase of PP (see Section 4.1.4), each contract is stored as single document within a MongoDB instance, enabling a first level of analysis on quality issues in terms of accuracy, completeness, and consistency of data. Table 4.2 shows, for each field of the contract, the type of data quality issue, the occurrence of such issue (in percent), the adopted solution (where available). The 41,65% of all contracts (almost 6 million in total) presents at least one of these issues. The analysis of data inconsistency issues mentioned in Subsection 4.1.3 shows that contracts in which the beneficiary is more than one correspond to 1.78% of all contracts; contracts in which the amount of money is paid, but no recipient is present in the data correspond to 4.30% of all contracts; contracts in which the sum reported as paid is greater than the sum initially awarded to the beneficiary correspond 5.96% of all contracts.

Exploiting semantic principles, a KG on procurement data is created to reduce fragmentation and to identify further inconsistencies. The dimension of the dataset built according to linked data principles is available in Table 4.3. Such dataset is published using the *Virtuoso Triple Store*[26] and can be queried via SPARQL endpoint[27].

4.1.6 Discussion on Inconsistency Issues

As explained in Subsection 4.1.3, there are some inconsistencies that are visible only after completing a data integration process, that includes manual approaches and automatic techniques. Three different cases are reported in this study:

[26]More information available at: https://virtuoso.openlinksw.com/

[27]The SPARQL endpoint on public procurement data is available at: https://contrattipubblici.org/sparql

Table 4.2: Accuracy, completeness, and consistency degree in PP data

Field	Error	Occ. (%)	Solution
Completeness			
Start date	missing	12.25	nothing
End date	missing	21.61	nothing
Agreed price	missing	0.06	nothing
Payment	missing	0.20	nothing
Procedure type	missing	0.11	nothing
Business Entity ID	missing	1,05	hash value
Accuracy			
Identifier	syntactic errors	0.96	string cleaned
	semantic errors	5.83	hash value
Start date	semantic errors	1.36	nothing
End date	semantic errors	2.00	nothing
Agreed price	syntactic errors	0.94	string cleaned
	semantic errors	0.23	nothing
Payment	syntactic errors	0.76	string cleaned
	semantic errors	0.65	nothing
Procedure type	syntactic errors	2.81	optimal string match
Business Entity ID	semantic errors	1,08	hash value
Consistency			
Start date	non standard format	5.63	uniformed to ISO 8601
End date	non standard format	5.20	uniformed to ISO 8601
Beneficiary	more than one beneficiary	1.78	nothing
Payment	payment without winner	4.30	nothing
	greater than awarded price	5.96	nothing

Table 4.3: Characteristics of KG built integrating Italian procurement information

Dimension	Value
RDF triples	168,961,163
entities	22,436,784
contracts	5,783,968
public bodies	16,593
companies	652,121
links to external datasets	13,486

- business entities with more than one business name;
- CIGs that identify more than one contract;
- incoherent payments among different versions of an ongoing contract.

The first of this case can be detected with the SPARQL query available in Listing 2:

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX gr: <http://purl.org/goodrelations/v1#>

SELECT (COUNT(DISTINCT ?be)) WHERE {
  {
   SELECT DISTINCT(?be) WHERE {
    ?be rdfs:label ?label .
    ?be a gr:BusinessEntity .
   }
   GROUP BY ?be HAVING (count(*)>1)
  }
}
```

Listing 2: SPARQL query SQ4 to retrieve business entities with more than one business name

This issue can be fixed exploiting the most important feature of linked data, namely the interlink among different datasets. In fact, a unique business name on a subset of business entities (in our case the contracting authorities) is obtained building links to the Italian public administration index of SPCData, shown in Section 4.1.4. From this dataset, exposed as linked data, the official business name of contracting authorities are retrieved, using as primary key their identification code (in our domain the VAT number of the contracting authority). In this way, the consistency of information is improved for a subset of business entities, enabling the opportunity to obtain valuable results.

The second case consists in the duplication of CIG for different contracts and can be detected with the SPARQL query available in Listing 3:

The solution to this issue is generating a hash value, avoiding ambiguity due to duplicate CIGs, to build contracts URIs. In this way, different contracts, misidentified by the same CIG, are separated in different entities: the URI is created through a hash value generated combining the identity code of the contracting authority, the awarded amount, and the procedure type mentioned in the contract. The user is therefore able to detect this kind of error and he can semantically distinguish different contracts identified by the same CIG. Nevertheless, more context information is necessary to establish which is the correct CIG attribution for a specific contract.

The last problem is tracking incoherent payments published in different XML files of ongoing contracts. This problem is currently not solvable with the current

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX gr: <http://purl.org/goodrelations/v1#>

PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX pc: <http://purl.org/procurement/public-contracts#>

SELECT (COUNT(DISTINCT ?contract)) WHERE {
  {
   SELECT DISTINCT(?contract) WHERE {
    ?contract dcterms:identifier ?CIG .
    ?contract a pc:Contract .
   }
   GROUP BY ?contract HAVING (count(*)>1)
  }
}
```

Listing 3: SPARQL query SQ5 to retrieve CIGs that identify more than one contract

approach, because of the lack of information on provenance and the reliability of different sources.

4.2 Semantic Search and Recommendation Systems in Academic Publications

This Section presents Geranium [167], a semantic platform to collect and organize the scientific knowledge of the Politecnico di Torino (Polito). The research achievements obtained with Geranium are: (i) a novel academic Knowledge Graph (KG) that semantically connects information on researchers and publications of Polito; (ii) a semantic search engine that aggregates such information and enables advanced features for the content exploration; (iii) a recommendation system which suggests, for instance, novel collaboration opportunities between researchers of different disciplines, who worked on the same topic.

The remainder of this Section includes the following parts. Subsection 4.2.1 presents an analysis of the semantic KGs in the academic publication domain. Subsection 4.2.2 describes the main components of the Geranium architecture, in regards to the KG generation and the content exploration and visualization. Subsection 4.2.3 includes details on the implementation of each module of the Geranium architecture.

4.2.1 Analysis of the domain

In the last years there has been a growing interest in scientific KGs, both from academic institutions and private organizations. Some of the most recent scholarly KGs are Semantic Scholar [7], CORE [90], AIDA [9], and OpenCitation [120]. In regards to the most used domain ontologies in the domain, it is worth to mention SWRC[28], BIBO [29], BiDO [30], SPAR [121], CSO [139], SKGO [**fathalla2020**].

A notable example is the Open Academic Graph[31] (OAC), built by merging together the Microsoft Academic Graph[32] and AMiner[33]. OAC has been publicly released to allow the study of citation networks and paper contents. Released in 2017, its first version has been obtained linking the matched publications from the above-mentioned KGs, collecting more than 3 hundred millions of publications. At the beginning, OAC included only the authors and the journals as entities, while the other metadata have been collected as data attributes. In January 2019, the second version of OAC has been released, including further publications. However, the biggest change of this new version was the inclusion of authors and venues as entities, instead of being data attributes of the publication. Unlike the PKG, the OAC does not contain publication topics as entities, because the keywords chosen by the authors suffer of the same limitations of IRIS: they are not explicitly related to semantic topics.

Among the tools similar to Geranium, which employ KGs to build a search engine on scholarly data, the Wiser research project [32] is noteworthy. Wiser is a search tool developed by the University of Pisa and publicly released at the beginning of 2019. The KG adopted by Wiser is composed of approximately 1.500 authors, 65.000 publications, and 35.000 topics. The system has proven to be particularly effective, representing a strategic tool and being actively used by the Technological Transfer Office of the University of Pisa to easily find expertise profiles in a given research field.

One of the main components of the PKG pipeline is TellMeFirst (TMF) [132], a system used to automatically extract the scientific topics from the publications abstracts. By automatically extracting the topics, TMF allows to add them as entities in the PKG, so that each publication is directly linked to its main topics, and each topic is linked to all the publications of which it is a subject. Other tools, such as the CSO Classifier [140], are able to extract the subjects of a publication.

[28] http://ontoware.org/swrc

[29] http://bibliontology.com

[30] http://purl.org/spar/bido

[31] https://www.openacademic.ai/oag/

[32] https://academic.microsoft.com/

[33] https://www.aminer.cn/

This Classifier is able to automatically classify a research paper according to the Computer Science Ontology[34] (CSO), an automatically generated ontology of research topics in computer science. The fact that the CSO Classifier relies on a predefined ontology has some disadvantages with respect to TMF. For instance, the Computer Science Ontology is restricted to the computer science field, while TMF uses DBpedia as its source of knowledge and it is able to extract topics regarding different research fields. However, the approach of CSO Classifier has also its advantages: considering that the ontology is more focused on a specific domain, the classification could be more accurate, and the structure of the ontology itself may be tailored for such classification task.

4.2.2 Approach and Methodology

The approach adopted for the development of the Geranium platform is based on a pipeline architecture. This architecture, available in Figure 4.3, is composed of the following modules:

1. The *Builder*: it creates an initial version of the Polito KG (PKG), which includes: (i) basic metadata on scientific publications and researchers; (ii) semantic topics, which are automatically extracted from the publication abstracts.

2. The *Enhancer*: it builds the training dataset and exploits Graph Neural Network (GAE) techniques to predict unseen facts within the PKG.

3. The *Viewer*: it allows to query, visualize, and explore the PKG built in the previous steps.

The Builder

The *Builder* takes as input the data from IRIS[35], which is the system adopted by Polito to store and spread the scientific papers written by its researchers. The database dump adopted to build the PKG includes information on publications that are released in a period of ten years, from 2008 to 2018. The goal of the Builder is to translate the data contained in the dump in a set of semantically coherent RDF facts. To reach this purpose, an ontology that describes the domain of the IRIS scholarly data has been defined, exploiting existing vocabularies available on the Web. The Builder analyzes each record in the JSON dump and maps the information contained in the record with the concepts and the properties

[34] https://cso.kmi.open.ac.uk/home

[35] https://iris.polito.it/

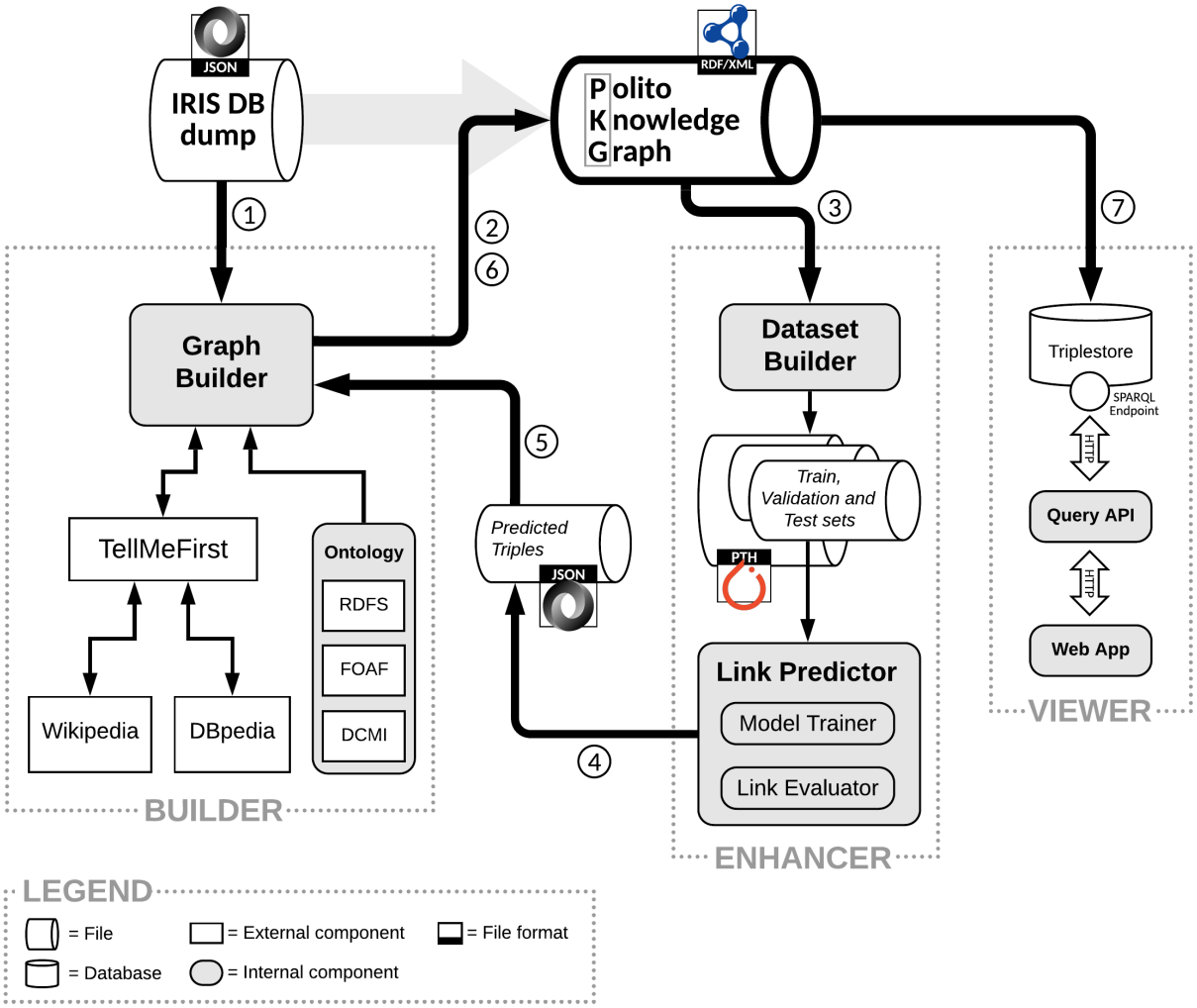

Figure 4.3: Schema of the pipelined software architecture developed to build, enhance and visualize the PKG. The legend is showed in the bottom left corner of the Figure. The circled numbers are used to identify the steps in the pipeline.

defined by the ontology. For instance, considering a publication including more than one contributor, the ontology allows to distinguish the main author and the other co-authors employing different predicates (or object properties). The Builder exploits also automatic techniques to extract semantic topics from the publication abstracts. To reach this purpose, the Builder exploits TellMeFirst (TMF) [132], a tool for the automatic extraction of semantic topics from texts. These topics are uniquely identified by URIs derived from the DBpedia KG. DBpedia is a research project that includes facts that are semi-automatically extracted from Wikipedia [96]. The semantic topics extracted with TMF are add as new entities in the PKG and linked to the publications. The output of the Builder is an initial version of the

PKG, which includes a coherent description of uniquely identified and semantically connected authors, publications, journals, and semantic topics.

The Enhancer

The Enhancer has the goal to prepare the dataset for the the training phase, performed by the GAE, in order to predict new facts withing the KG. The Enhancer includes three main components: (i) the Dataset Builder, (ii) the Model Trainer, (iii) the Link Evaluator.

The Dataset Builder is in charge of translating the PKG into a usable dataset for the Model Trainer, because the RDF facts can not be directly used as input data for the link prediction mechanism. Therefore, the dataset is splitted into three disjoint[36] sets: (i) the training set, (ii) the validation set, (iii) the test set.

The Model Trainer is a GAE that uses these three sets to train, validate, and test the link prediction model. The training set is used to train the model at each epoch, while the validation set is used to evaluate which model parameters to keep, identifying the best epoch. The test set is used to evaluate the accuracy of the model, loaded with the best epoch parameters, upon unseen facts.

The Link Evaluator loads the best model found during the training phase and uses it to evaluate unseen facts, according to a specific score. The predicted facts which can be added to the PKG must comply two specific constrains: (i) they receive a high score from the link prediction mechanism; (ii) they are semantically correct according to the domain and the range defined by the ontology for the object property included in the fact. These facts are added to the PKG, in order to obtain an enhanced version of the PKG. This enhanced version includes missing topics, similar author or journal profiles, that can be used to empower the recommendations for the Polito researchers.

The Viewer

The Viewer allows to explore the scientific publications of PoliTO and to visualize suggested recommendations for Polito researchers. The Viewer is composed of a triplestore, an API layer, and a Web application. The triplestore stores the PKG and exposes a SPARQL endpoint. The API layer allows to retrieve the information contained in the KG and translates the response obtained from the triplestore in a JSON file, which is sent back to the client. The Web application is the entry point for the user, implementing the functionalities of a modern search engine. It allows to query and visualize the data contained in the KG through the use of a friendly interface. The user can search for a topic of interest and obtain as result the list of all the publications, authors and journals that are linked to such topic. The

[36]Two sets are said to be disjoint if they have no elements in common.

predictions are used to add suggestions to the results: searching for an author, the Web application shows its actual research interests, but also other topics in which she may interested in, or researchers with similar profiles who have never been her co-authors.

4.2.3 Implementation Details

This Section reports the implementation details of the Geranium architecture.

Building the initial PKG

The goal of the Builder (Subsection 4.2.2) is to create the PKG from the data and metadata available in the IRIS database and from the semantic topics extracted by TMF. The IRIS dump is a JSON file that currently includes 23.268 records, for the time period 2008-2018. The metadata adopted in the IRIS platform are useful to manage the publication process. Considering this aspect, only a subset of the IRIS metadata is selected for the KG generation:

1. The publication identifier.
2. The title.
3. The abstract.
4. The author name, surname, and identifier.
5. The contributors and co-authors names, surnames, and identifiers (if present).
6. The date of publication.
7. The journal title and ISSN (if present).
8. The keywords entered by the authors to tag the publication.

The publication identifier is a unique code associated to each paper. Authors and co-authors, which are part of the Polito staff, are also uniquely identified by an id in the IRIS management system. External researchers, that can be contributors or co-authors of a publication are identified using their name and surname. The metadata of the papers that are published in a journal include its title and its ISSN identifier. On the other side, if the paper is included within a conference proceeding, there are no unique identifiers, while the name of the conference is available. Other information related to the publication, such as the title, the abstract, and the keywords are reported as strings among the metadata. Considering the semantics of the metadata, the ontology adopted to construct the facts includes the following classes:

1. The *Publication*.
2. The *Author*.
3. The *Journal*.
4. The *AuthorKeyword*.
5. The *TMFResource*.

Data and object properties defined in RDFS [93], FOAF[65], and DCMI[174] are used to semantically connect instances of the previous classes. Figure 4.4 depicts the schema of the PKG.

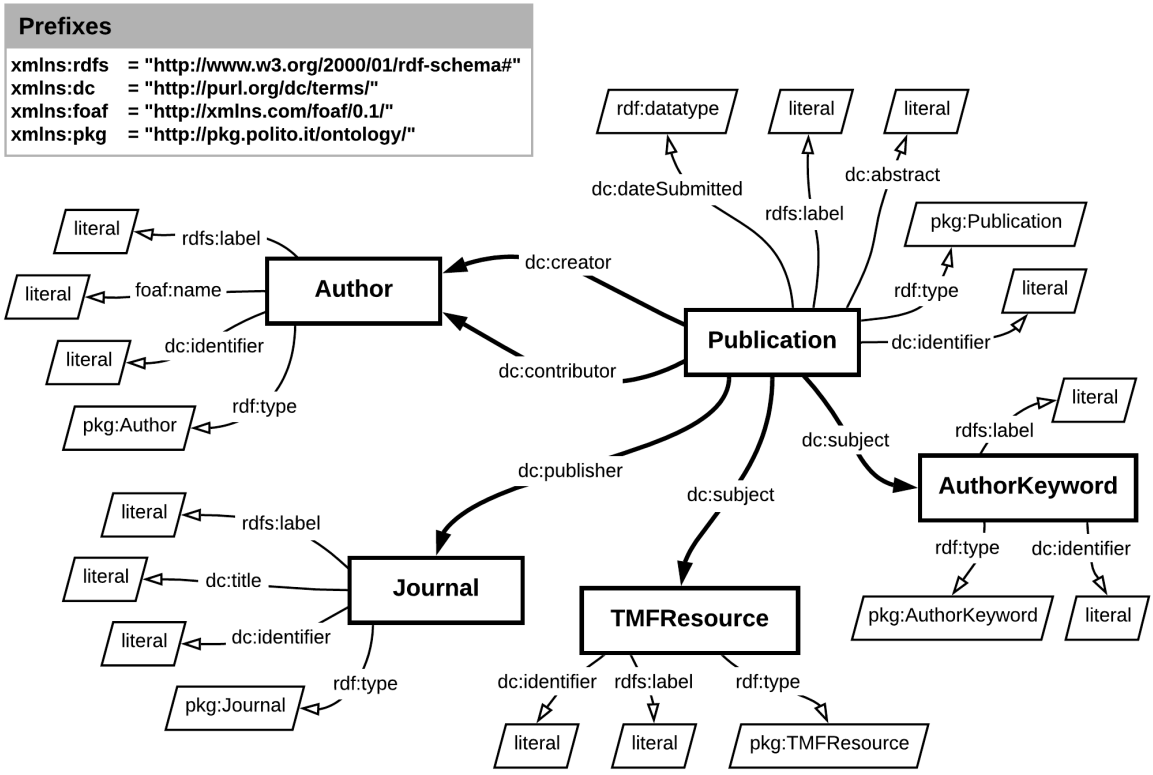

Figure 4.4: Schema of the PKG. The external ontologies used to define the structure are shown in the prefixes table.

The Graph Builder

The Graph Builder is the main component fo the Builder module and it is implemented as a Python command-line script. This script uses the *rdflib*[37] library

[37] https://github.com/RDFLib/rdflib

Table 4.4: Number of entities and edges in the Polito Knowledge Graph.

Polito Knowledge Graph				
Number of entities per-class				
Publication	Author	Journal	TMFResource	AuthorKeyword
23,268	34,886	3,211	16,988	41,807
Number of edges per-property				
Creator	Contributor	Publisher	Subject (TMF)	Subject (Keywords)
23,268	80,819	11,243	107,093	77,492

to create and manage an in-memory RDF representation of the KG. The script takes as input the path to the JSON dump of IRIS and the ontology. In order to increase the performance of the KG generation process, the script creates a *ThreadPoolExecutor* to enable an asynchronous execution of the functions, according to a predefined number of threads. Each thread asynchronously executes a function that processes a single record of the dump, matching the record data with the classes and the properties of the ontology, and creating the entities and the facts that populate the PKG. The internal representation created by *rdflib* is serialized and exported as an XML file that contains the definition of all the RDF facts included in the initial version of the PKG. The second step performed by the Builder is the extraction of semantic topics from the public abstracts, by means of TMF. These topics are added to the KG as *TMFResource* entities, and are linked to the publication by means of the *dc:subject* relation. Also in this case, the multi-thread implementation of the code reduces the processing time, parallelizing the requests to the RESTful APIs of TMF.

Table 4.4 summarizes the PKG features after the topic extraction performed by TMF. It reports the size of the KG, showing the number of entities categorized under each class, and the number of edges for each relation type.

The *AuthorKeyword* class represents the keywords freely added by the authors to label the publication. These keywords are included in the PKG, however they are not part of a controlled vocabulary and for this reason they are not useful to connect publications focused on the same topics. On the other side, TMF allows to extract the relevant semantic topics from the publication abstracts, allowing to link the papers to the same unique and unambiguous semantic entities.

Enhancing results

The goal of the enhancement process is to exploit the provide new recommendations based on the link prediction method based on the GAE. Qualitative and quantitative results of this specific task are discussed in Chapter 7.

Visualizing Search Results

The Viewer allows to explore the scientific publications of PoliTO and to visualize suggested recommendation for Polito researchers. The Viewer is composed of a triplestore, an API layer, and a Web application. The triplestore is a specialized DBMS for storing and retrieving RDF facts. It stores an online copy of the PKG and exposes a SPARQL endpoint that allows to perform query on it. The API layer exposes a RESTful service that allows to retrieve the information contained in the KG, hiding the SPARQL-logic complexity. It accepts HTTP requests with URL-encoded parameters and proceeds to perform the query against the SPARQL endpoint exposed by the triplestore, by matching the parameters upon some predefined queries. Then, it translates the response obtained from the triplestore in a JSON file, which is sent back to the client. The Web application is the entry point for the user, implementing the functionalities of a modern search engine. It allows to query and visualize the data contained in the KG through the use of a friendly interface. For example the user, who should typically be a researcher of the Politecnico di Torino, can search for a topic of interest and obtain as result the list of all the publications, authors, and journals that are linked to such topic. A screenshot on the results obtained searching for the publications on "Carbon Nanotube" is available in Figure 4.5. The interface presents an overview of the publication information, including the title, the authors, the other research topics extracted from the abstract. In the upper-right area there is a bar graph that can be exploited to filter the publication per year.

Predicted facts can be used to obtain insights and recommendations that can be shown by the Viewer Module in the search results. Leveraging such predictions, the recommendation system can suggest unexplored research topics to the researchers, scientific journals that have published papers related to their field of research, or the profiles of other researchers who share the same research interest, but that come from different departments or disciplines. The predictions of the GAE model are used to add interesting recommendations to the results. Considering the researcher page, the system suggests new topics which are not directly covered by the author and other researchers with similar profiles. A screenshot of the suggested topics and researchers on the author page is available in Figure 4.6.

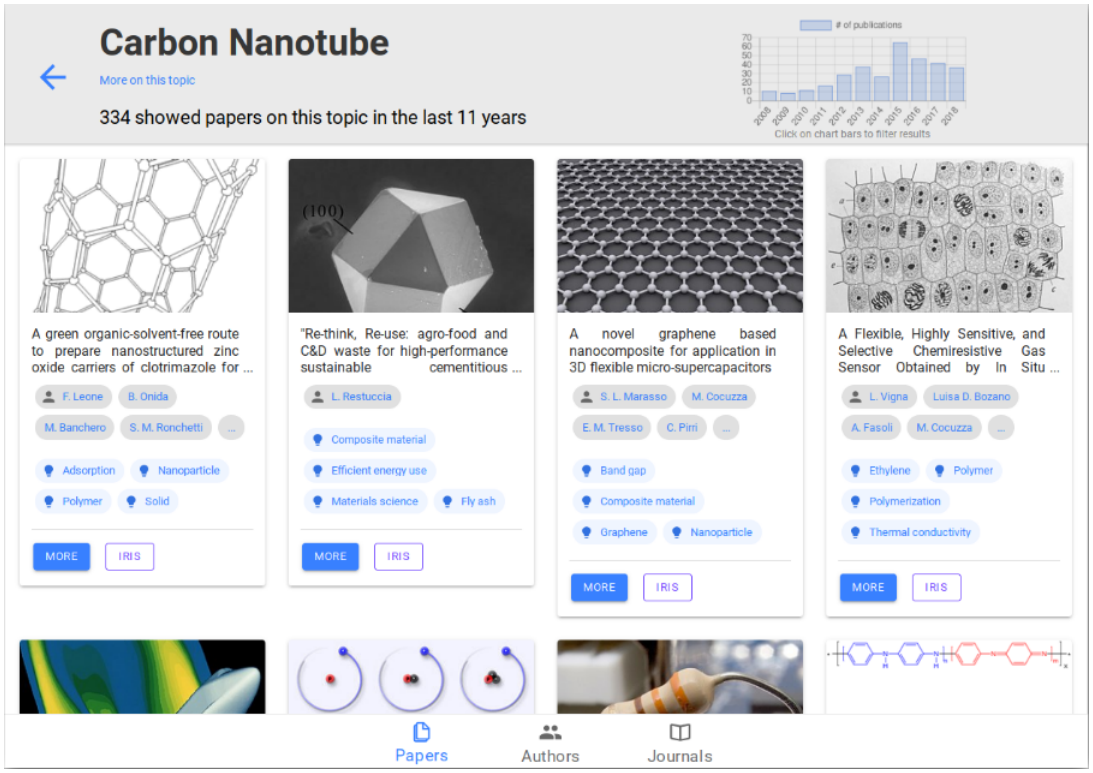

Figure 4.5: Screenshot of the results obtained searching for publications on the Carbon Nanotube research topic

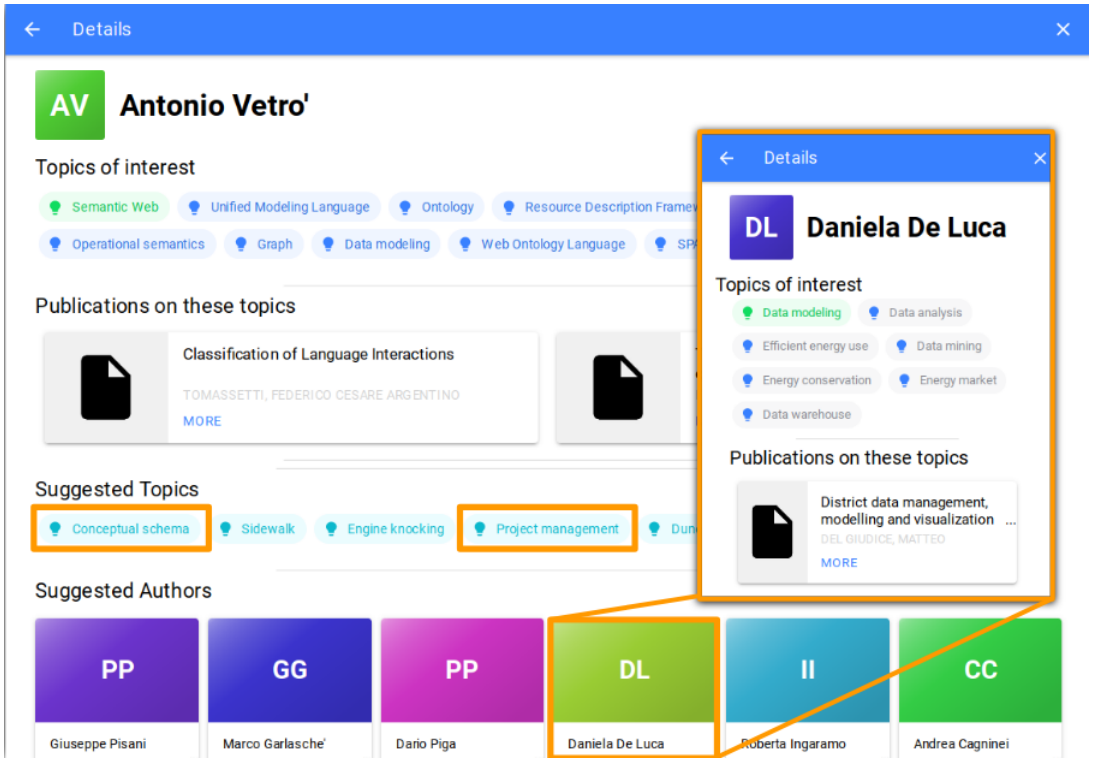

Figure 4.6: Screenshot of the the suggested topics and researchers the author page.

Chapter 5

Building Semantic Models with Word2Vec and SPARQL

This Chapter illustrates the initial research study conducted on semantic modeling. The study proposes a semi-automatic approach for constructing semantic models, based on the training of a Neural Language Model (NLM), i.e. Word2Vec [108], exploiting SPARQL queries performed on Knowledge Graphs (KGs) as training set. The graph patterns included in the body of SPARQL queries incorporate semantic information expressed through ontologies. As a matter of fact, consider the following pattern:

```
<http://dbpedia.org/resource/Alessandro\_Manzoni>
<http://dbpedia.org/ontology/birthPlace>
?birthPlace
```

It allows the user to retrieve the Alessandro Manzoni's place of birth. This query is particularly suitable to become a sentence for training a simple NLM for two reasons: (i) articles, conjunctions, adverbs, and prepositions are not present; (ii) the vocabulary used is limited, but rich in explicit semantics: properties like dbo:birthPlace[1] are defined by an ontology. The method assigns by means of the NLM a vector representation (*word embedding*) to SPARQL variables included in the graph pattern. A clustering technique aggregates SPARQL variables characterized by a closer vector representations, and each cluster is manually labeled with the proper semantic type and semantic relation. For instance, the cluster including the ?birthPlace, ?bp, and ?birth_place variables are labeled with the semantic

[1]Hereafter abbreviated versions for URIs are adopted for brevity reasons. In case of resources such as <http://dbpedia.org/resource/Alessandro_Manzoni>, dbpedia:Alessandro_Manzoni is used, while in case of ontology properties such as <http://dbpedia.org/ontology/birthPlace>, dbo:birthPlace is used

type "Place:name" and the semantic relation "dbo:birthPlace". The approach is evaluated according to the semantic accuracy metric, defined by the ISO standard 25012 [127] as the closeness of the data values to a set of values defined in a domain considered semantically correct.

This Chapter is structured as follows. Section 5.1 describes the key ideas behind the adopted approach for reconstructing the semantics of the data source. Section 5.2 describes implementation details. Section 5.3 reports details on the evaluation process, the preliminary results, and the limitation of this approach.

5.1 Approach

This Section reports: (i) a detail description on the adoption of Word2Vec to assign embeddings to SPARQL variables; (ii) an explanation of the syntactic similarities between the SPARQL variables and the attributes of a data source, in order to reconstruct the semantics of the latter; (iii) a definition of the concept of semantic model serialization.

5.1.1 Embedding Representation of SPARQL Variables

Graph patterns mentioned in SPARQL queries like "?person dbo:birthPlace ?birthPlace" are assertions characterized by a subject, a predicate, and an object. For such reason, they can be considered as natural language sentences, in which the words correspond to the elements of the triples.

Furthermore, variables mentioned in such triples, for instance ?birthPlace, have two peculiar features: (i) they contain a well defined semantic type (a place in our case); (ii) they have specific relationships with other entities in the KG (someone's birthplace in our case).

To better understand such peculiarities, consider the 2 following examples of SPARQL query bodies:

```
?person dbo:birthPlace ?birthPlace .
?birthPlace dbp:latitude ?lat .
?birthPlace dbp:longitude ?long .

?person dbo:birthPlace ?bp .
?bp dbp:longitude ?lat .
?bp dbp:longitude ?long .
```

In this case the variables ?birthPlace and ?bp are characterized by a similar context: they are preceded by either dbo:birthPlace and ?person, while they are followed by dbp:latitude and ?lat.

As explained by Ristoski and Paulheim [130], the goal of a NLM like Word2Vec is "to estimate the likelihood of a specific sequence of words appearing in a corpus, explicitly modeling the assumption that closer words in the word sequence are statistically more dependent". Therefore, NLMs are able to generate word embeddings that express semantic similarities. According to these principles, in our specific case variables like ?birthPlace and ?bp are characterized by a similar vector representation.

5.1.2 SPARQL Variables and Data Attributes

The approach exploits potential syntactic similarities between attributes of a data source and SPARQL query variables. The assumption is that traditional problems related to the name conflicts [122] between traditional data sources and ontologies are mitigated. For instance, users tend to write short names also in SPARQL variables than long speaking names. In other cases, users tend to use plural instead of singular expressions, where more than a single result is expected: in a SPARQL query that retrieves all actors of a specific movie, users are inclined to use ?actors instead of ?actor.

5.1.3 Semantic Model Serialization

The graph representation of semantic models has to be serialized using a mapping language. Assuming that the table reported in Fig. 5.1 is published in the CSV format with the name "authors.csv", the RML file that describes the semantic model on the basis of the ontology of Fig. 5.2 includes RDF statements reported in Fig. 5.3.

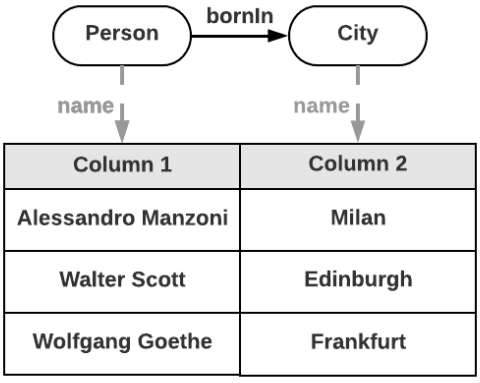

Figure 5.1: Example of semantic model on a table source

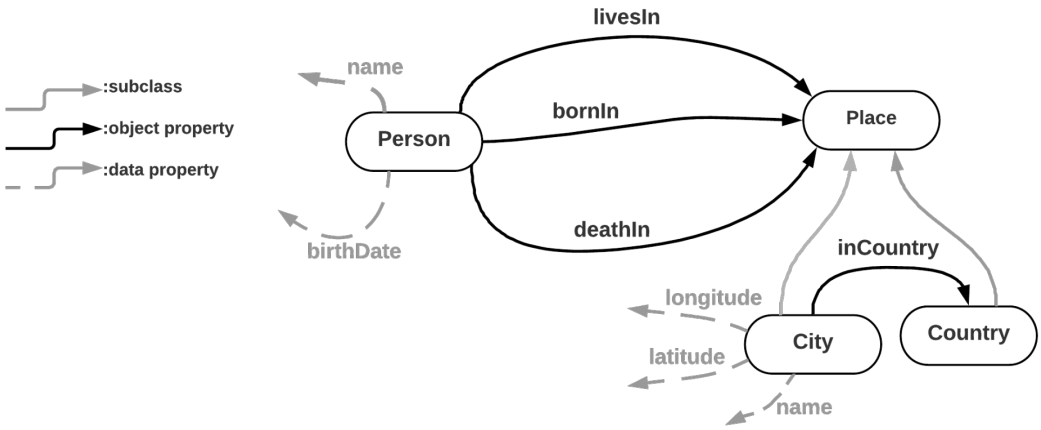

Figure 5.2: Example of ontology

```
<#Writers>
    rml:logicalSource [
        rml:source "authors.csv";
        rml:referenceFormulation ql:CSV
    ];

    rr:subjectMap [
        rr:template
"http://mysite/id/writer/{writers}";
        rr:class myontology:Person
    ];

    rr:predicateObjectMap [
        rr:predicate myontology:name;
        rr:objectMap [
            rml:reference "writers";
        ]
    ];

    rr:predicateObjectMap [
        rr:predicate myontology:birthPlace;
        rr:objectMap [
            rr:parentTriplesMap <#Places>;
        ]
    ].
```

```
<#Places>
    rml:logicalSource [
        rml:source "authors.csv";
        rml:referenceFormulation ql:CSV
    ];

    rr:subjectMap [
        rr:template
"http://mysite/id/city/{birthPlace}";
        rr:class myontology:City
    ];

    rr:predicateObjectMap [
        rr:predicate myontology:name;
        rr:objectMap [
            rml:reference "birthPlace";
        ]
    ].
```

Figure 5.3: Example of RML file

5.2 Implementation Details

This Section illustrate all stages to generate a semantic model, exploiting SPARQL queries as training sentences for a NLM. Each stage is performed by different software modules that are shown in Fig. 5.4

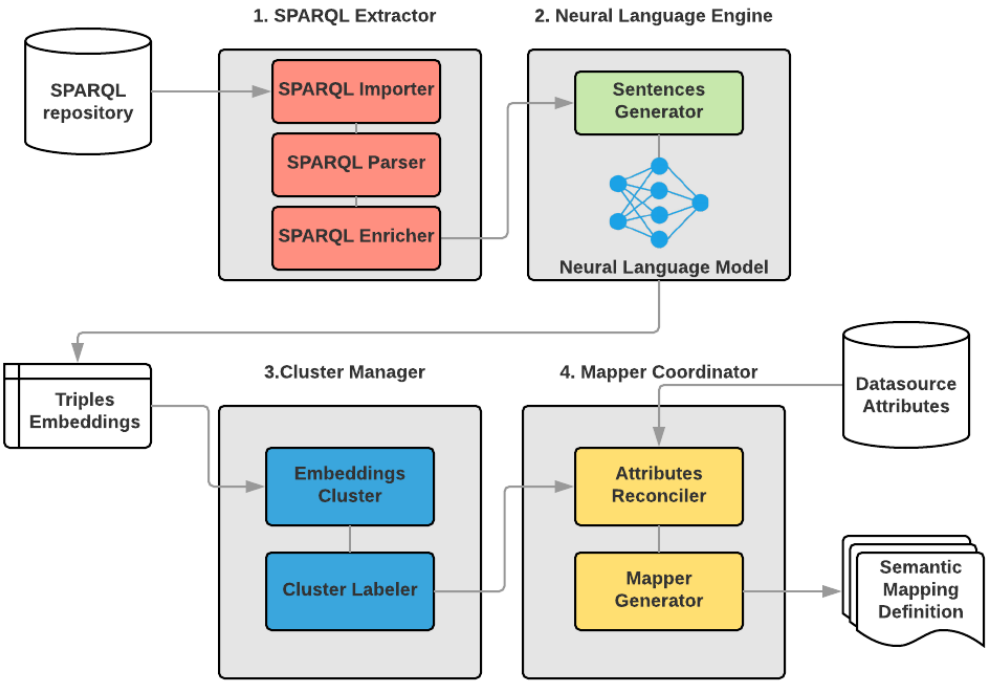

Figure 5.4: Modules and components of the pipeline for the generation of the semantic model

5.2.1 SPARQL Extractor

The SPARQL Extractor module (1 in Fig. 5.4) conducts a pre-processing stage in order to prepare a set of SPARQL queries as input of the NLM. To accomplish such goal, it uses three software components: the SPARQL Importer, the SPARQL Parser, and the SPARQL Enricher.

The SPARQL Importer downloads SPARQL queries made available by the LSQ project. Such SPARQL queries are published according to a specific ontology and can be retrieved from an endpoint available on the Web[2]. The download process is performed through a pipeline of 3 different queries to retrieve:

1. all properties mentioned in the body of stored queries. The result of the query provides the URIs of properties (e.g., dbp:birthPlace);

2. all URIs of queries data that contain such properties. For instance, the URI `http://lsq.aksw.org/res/DBpedia-q135894` describes a query that

[2]The LSQ SPARQL endpoint is available at: `http://lsq.aksw.org/sparql`.

contains the property dbp:birthPlace in the WHERE and in the OPTION clauses[3];

3. the body of each query identified by a URI. Considering http://lsq.aksw.org/res/DBpedia-q135894, it is possible to obtain the entire text of the query, exploiting the property http://spinrdf.org/sp#text defined by the LSQ ontology.

The SPARQL Parser component extracts triples from text of queries retrieved by the SPARQL component. Such triples are located inside the WHERE and the OPTION clauses, for instance "dbpedia:Alessandro_Manzoni dbo:birthPlace ?birthPlace". Triple patterns like this compose the training set for the NLM (see Section 4.2). The SPARQL Parser component is developed by means of the SPARQL.js library[4].

Finally, the goal of the SPARQL Enricher is to harmonize the context of SPARQL variables that express the same semantics. To clarify the behaviour of this component, please consider the following triple examples:

```
dbpedia:Alessandro_Manzoni dbo:birthPlace ?birthPlace .

dbpedia:Walter_Scott dbo:birthPlace ?birthPlace .

?person dbo:birthPlace ?birthplace .
```

In this case, the context of ?birthPlace and ?birthplace are not very similar, because the first one is included in sentences with different subjects: dbpedia:Alessandro_Manzoni, dbpedia:Walter_Scott. The third one, instead, is included in a sentence where ?person is the subject. Nevertheless, Alessandro Manzoni and Walter Scott are all entities categorized under the DBpedia class Person (http://dbpedia.org/ontology/person).

The SPARQL Enricher component retrieves the label of the highest-level classes of the concepts mentioned in the SPARQL queries (in this case the label of the class http://dbpedia.org/ontology/person is "person") and adds a new triple for each concept. In this case, 2 new triples in the form ?person dbo:birthPlace ?birthPlace, are added to the training set.

---

[3]Including triples mentioned in the OPTION clause, it is possible to extend the training set.

[4]The GitHub repository of the tool is available at https://github.com/RubenVerborgh/SPARQL.js/

5.2.2 Neural Language Engine

The goal of the Neural Language Engine module (2 in Fig. 5.4) is to assign an embedding representation to variables included in triples retrieved by the SPARQL Extractor. To perform this task, the module uses two software components: the Sentence Generator and the Neural Language Model (NLM).

The Sentence Generator component takes as input SPARQL triples and transform them in sentences to train the NLM. In details, the triple "dbpedia:Alessandro_Manzoni dbo:birthPlace ?birthPlace", for instance, is treated as a sentence made of different words: (i) dbpedia:Alessandro_Manzoni; (ii) dbo:birthPlace; (iii) ?birthPlace.

The NLM component generates the vector representation of the SPARQL variables, taking as input the sentences produced by the Sentence Generator. In this context, variables like ?birthplace and ?bp share a similar context and therefore they can be aggregate in the same cluster as it is described in the following stage. The NLM is an implementation of the Skip-Gram variation of Word2Vec, which is introduced discussed in Section 2.3.2.

5.2.3 Cluster Manager

The objective of the Cluster Manager module (3 in Fig. 5.4) is to assign a RML template[5] to clusters built on the embedding representation of SPARQL variables. Unlike previous software modules that work fully automatically, the Cluster Manager needs user intervention to correct the results of the clustering process and the definition of the RML template. To achieve its goal, such module exploits two software components: the Embedding Cluster and the Cluster Labeler.

The Embedding Cluster component aggregates vectors representing SPARQL variables located in a close proximity, according to the cosine similarity[6]. The algorithm used for the clustering is a combination between DBScan and K-means. At this stage, the user can adjust through a GUI the results of the clustering process in case of a wrong variables grouping.

The Cluster Labeler component provides to the user another GUI through which he can assign the RML templates according to the clusters generated by the previous component.

Here is available an example of how the Cluster Manager module works. Consider the following example of different clusters of SPARQL variables created by the Embedding Cluster component:

[5] An RML template has essentially the same contents of the RML file available in Figure 5.3, except that some parts of the code are replaced by parameters whose content is filled by the Cluster Manager itself and by the Mapper Coordinator module described in the next Section

[6] Cosine similarity is a measure of similarity between two non-zero vectors of an inner product space that measures the cosine of the angle between them.

- Cluster 1: ?person, ?p, ?people.
- Cluster 2: ?birthplace, ?birthPlace, ?bp, ?birth_place.

On the basis of these clusters, the RML template is composed by 3 elements (see Fig. 5.5):

- #[1] (in blue in Fig. 5.5) assigns the semantic type dbpedia:Person_name to SPARQL variables ?person, ?p, ?people.
- #[2] (in red in Fig. 5.5) assigns the semantic type dbpedia:Place_name to SPARQL variables ?birthplace, ?birthPlace, ?bp, ?birth_place.
- #[3] (in green in Fig. 5.5) assigns the relation dbo:birthPlace between the just mentioned semantic types.

SPARQL variables are included in the RML template and they are directly linked with the reference element (in bold in Fig. 5.5). The next module adds further information to the RML template that finally contribute to reconstruct the semantics of a data source.

5.2.4 Mapper Coordinator

The Mapper Coordinator module generates the semantic model (in the form of RML template) between the data source and the domain ontology. To reach this result, the module makes use of 2 different software components: the Attributes Reconciler and the Mapper Generator.

The Attributes Reconciler takes two inputs: (i) the RML template generated by the Cluster Manager module; (ii) the data source for which a semantic reconstruction is needed. The goal of such component is to reconcile the variables mentioned in the RML template and the attributes of the data source, considering syntactic similarities. To map the single attribute to the correct cluster of variables a score is computed, comparing the median value of the normalized Levenshtein distances [180] between the attribute and each element of the cluster. According to the highest value of this score, the component proceeds with the assignment.

To clarify this step, consider the table in Figure 5.1, with the two columns "person" and "birthPlace" as a CSV file entitled "authors.csv". The Attribute Reconciler component compares the word "person" with all the variables mentioned in the RML templates: in our example case, it is compared with the Cluster 1 (?person, ?p, ?people) and the Cluster 2 (?birthplace, ?birthPlace, ?bp, ?birth_place) defined in the previous Section. The attribute author is assigned to the first element of the RML template (#[1] (in blue in Fig. 5.5) with a specific score (see sm:score value in Fig. 5.5). The same process is executed for the "birth_place" attribute, that is assigned to the second element of the RML template (#[2] (in blue in Fig. 5.5) with a specific score (see sm:score value in Fig. 5.5).

```
#[1]
<#Writers>
    rml:logicalSource [
        rml:source "authors.csv";
        rml:referenceFormulation ql:CSV
    ];

    rr:subjectMap [
        rr:template
"http://mysite/id/writer/{person}";
        rr:class http://dbpedia.org/ontology/Person
    ];

    rr:predicateObjectMap [
        rr:predicate myontology:name;
        rr:objectMap [
             rml:reference "{person}";
        ];

    sm:variables ?person, ?p, ?people;

    sm:score 0.83;

#[3]
rr:predicateObjectMap [
        Rr:predicate
http://dbpedia.org/ontology/birthPlace;
        rr:objectMap [
            rr:parentTriplesMap <#Places>;
        ]
    ].
```

```
#[2]
<#Places>
    rml:logicalSource [
        rml:source "authors.csv";
        rml:referenceFormulation ql:CSV
    ];

    rr:subjectMap [
        rr:template
"http://mysite/id/city/{birthPlace}";
        rr:class
http://dbpedia.org/ontology/Place
    ];

    sm:variables ?birthplace,
?birthPlace, ?bp, ?birth_place;

    sm:score 0.9;

    rr:predicateObjectMap [
        rr:predicate myontology:name;
        rr:objectMap [
            rml:reference "{birthPlace}";
        ]
    ].
```

Figure 5.5: Example of RML template

5.3 Evaluation

This Section reports details on the evaluation design and the obtained results.

5.3.1 Design

The goal of the experiment is to measure the semantic accuracy of the semantic models generated with the described approach. As mentioned in previous Sections, the semantic accuracy can be defined as the closeness of the data values to a set of values defined in a domain considered semantically correct. For this reason, the semantic model generated by the system is compared with a semantic model produced by a domain expert on a specific field of knowledge. Assuming that the RML file of the semantic model produced by the domain expert is identified by DSM and the RML file of the semantic model generated by the system is identified

by SSM, the precision is computed considering the intersection between the triples mentioned in the two RML files and the number of triples in the RML file created by the domain expert.

$$precision = \frac{triples(DSM) \bigcap triples(SSM)}{triples(DSM)}$$

The precision assumes a value between 0 and 1. For the evaluation of the semantic accuracy, it is considered only the precision and not the recall metric, because domain experts can add much more RML triples to define the semantic model than a semi-automatic system, even if these are not specified in the original data source. For instance, they can include triples related to the description of an entity, using the property http://dbpedia.org/ontology/description of the DBpedia ontology. Nevertheless, this new assertion includes additional information which is not strictly related to semantic accuracy, even though it may be useful to the user to better understand the meaning of a specific resource.

To reach their goal, human experts and the system share some details in order to avoid differences in RML triples that are not strictly related to the semantic mapping process. In particular, (i) they share knowledge about the ontology as starting point to create the semantic model, (ii) they share the root of the URI in order to create resources of concepts mentioned in the data source. To clarify this second point, assume that for identified entities in the data source: both the domain expert and the system have to use the URI http://mydomain/entities/_NAME_, where _NAME_ is the value of the attribute of the data source. In this way, it is avoided to create differences in terms of RML triples not related to the semantic mapping process, that can lead to a decrease in the value of precision.

5.3.2 Results and Discussion

The training set is composed of SPARQL queries published by the LSQ project. Such project provides SPARQL queries performed on endpoints of different research projects: DBpedia[7], Linked Geo Data[8], Semantic Web Dog Food[9], British Museum[10]. From SPARQL queries retrieved from LSQ, 427.186 triples have been extracted. Each of this triple constitutes a training sample for the Word2Vec model.

For the semantic model generation task, only the DBpedia ontology[11] is chosen,

[7]Project website: http://dbpedia.org

[8]Project website: http://linkedgeodata.org

[9]Project website: http://data.semanticweb.org

[10]Project website: http://bm.rkbexplorer.com

[11]More information on the DBpedia ontology is available at: http://wiki.dbpedia.org/services-resources/ontology.

whose properties are the most used in the SPARQL queries.

To show the potential of the approach, we the semantic modeling process is tested with 3 different data sources, that cover the same data using different attributes:

1. The Wikipedia infobox template for a person[12].
2. Web tables of the Famous Birthdays website[13].
3. Web tables of the Biography.com website[14].

All these data sources include the data attributes available in Table 5.1.

Wikipedia	**Famous Birthdays.com**	**Biography.com**
name	NONE	name
birth_date	birthday	birth date
birth_place	birthplace	place of birth
death_date	death date	death date
death_place	NONE	place of death

Table 5.1: Attributes of data sources reported in Wikipedia, Famous Birthdays.com, Biography.com

Here are available the precision values of the semi-automatic mapping generation process on the three specific cases:

- Wikipedia: 1
- Famous Birthday.com: 0.3
- Biography.com: 0.6

In the case of Wikipedia, there is a complete overlap between the RML triples generated by the system and by the domain expert. The issue with FamousBirthdays.com is that the name of the subject entity is not directly reported in the Web table, but it is reported in another section of the Web page. In the case of Biography.com, the low of precision is related to the strings "place of birth" and "place of death", because they are never used as variables in SPARQL queries and the module that exploits the normalized Levenshtein distance was not able to recognize the semantic affinity with other expressions like "birth_place" and "death_place"

[12]More information at: https://en.wikipedia.org/wiki/Template:Infobox_person

[13]More information at: https://www.famousbirthdays.com

[14]More information at: https://www.biography.com

syntactically very distant. Considering this issue, the adoption of lexical databases, such as Wordnet, can contribute to expand the vocabulary of SPARQL variables. Moreover, this approach does not take full advantage of the graph structure for the learning process: SPARQL queries include a limited number of graph patterns and Word2Vec treats these patterns as plain text. More advanced techniques based on the structure of the graph, described in Chapter 4, are able to address these specific issues.

Chapter 6

SeMi: Building Semantic Models with Graph Neural Networks

This Chapter presents SeMi (SEmantic Modeling MachIne), a system that provides a modular pipeline to reconstruct the semantic model of a data source. SeMi covers the entire process of semantic modeling: (i) it provides a semi-automatic step to detect semantic types; (ii) it exploits a novel approach to inference semantic relations, based on a Graph Auto-Encoder (GAE) trained on background linked data. The approach developed within SeMi takes inspiration from the work of Taheriyan et al. [157]. In this article the authors describe a method that exploits linked data as background knowledge to infer semantic relations within a data source. They perform a manual extraction of features from linked data with SPARQL [46] queries. These features include various types of complex graph patterns, that represent semantic relations of different lengths. The extraction of these patterns requires a compound feature engineering process, it is not scalable as the length of the semantic relation increases, and it requires prior knowledge of the linked data structure. On the contrary, the SeMi method to extract features from linked data is automatic. The proposed approach is based on a GAE that automatically learns latent features from the local neighborhood structures of the linked data graph. These latent features are aggregated into entity and property embeddings, which are employed to predict the semantic relations within the target source. The adoption of the embeddings increases the accuracy in reconstructing the correct relations in complex data sources, compared to manually-selected features. Furthermore, the GAE training is a more scalable procedure than the extraction of increasingly complex graph patterns through SPARQL queries.

The remainder of this Chapter includes the following sections. Section 6.1 provides details on the system goal of SeMi and its main architectural requirements. Section 6.2 introduces the pipeline components for the semantic model generation, whose implementation details are reported in Section 6.3. Details on the evaluation method, driven by the analysis of the generated semantic models, and the results

obtained by SeMi against the state of the art are described in Section 6.4. Further results achieved by SeMi with a different validation procedure, driven by a query-based analysis, are reported in Section 6.5. Section 6.6 reports the performance of SeMi in the public procurement scenario, adopting also in this case. a query-based analysis.

6.1 Goal and Main Architectural Requirements

The goal of the SeMi tool is to produce a semantic model, given a data source, a domain ontology, and a background linked data as input. To achieve such a general goal, two architectural requirements have been identified:

- *A flexible and modular pipeline*: the first requirement consists in the development of a pipeline where each component can be individually improved or replaced, for a variety of purposes, such as tailoring the tool to a very specific domain, or injecting user input in an intermediate step. In the current implementation of SeMi , only the semantic labeling step enables the possibility of user-based refinements.
- *A GAE model included in a production pipeline*: the implementations of GAEs in research literature (i) show results in comparison to other models (ii) on available benchmarks and (iii) according to specific evaluation metrics. Nevertheless, the usage of these models in a data pipeline to reach a specific purpose is a complex task. According to this requirement, a re-engineered version of an existing GAE has been purposed for the semantic relations inference.

6.2 SeMi Pipeline Components

The pipeline to produce semantic models includes five main components (Figure 6.1):

- Semantic Type Detector (STD);
- Multi-Edge and Weighted Graph Generator (MEWGG);
- Semantic Model Builder (SMB);
- Link Predictor (LP);
- Semantic Model Refiner (SMR).

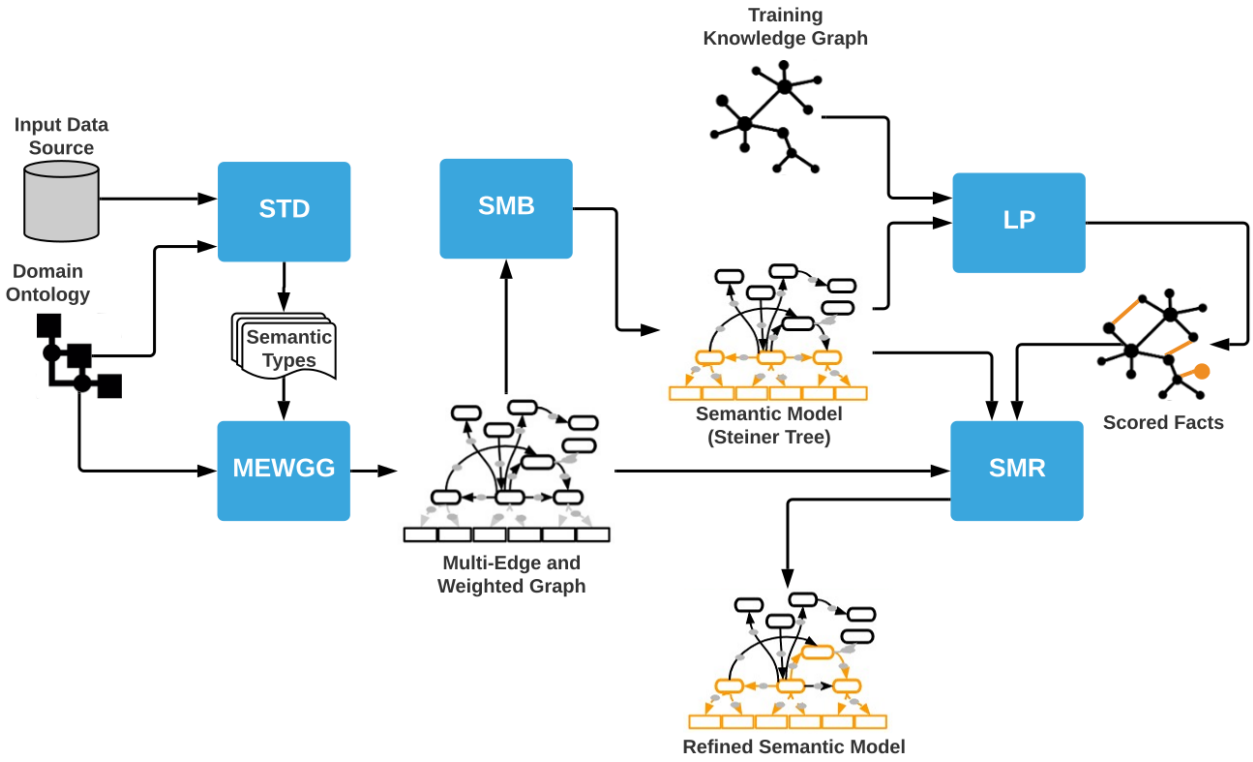

Figure 6.1: Pipeline components of SeMi

6.2.1 Semantic Type Detector

The STD component takes a data source and an ontology as input. The goal of the STD is to assign a semantic type (also called *semantic label*) to each attribute of a data source. Such semantic label consists of a combination of an ontology class and an ontology data property.

6.2.2 Multi-Edge and Weighted Graph Generator

The MEWGG component receives the semantic types produced by the STD and the domain ontology. The goal of the MEWGG is to build a multi-edge and weighted graph that includes all plausible semantic models for the target source. More specifically, MEWG reconstructs all possible semantic relations between all the semantic types, according to the object properties of the domain ontology. The weights assigned to these object properties are only based on the structure of the ontology (see Section 6.3.2).

6.2.3 Semantic Model Builder

The SMB component takes the graph produced by the MEWGG as input. The goal of the SMB is to select an initial semantic model, among all plausible semantic models, which encompasses the minimum cost path on the graph that connects all semantic types. The detection of this path in the graph can be considered a steiner tree problem [81]. To resolve the steiner tree problem, the approach provided by Kou et al. [91] has been adopted. In this case, the classes of the semantic types are the steiner nodes and the generated semantic model describes the target source in terms of classes and properties of the domain ontology. The detection of the steiner tree in the graph produced by the MEWGG has the following limit: the object property weights are based only on the structure of the ontology and they do not necessarily reflect the correct semantic interpretation of the target source.

6.2.4 Link Predictor

The LP component is characterized by an offline and online stage. In the offline stage, the input of the LP are RDF facts from background linked data repositories. This background knowledge covers data belonging to the same domain of the target source, it adopts a subset of the properties declared in the ontology, and it includes instances of classes of the domain ontology. Linked Data is adopted for training the GAE model, whose output are the embeddings of entities and object properties of RDF facts seen during the training process. In the online stage, these embeddings are used to score the unseen RDF facts, resulting from all plausible semantic models produced by the MEWGG. In details, the RDF fact scores are properly used to adjust the weights assigned to the graph by the MEWGG, incorporating information from the background knowledge provided by linked data. The intuition behind this refinement is that properties used by other people to semantically describe data in a domain are more likely to represent the semantics of the target source in the same domain.

The adoption of the LP component represents a step towards in the semantic modeling process. In fact, in the work of Taheriyan et al. [157] the weights of the semantic relations derived from complex graph patterns are assigned through an inverse relation of their frequency in the background knowledge. SeMi adopts latent information embodied in the background knowledge, which are not the result of a manual features extraction from the linked data, but it is automatically extracted by means of the GAE (see implementation details in subsection 6.3.4).

6.2.5 Semantic Model Refiner

The SMR component receives the initial semantic model built by the SMB, the embeddings to score unseen RDF facts, and all plausible semantic models included

in the MEWGG. The SMR computes the aggregation of the RDF score facts that refer to the same semantic relation. Comparing such aggregated values, the SMR replaces (or confirms) the semantic relations that link semantic types classes and performs a new steiner tree detection. Therefore, the goal of the SMR is to provide a new semantic model that describes in a more accurate and rich way the semantics of the target source.

6.3 Implementation Details

This Section discusses the implementation of the SeMi current release.

6.3.1 Semantic Type Detection querying Indexes

To assign a semantic label, the STD measures the similarity [107] between the data values of unlabeled attributes of the target source and the the data values of sources with labeled attributes. Such labeled data sources are stored within an Elasticsearch [64] Lucene index: therefore, the STD composes and performs a Lucene query to obtain a ranking of scored semantic types for each attribute of the target source. After this ranking, the user can select which semantic types correctly label the attributes of the target source.

The current implementation of the STD is available in Node.js, because it is suitable to interact with RESTful services provided by Elasticsearch.

6.3.2 Incremental Generation of the Multi-Edge and Weighted Graph

The MEWGG component incrementally creates a graph G through the following steps (see Algorithm 1 for more details):

- ***Addition of semantic types*** (lines 2-6 of Algorithm 1): for each semantic type the algorithm creates and adds to G the following graph structures: (i) a class node, (ii) a data node, (iii) a weighted edge between the two nodes. Then, it assigns a weight of 1 to this edge. In the public procurement scenario, if *pc:Contract_dcterms:identifier* is the semantic type of the source attribute *pc*, the algorithms creates the following graph structures: (i) a class node labeled as *pc:Contract*; (ii) a data node labeled as *pc*; (iii) a weighted edge labeled as *dcterms:identifier* from the *pc:Contract* class node to the *pc* data node (Figure 6.2).
- ***Addition of closure nodes*** (lines 7-10 of Algorithm 1): for each class node in G the algorithm performs a SPARQL query [46] to get the related ontology classes. Such classes are added as new class nodes (*closures*) to G .

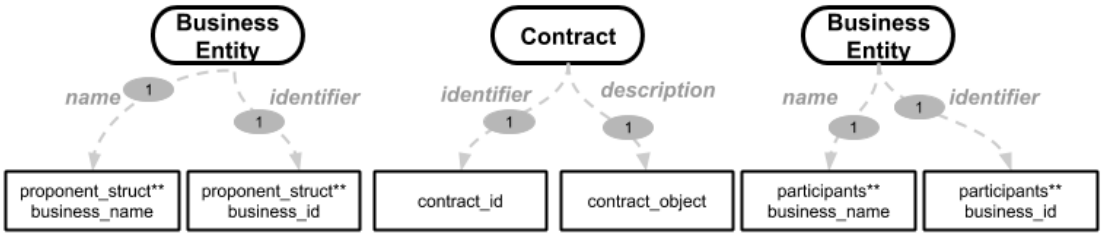

Figure 6.2: Multi-edge and weighted graph including all semantic types in the public procurement domain.

- ***Addition of edges between class nodes*** (lines 11-22 of Algorithm 1): object properties (see Section 6.2) connecting all class nodes in G are retrieved from the ontology through a SPARQL query. Such properties are added to G as new edges.
- ***Assignment of a weight to each new edge***: different types of object property p can connect the class nodes c_u and c_v in G :
  - *direct properties* (lines 15-16 of Algorithm 1): p is a direct property between c_u and c_v if they are respectively defined as domain and range of the p in the ontology. The algorithm assigns a weight of 100 to edges corresponding to direct properties;
  - *inherited properties* (lines 17-18 of Algorithm 1): p is an inherited property between c_u and c_v if its domain contains one of the super classes of c_u and its range contains one of the super classes of c_v. The algorithm assigns a weight of $100 + \epsilon$ to edges corresponding to inherited properties;
  - *subclass properties* (lines 19-20 of Algorithm 1): p is a subclass property between c_u and c_v if they are linked by a special property in the ontology called *rdfs:subClassOf*. The algorithm assigns a weight of $100/\epsilon$ to edges corresponding to subclass properties.

  Figure 6.3 shows the generated multi-edge weighted graph including new nodes and edges.

This component is implemented in Node.js, because it is suitable to manage the response of SPARQL queries in an asynchronous way.

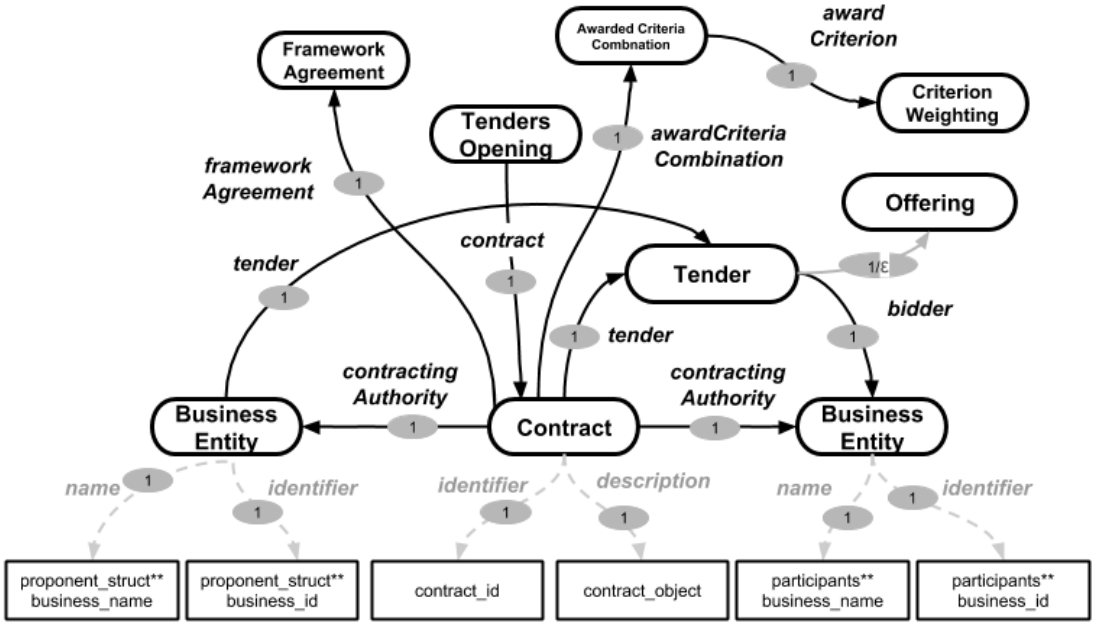

Figure 6.3: Multi-edge and weighted graph including all plausible semantic models in the public procurement domain.

Algorithm 1: Generate the Multi-Edge and Weighted Graph

Input: Semantic Types $STs\{class, data\_prop, source\_attr\}$
Input: Domain Ontology $O\{classes, object\_props\}$
Output: Graph G

for *loop on STs* **do**
  $G.add\_class\_node(ST\{class\})$;
  $G.add\_data\_node(ST\{source\_attr\})$;
  $G.add\_edge(ST\{class\}, ST\{source\_attr\}, ST\{data\_prop\}, w = 1)$;
end
$closures \leftarrow sparql\_closure(O\{classes\})$;
for *loop on closures* **do**
  $G.add\_class\_node(closure)$;
end
$c_us \leftarrow get\_class\_nodes(G)$;
$c_vs \leftarrow get\_class\_nodes(G)$;
for *loop on* c_us **do**
  for *loop on* c_vs **do**
    $direct\_props \leftarrow sparql\_direct(c_u, c_v, O\{object\_props\})$;
    $G.add\_edges(c_u, c_v, direct\_props, w = 100)$;
    $inherited\_props \leftarrow sparql\_inherited(c_u, c_v, O\{object\_props\})$;
    $G.add\_edges(c_u, c_v, inherited\_props, w = 100 + \epsilon)$;
    $subclass\_props \leftarrow sparql\_subclass(c_u, c_v, O\{object\_props\})$;
    $G.add\_edges(c_u, c_v, subclass\_props, w = 100/\epsilon)$;
  end
end

6.3.3 Semantic Model Definition through Steiner Trees and SPARQL Syntax

The detection of the of the shortest path within the graph that connects the class nodes of the semantic types is a steiner tree problem. The time complexity of the steiner algorithm (see Section 6.2.3) is equal to $O(|N_d||N_c|^2)$ in which N_d is the set of data nodes and N_c is the set of class nodes in G . Considering the dataset adopted in the experimental evaluation (see Subsection 6.4.1 for more details), the time complexity for this step is negligible compared to other steps, for instance the training time of the GAE. However, it can be an indicator for other users that intend to adopt SeMi for building their own KGs. Figure 6.4 shows the semantic model automatically generated through the detection of the steiner tree (in orange color). In this case, the "contractingAuthority" object property that connects the contract and the business entity in the right side of the graph is wrongly inferred with respect to the semantics of the input data source.

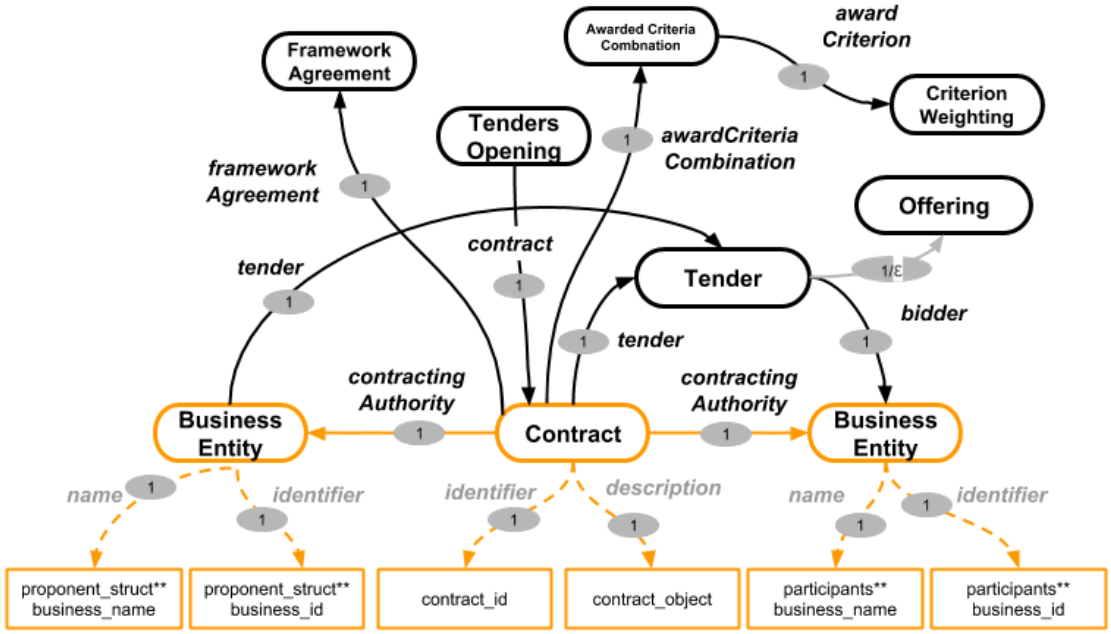

Figure 6.4: Automatically generated semantic model achieved applying the steiner tree detection on the multi-edge and weighted graph.

The detected steiner tree represents, in the form of a graph, the initial semantic model. Nevertheless, the semantic model needs to be converted using the syntax of a mapping language for the statement generation. SeMi adopts a specific SPARQL syntax that can be processed by the JARQL library [143] (theoretical details on the mapping step are available in Section 2.2.1). The source code that converts the graph representation of the semantic model in SPARQL is implemented in Node.js, while JARQL is implemented in JAVA within an external library.

6.3.4 GAE Architecture for Link Prediction

The LP is based on a GAE architecture. The encoder component is an extension of the isotropic formulation of the Vanilla GCN (see Subsection 2.3.3), which is known as Relational Graph Convolutional Networks (R-GCNs), which take into considerations multiple relations between nodes. Recalling equation 2.7:

$$h_i^{l+1} = ReLU\left(h_i^l\, W_1^l \;+\; \frac{1}{deg_i} \sum_{j \in N_i} h_j^l\, W_2^l\right)$$

,

the R-GCN propagation formula is equal to:

$$h_i^{l+1} = ReLU\left(h_i^l\, W_1^l \;+\; \frac{1}{deg_i} \sum_{r \in R} \sum_{j \in N_i} h_j^l\, W_r^l\right), \tag{6.1}$$

where $r \in R$ introduces node aggregation of i neighbours under the relation r.

As happen for the Vanilla GCN, the computation of Equation 6.1 is performed in parallel for all nodes at each network update. By stacking up several layers, it is possible to capture and encode the relations between nodes across multiple steps.

The decoder employs the DistMult factorization method [178], as done by Schlichtkrull et al. [144]. DistMult associates each relation r with a diagonal matrix R_r, and the score for a given candidate fact is computed as follows (line 12 of Algorithm 2):

$$f(s, r, o) = e_s^T R_r e_o \tag{6.2}$$

The model is trained with background linked data with negative sampling: for each training sample, a set of negative samples is generated by randomly corrupting either the subject or the object of the fact. The model is optimized so that the positive facts are scored higher than the negative ones.

To include the LP in the semantic model pipeline the implementation of [144] has been modified, in order to produce embeddings that are used by the next block to score RDF facts generated by the plausible semantic models of the target source. The LP is implemented in Python adopting the Deep Graph Library [171] based on the PyTorch framework.

Algorithm 2: Link prediction with the GAE

Input: $complete\_set$, $train\_set$, $valid\_set$, $test\_set$, adj_matrix
Output: $entity\_embs$, $property\_embs$

$entity\_dict \leftarrow create\_entity\_dict(complete\_set)$;
$property\_dict \leftarrow create\_property\_dict(complete\_set)$;
$entity\_embs \leftarrow initialize\_embs(entity\_dict)$;
$property\_embs \leftarrow initialize\_embs(property\_dict)$;
/* Forward: aggregate and update entity embeddings using the local neighborhood structure */
$entity\_embs \leftarrow update\_features(entity\_embs, adj\_matrix)$;
/* Backward: computing gradients and update entity and property embeddings to reconstruct the facts */
$s\_tr_{id}, p\_tr_{id}, o\_tr_{id} \leftarrow extract\_ids(train\_set)$;
$s\_va_{id}, p\_va_{id}, o\_va_{id} \leftarrow extract\_ids(valid\_set)$;
$s\_te_{id}, p\_te_{id}, o\_te_{id} \leftarrow extract\_ids(test\_set)$;
for *epochs* **do**
    $grads \leftarrow compute\_grads(emb(s\_tr_{id}) * emb(p\_tr_{id}) * emb(o\_tr_{id}))$;
    $entity\_embs, property\_embs \leftarrow update\_weights(grads)$;
    $model \leftarrow best\_on\_validation(emb(s\_va_{id}), emb(p\_va_{id}), emb(o\_va_{id}))$;
end
/* Evaluation: evaluate the best model on the test set and return computed embeddings */
$fact\_scores \leftarrow evaluate(emb(s\_te_{id}), emb(p\_te_{id}), emb(o\_te_{id}), model)$;

6.3.5 Semantic Model Refinement Based on Fact Scores

The SMR determines whether to refine or validate the initial semantic model. The weight (cost) of each edge included within the semantic relation are updated according to the aggregated scores of the related RDF facts according to the equation 6.3

$$cost(r_i) = \frac{1}{\frac{1}{|\tau|}\sum_{s,r_i,o}\sigma(f(s,r_i,o))} \tag{6.3}$$

On the basis of this new weights, the algorithm performs a new steiner tree detection and propagates any changes to the initial semantic model. Figure 6.5 shows the refined semantic model in orange color, and the wrong semantic relation from the initial semantic model depicted with the red color.

This component is implemented in Python.

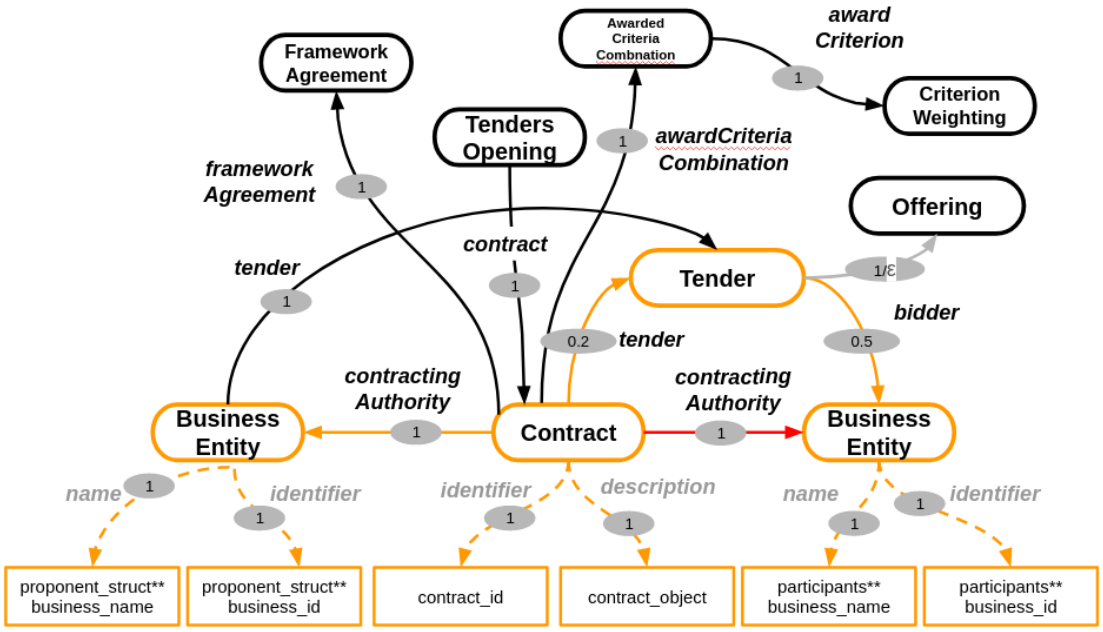

Figure 6.5: Refined semantic model obtained with the link prediction mechanism.

6.4 Evaluation Based on the Semantic Model

The objective is to investigate the capability of SeMi in inferencing accurate semantic relations within a target source. This Section describes the dataset adopted for the evaluation and reports details on the validation procedure, discussing the obtained results.

6.4.1 Evaluation Dataset

The evaluation dataset includes 15 target sources available in JSON format on the advertising domain. The domain ontology is an extension of Schema.org [67], which contains 736 classes and 1081 properties. To prepare the background knowledge for each target source the leave-one-out setting has been employed. In practice, if k is the number of sources in our dataset, the background linked data assigned to each target source is created from the RDF facts obtained by the other $k-1$ sources. In other words, each background knowledge includes RDF facts which come from all the sources, except those obtained from the target source. The leave-one-out setting guarantees that the background knowledge does not contain the facts related to the semantic relations within the target source, that have to be predicted in the experiment. Nevertheless, RDF facts from the target source, specifically related to the semantic types, have been included. This step has no impact on the performance of semantic relation inference, because semantic type

facts are not considered during the experimental evaluation. However, additional facts derived from semantic types are required within the training process, because the GAE has to learn the latent features of all entities, including the target source entities, in order to execute the link prediction task. To ensure the quality of linked data repositories adopted as background knowledge, the RDF facts are generated using the ground-truth semantic models.

The original dataset, including the target sources, the ontology, and the ground-truth semantic models are available in the Taheriyan GitHub repository[1]. The evaluation has been performed on this specific dataset, because it represents the case in which the approach of Taheryan et al. [157] obtained the best performance. The background linked data for training the GAE are novels and constructed specifically for this experiment. To simplify the access to all data employed in the evaluation, two folders have been created in the SeMi GitHub repository: (i) input sources, ontologies, and background linked data are available at https://github.com/giuseppefutia/semi/tree/master/data/taheriyan2016/task_04; this folder includes also the semantic models generated by the SeMi tool. The ground truth semantic models and the semantic models computed by baseline algorithms (see Subection 6.2.2 for more details) are available at https://github.com/giuseppefutia/semi/tree/master/evaluation/taheriyan2016/task_04. Details on the input sources (number of attributes), on the background linked data (number of entities and object properties), and on ground-truth semantic models (number of nodes and semantic relations) are reported in Table 6.1.

6.4.2 Evaluation Procedure and Results

The evaluation procedure relies on two different steps: (i) the validation of the GAE; (ii) the validation of the semantic relation inference task.

Validation of the GAE

The first step of the evaluation procedure consists in the validation of the LP mechanism, based on the GAE trained with background linked data. Each background linked data assigned to a target source has been splitted into three different datasets: the training set, the validation set, and the test set. Then, these datasets are taken as input by the GAE and used to perform the LP. To measure the performance of the LP, the standard Mean Reciprocal Rank (MRR) [33] has been employed. The MRR provides an insight on the correctness of the facts reconstructed by the GAE, exploiting the learned embeddings of entities and object properties. For each background linked data, Table 6.2 reports: (i) details on the number of facts included in the training set, the validation set, and the test set respectively;

[1] https://github.com/taheriyan/iswc-2016/blob/master/weapon-ads.zip

Sources	#attrs	Background LD		Ground-Truth SMs	
		#entities	#facts	#nodes	#links
alaskaslist	8	3396	6954	12	3
armslist	20	3396	6793	15	4
dallasguns	15	3379	6940	23	7
elpasoguntrader	8	3396	7044	13	4
floridagunclassifieds	16	3396	6904	23	6
floridaguntrader	10	3396	6774	15	4
gunsinternational	10	3396	6945	19	4
hawaiiguntrader	7	3396	7122	11	3
kyclassifieds	10	3396	6945	14	3
montanagunclassifieds	9	3396	7104	14	4
msguntrader	11	3375	7086	16	4
nextechclassifieds	20	3396	6198	32	11
shooterswap	11	3396	7041	15	3
tennesseegunexchange	14	3396	7104	21	6
theoutdoorstrader	12	3396	6784	18	5

Table 6.1: Details on target sources, background linked data, and ground truth semantic models

(ii) the resulting MRR on the test dataset. An overview of the most significant hyperparameters used to train the GAE is available in Table 6.3.

To understand the effectiveness of the GAE on our background linked data, the obtained results have been compared with the MRR values obtained by the GAE on FB15-k237[162], one of the most well-known dataset for benchmarking KG completion tasks. These MRR values reported in literature [144] are:

Sources	Background LD - #Facts			Mean Reciprocal Rank (MRR)		
	training	validation	testing	Raw	Hits @1	Hits @3
alaskaslist	6264	345	345	0.202556	0.171014	0.221739
armslist	6123	335	335	0.189313	0.156716	0.214925
dallasguns	6250	345	345	0.222723	0.201449	0.233333
elpasoguntrader	6344	350	350	0.175496	0.135714	0.198571
floridagunclassifieds	6214	345	345	0.213165	0.191304	0.224638
floridaguntrader	6104	335	335	0.207233	0.174627	0.229851
gunsinternational	6264	345	345	0.205095	0.188406	0.211594
hawaiiguntrader	6412	355	355	0.208059	0.180282	0.223944
kyclassifieds	6255	345	345	0.191376	0.163768	0.207246
montanagunclassifieds	6394	355	355	0.233740	0.212676	0.245070
msguntrader	6386	350	350	0.209148	0.188571	0.222857
nextechclassifieds	5588	305	305	0.204046	0.177049	0.216393
shooterswap	6341	350	350	0.226965	0.205714	0.241429
tennesseegunexchange	3694	355	355	0.203350	0.180282	0.214085
theoutdoorstrader	6114	335	335	0.185680	0.159701	0.205970

Table 6.2: Number of facts in the training, the validation, and the testing set and the MRR values obtained by the GAE on each background linked data

- MRR Raw: 0.158
- Hits 1: 0.153
- Hits 3: 0.258

MRR values obtained on background linked data (Raw and Hits 1) are higher

Hyperparameters	Values
Dropout	0.2
Hidden Layers	2
Hidden Neurons	100
Learning Rate	1e-2
Epochs	6000
Regularization	0.01
Edges Sample Size	1000
Negative Sampling	10

Table 6.3: GAE hyperparameters

than the MRR values obtained on FB15-k237, therefore the GAE performed very well on the background linked data. In practice, this means that entity and object property embeddings encode in a proper way the local neighborhood structure of the background knowledge. As a consequence, these embeddings can be suitable to score facts derived from all plausible semantic relations in the target source, identifying the most accurate ones. To further explore the effectiveness of this approach, Table 6.4 shows the differences in terms of MRR values between GAE and the DistMult scoring function (without the GNN). Observing the achieved MRR values (Raw, Hits 1, and Hits 3 always outperform), GAE always outperforms DistMult.

Validation of the Semantic Relation Inference Task

The second step of the experimental evaluation investigates if SeMi performs better than the approach of Taheriyan et al. [157], that is currently implemented in Karma [89]. Furthermore, SeMi has been compared with two different base lines: (i) a method exploiting only the frequency of semantic relations of length 1 within linked data (no heuristics to extract and rank complex the graph patterns, as done by Taheriyan et al. [157]); (ii) a method based on the detection of a steiner tree on a multi-edge and weighted graph, whose weights are assigned using only the ontology structure (no background knowledge). The evaluation procedure

Sources	MRR - DistMult			MRR - GAE		
	Raw	**Hits @1**	**Hits @3**	**Raw**	**Hits @1**	**Hits @3**
alaskaslist	0.136027	0.086957	0.166667	**0.202556**	**0.171014**	**0.221739**
armslist	0.118453	0.064179	0.150746	**0.189313**	**0.156716**	**0.214925**
dallasguns	0.125863	0.071014	0.155072	**0.222723**	**0.201449**	**0.233333**
elpasoguntrader	0.119025	0.061429	0.155714	**0.175496**	**0.135714**	**0.198571**
floridagunclassifieds	0.095586	0.042029	0.128986	**0.213165**	**0.191304**	**0.224638**
floridaguntrader	0.121979	0.065672	0.156716	**0.207233**	**0.174627**	**0.229851**
gunsinternational	0.126867	0.073913	0.163768	**0.205095**	**0.188406**	**0.211594**
hawaiiguntrader	0.117694	0.066197	0.145070	**0.208059**	**0.180282**	**0.223944**
kyclassifieds	0.110949	0.060870	0.130435	**0.191376**	**0.163768**	**0.207246**
montanagunclassifieds	0.060563	0.146479	0.208451	**0.233740**	**0.212676**	**0.245070**
msguntrader	0.111692	0.062857	0.128571	**0.209148**	**0.188571**	**0.222857**
nextechclassifieds	0.103477	0.054098	0.132787	**0.204046**	**0.177049**	**0.216393**
shooterswap	0.103645	0.057143	0.128571	**0.226965**	**0.205714**	**0.241429**
tennesseegunexchange	0.160429	0.108451	0.198592	**0.203350**	**0.180282**	**0.214085**
theoutdoorstrader	0.087652	0.032836	0.117910	**0.185680**	**0.159701**	**0.205970**

Table 6.4: Number of facts in the training, the validation, and the testing set and the MRR values obtained by the GAE on each background linked data

focused on the semantic relation inference, therefore the correct semantic types were already available. The accuracy of semantic models has been computed in terms of precision and recall, by comparing them with the ground-truth semantic models. If the correct semantic model of the source s is denoted as sm and the semantic model computed by the system is denoted as sm', precision and recall are defined by Taheriyan et al. [157] as follows:

$$precision = \frac{rel(sm) \cap rel(sm')}{rel(sm')} \tag{6.4}$$

$$recall = \frac{rel(sm) \cap rel(sm')}{rel(sm)} \tag{6.5}$$

where $rel(sm)$ is the set of triples (u, v, e): e is an object property from the ontology class u to the ontology class v. Table 6.11 reports the results in terms of precision and recall obtained by: (i) the SeMi system; (ii) the approach of Taheriyan et al. [157]) (Tahe in the Table); (iii) the baseline exploiting only the frequency of semantic relations of length 1 (Occs in the Table); (iv) the baseline using the steiner tree performed on a weighted graph based on the ontology structure (Stei in the Table).

In the evaluation experiment SeMi always obtained a better accuracy in terms of precision and recall, compared to: (i) the baseline that captures the frequency of semantic relations of length 1; (ii) the baseline of the steiner tree built on the graph weighted according to the ontology structure. The experiment employed the dataset in which the Taheriyan et al. [157] approach obtained the best results. The results show that SeMi outperformed the state of the art in case of the following data sources: "dallasguns", "floridagunclassifieds", "gunsinternational", and "shooterswap". These sources have the most complex structure in terms of number of nodes and links in the ground-truth semantic models (see Table 6.1 for more details). The accuracy improvement in these specific cases can be explained considering the different levels of features extraction. The system of Taheriyan identifies the best semantic relation considering two metrics: (i) cost and (ii) coherence. The cost of the semantic relation derived from a graph pattern is computed according to an inverse function of its popularity. As a consequence, computing the minimum cost is equivalent to select the most frequent semantic relation. On the other side, the coherence gives priority to longer semantic relations. From a different perspective, SeMi assigns to each plausible semantic relation a cost, aggregating the scores obtained by each RDF fact of the semantic relation. This score is computed by the DistMult factorization method (see Subsection 5.4) that takes as input the embeddings of the subject, the predicate, and the object of the RDF fact. Embeddings represent more granular and latent information that incorporate hidden regularities in the data that can not be detected adopting a manual approach for the extraction of features. For many other data sources, SeMi reached the accuracy in terms of precision and recall of the state of the art [157]. Therefore, SeMi is able to distinguish very well the correct relation, exploiting the neighborhood structure of the graph. On the other side, the performance of SeMi in terms of precision drastically lowed in presence of many data attributes within sources that are characterized by the same semantic type (see "elpasoguntrader" and "nextechclassifieds"). For instance, the "nextechclassifieds" source includes 5 different attributes that are labeled with

Sources	Precision				Recall			
	SeMi	**Tahe**	**Occs**	**Stei**	**SeMi**	**Tahe**	**Occs**	**Stei**
alaskaslist	**1**	1	0.667	0	**1**	1	0.667	0
armslist	**0.750**	0.750	0.500	0	**0.750**	0.750	0.500	0
dallasguns	**0.667**	0.570	0.500	0	**0.570**	0.570	0.428	0
elpasoguntrader	0.500	1	0.500	0.250	0.500	0.750	0.500	0.250
floridagunclassifieds	**0.833**	0.800	0.167	0	**0.833**	0.670	0.167	0
floridaguntrader	**1**	1	0.750	0	**1**	1	0.750	0
gunsinternational	**0.750**	0.600	0.250	0	**0.750**	0.750	0.250	0
hawaiiguntrader	**1**	1	1	0	**1**	1	1	0
kyclassifieds	**1**	1	0.333	0.333	**1**	1	0.333	0.333
montanagunclassifieds	0.750	1	0.500	0	0.750	1	0.500	0
msguntrader	**0.670**	0.670	0.667	0	**0.500**	0.500	0.500	0
nextechclassifieds	0.454	1	0.182	0	**0.454**	0.360	0.182	0
shooterswap	**1**	0.750	1	0	**1**	1	1	0
tennesseegunexchange	0.667	1	0.500	0.167	0.667	1	0.500	0.167
theoutdoorstrader	0.800	0.830	0.200	0.200	0.800	1	0.200	0.200

Table 6.5: Results of the semantic relation inference in terms of precision and recall

the ontology class “schema:Offer”. According to the ground-truth semantic model of this source, the class of the semantic type “schema:Offer1” is linked to the other 4 entities classes with the object property “schema:relatedTo”. Nevertheless, this type of graph structure represents an anomaly because it never appears in the semantic models of the other sources, that have been exploited for creating the background knowledge of “nextechclassifieds”. Including in the background linked data analogous graph structures, the performance of SeMi should increase.

To further understand the impact of GNN in SeMi , a specific analysis is conducted removing the GNN component from the GAE, performing the SRI task with the DistMult factorization method. Table 6.6 reports the results in terms of precision and recall of SeMi with GAE and SeMi with DistMult. The results show that in most cases SeMi with GAE performs better than SeMi with DistMult. In the case of the "elpasoguntrader" SeMi with DistMult outperforms SeMi with GAE. The main reason is related to the structure of this source which is not so common in the other datasets, so the GNN is not able to recognize this specific graph pattern in the background linked data.

Regarding the scalability issues, in their work, Taheriyan et al. [157] underlines that performing a single SPARQL query on Virtuoso [51] repository with more than three million of triple requires approximately one hour for the graph patterns of length five (Mac OS X System, 2.3 GHz Intel Core i7 CPU, 16 GB of RAM). This time is expected to grow exponentially as the dimension of semantic relations increase. The training of the GAE is the operation performed by SeMi , that specifically involves the background linked data, as the SPARQL queries performed by Taheriyan et al [157]. The training execution time is less than 30 minutes and does not encounter issues related to the growing of semantic relations complexity. The training step has been performed on a Centos 7 - OpenHPC 1.3 System, nVidia Tesla V100 SXM GPU, 32 GB of memory, 5120 cuda cores.

6.5 Evaluation based on the Relational-To-Ontology Mapping

The evaluation method described in this Section has been performed with RODI [122], benchmark suite for comparing SeMi against tools that map schemata of relational databases to ontologies. This method exploited the results achieved by a campaign of queries performed on a *reference* data source and a *test* data source. In details, RODI compares the results of the SQL queries performed on relational databases (the reference source) against the results of SPARQL queries performed on KGs generated by the tools of the benchmark and our tool (collectively considered the test source). SQL and SPARQL queries defined for the evaluation are *semantically equivalent*, as they intend to retrieve the same results. Within this evaluation procedure, SeMi has been compared against other systems for relational-to-ontology mapping, namely: Bootox [84], D2RQ [23], MIRROR [105], and Ontop [28].

The evaluation driven by query results has two main advantages. Firstly, it allows the inclusion of systems that generate KGs, exploiting so different languages and standards to describe semantic models of data sources. For instance, tools such

Sources	Precision		Recall	
	SeMi (GAE)	SeMi (DM)	SeMi (GAE)	SeMi (DM)
alaskaslist	1	1	1	1
armslist	0.750	0.750	0.750	0.750
dallasguns	**0.667**	0.500	**0.570**	0.428
elpasoguntrader	0.500	**0.750**	0.500	**0.750**
floridagunclassifieds	0.833	0.833	0.833	0.833
floridaguntrader	**1**	0.500	**1**	0.500
gunsinternational	0.750	0.750	0.750	0.750
hawaiiguntrader	1	1	1	1
kyclassifieds	1	1	1	1
montanagunclassifieds	**0.750**	0.500	**0.750**	0.500
msguntrader	0.670	0.670	0.500	0.500
nextechclassifieds	**0.454**	0.367	**0.454**	0.367
shooterswap	1	1	1	1
tennesseegunexchange	**0.667**	0.500	**0.667**	0.500
theoutdoorstrader	**0.800**	0.600	**0.800**	0.600

Table 6.6: Results in terms of precision and recall obtained by SeMi with GAE and DistMult

as MIRROR [105] adopt the R2RML mapping language[2], the D2RQ [23] platform relies on its own native language to define semantic models, while SeMi exploits a formalization based on the syntax of SPARQL (see Section 6.3.3). Secondly,

[2]A detailed description of R2RML is available at: https://www.w3.org/TR/r2rml/

mappings that correctly satisfy user needs -encoded as queries- are directly usable for a variety of purposes, as also stressed by Pinkel et al. [122][3].

Therefore, the evaluation approach described in this Section is based on the principle of *utility* of semantic modeling (as in [122]) rather than comparing generated semantic models directly to a reference semantic model[4]. The idea of utility is related to the practical reasons for which data engineers model the semantics of data sources: they are not interested in the analysis of semantic models, but they want to evaluate the resulting KGs, because these are the artifacts necessary for the end users' purposes.

6.5.1 Evaluation Procedure and Results

This Subsection describe the comparison of SeMi against other systems, with respect to relational-to-ontology mapping challenges. Such challenges are related to the structural differences between relational schemata and ontology and they have a direct impact in the generation of semantic models. Below are described which differences determine the main consequences for the semantic relation inference task:

1. *Artifacts Normalization*: in some cases relational schemata are optimized for update-intensive workloads and do not aggregate information at the ontology conceptual level. For this reason, semantic relations between two semantic types can be spread over different tables in relational schemata using many-to-many (n:m) relationships.

2. *Artifacts Denormalization*: in opposite cases schemata are optimized for read-intensive workloads. In this context, semantic relations between different semantic types are included in the same table of relational schemata.

3. *Key Conflicts*: identifiers in relational databases are usually implemented using primary keys and unique constraints, while ontologies use URIs. According to this principle, semantic relations are implemented in relational schemata using foreign keys.

Table 6.7 summarizes the challenges, reporting the relational schemata patterns causing them and the specific hurdles in building the semantic model.

[3]Pinkel et al. [122, p. 2]: "[w]hat matters at the end of the day in practice is whether the generated mappings are usable and useful for the task at hand. We therefore consider mapping quality as mapping utility with relation to a query workload posed against the mapped data"

[4]This method has been adopted in Section 6.4 for a better comparison with the state of the art

Challenge type	Relational database pattern	Specific difficulty
Artifacts Normalization	1:n relation	JOIN to relate entity IDs
	n:m relation	3-way JOIN to relate entity IDs
	Indirect n:m relation (using additional intermediary tables)	k-way JOIN to relate entity IDs
Artifact Denormalization	Correlated entities (in shared table)	Filter condition
Key conflicts	Missing references (no foreign key where relevant relations exist)	Unconstrained attributes as references

Table 6.7: Table of mapping challenges. Relational database patterns are also reported with specific difficulties in mapping challenge

Before providing details in the design of this evaluation procedure, the following Subsection describes how the RODI suite has been developed to measure how different systems address the above-mentioned challenges.

The RODI Benchmarking Suite

The RODI suite provides the so-called *benchmarking scenarios* to evaluate the capability of the systems to address relational-to-ontology challenges. Each scenario is composed of:

- a domain ontology as target of the semantic modeling process;
- a relational database where Pinkel et. al [122] injected specific adjustments reflecting relational-to-ontology challenges;
- a set of query pairs: a SPARQL query for KG generated from the domain ontology (test source) and a SQL query for the relational database (reference source). Each query pair is labeled with categories that summarize the high-level goal of the query in terms of semantic modeling. For instance, if a query pair is categorized with the labels “path-1” and “path-2”, it investigates whether the system is able to detect the correct semantic relation directly joining two tables (single JOIN) or using one intermediate label (double JOIN). Table 6.8 includes all query categories and the link with the mapping challenges.

For the evaluation purposes, the following changes have been introduced in the RODI benchmark suite:

- test sources expansion, including the KGs produced by SeMi ;

Category ID	Meaning	Mapping challenges
path-0	Semantic relation match that finds both related entities in the same table (1:1 or denormalized)	Artifacts Denormalization
path-1, path-2	Semantic relation match that finds related entities in two tables that can be directly joined (single JOIN) or joined through one intermediate table (two JOINs)	Artifacts Normalization, Artifacts Denormalization
path-n	Semantic relation match that requires n JOINs (n >2) to connect the tables that define entities on both sides	Artifacts Normalization
path-X	Additional tag for all queries that are tagged path-n, with any n >1 (denotes multi-hop JOIN of any length)	Artifacts Normalization
denorm	Type filtering required due to denormalization	Artifacts Denormalization
no-fk	JOINs without leading foreign keys	Key Conflicts

Table 6.8: Category labels used for each query pair with a link to the relevant challenge

- query pairs improvement, to increase the robustness of the evaluation;
- RODI's report fix, because the benchmaking tool reported only the number of rows contained in the queries output [5]. However, this type of result is not compatible with the scoring function and the metric defined below.

Scoring Function and Evaluation Metric

The scoring function reflects the utility of semantic models. This function compare the results of queries performed on a test source against the results of queries performed on a reference source. Based on this principle, a per-query F_1 score is computed and defined as the harmonic average of *precision* and *recall*:

$$F_1 = 2 * \frac{precision * recall}{precision + recall} \tag{6.6}$$

F_1 score assumes a value between 0 and 1 and its computation needs to satisfy the *structural equivalence* [123] of the results. To clarify how the F_1 score is computed, below is available an example of possible result sets obtained by a SPARQL query (the reference result set is obtained with a reference SQL query):

- Reference result set: municipality01, municipality02.

[5] A sample of queries included in RODI is available in the GitHub repository of Pinkel et. al [122]: https://github.com/chrpin/rodi/tree/master/data/cmt_mixed/queries. Indeed, most of the queries in RODI are written with the COUNT function that returns the number of rows that matches a specified criteria

- Result set A: municipality01, municipality02.
- Result set B: municipality01.
- Result set C: municipality01, municipality02, municipality03.

Computing values of precision and recall, the following results are achieved:

- Result set A (Precision: 1.0 - Recall: 1.0 - F_1: 1.0). It is structurally equivalent to the reference result set.
- Result set B (Precision: 1.0 - Recall: 0.5 - F_1: 0.6). It is equivalent only to a subset of the reference results, because it does not include municipality02.
- Result set C (Precision: 0.666 - Recall: 1.0 - F_1: 0.8). It lists all expected contracting authorities, but municipality03 is mistakenly classified as contracting authority by the tool.

According to this example, the tool A performs better than C and B in the semantic relation inference task.

Goal

The goal of the evaluation procedure is to compare by means of RODI the performance of SeMi against the following tools: Bootox [84], D2RQ [23], MIRROR [105], Ontop [28].

Data

RODI data employed for the evaluation procedure come from the domain of conferences. This domain has been selected for three main reasons:

- it is understandable also by non-domain experts;
- it is complex enough for testing systems in realistic cases;
- it has been successfully used also in other benchmarks [2].

The benchmark scenarios of this domain include 26 relational databases which vary in size and complexity. Each scenario runs between 6 and 10 query pairs.

Ontology	Number of classes	Number of data properties	Number of object properties
CMT	36	10	49
SIGKDD	49	11	17
Conference	38	23	13

Table 6.9: Target ontologies for the semantic modeling process

Ontologies

The ontologies included in the benchmark scenarios are provided by the Ontology Alignment Evaluation Initiative [2] (OAEI) and are developed by the OntoFarm project [151]. Within the conference domain, 3 different ontologies have been selected:

- CMT;
- SIGKDD;
- CONFERENCE.

As explained by Pinkel et. al [122] these ontologies are modeled on various views of the domain and their differences are based on different criteria: variation in size and modeling style, information cardinality (in particular, functionality of relationships), and the expressive power of the ontology language used. Table 6.9 shows the dimension of each ontology in terms of number of classes, number of data properties, and number of object properties.

Design

The evaluation is conducted by executing the following steps for each RODI scenario:

1. each tool is called by RODI to create the semantic model on the relational database, considering the target ontology;
2. each tool produces the KG based on the semantic model generated in the previous step;
3. SPARQL queries are performed against the KG (test source);
4. SQL queries are performed against the relational database (reference source);
5. the results are compared using the scoring function (see Section 6.5.1) for each query pair and then they are aggregated by means of the average in the light of different query categories.

At the end of this process, the RODI tool generates a report that includes the following results in terms of F_1 score, precision, and recall:

- the result of each query pair;
- the average of the results of query pairs, grouped according specific categories (see Section 6.5.1);
- the average of the results of all query pairs.

An example of report produced by RODI for a specific scenario is available in Table 6.10.

	F_1	**Precision**	**Recall**
Q42	1.0	1.0	1.0
Q43	1.0	1.0	1.0
Q46	0	0	0
path-3 (AVG)	0.5	0.5	0.5
All (AVG)	0.5	0.5	0.5

Table 6.10: Example of report generated by the RODI benchmarking suite. For each scenario, the report shows results of queries comparison in terms of F_1 score, precision, and recall showing also aggregated results according to the average dimension

Figure 6.6 shows the results obtained through the RODI benchmark. SeMi obtained better results with respect to other systems in queries categorized as path-n and path-X. On the other hand, Bootox exceeded the results of SeMi in queries categorized as path-0 and path1, path2[6]. Therefore, SeMi performed better in cases of normalized artifacts, where semantic relations are spread among different tables in a relational database. This means that, respect to other tools, the number of multiple JOINs does not affect on the inference of the correct semantic relations. In fact, once the semantic types have been detected, the semantic relations are established according to the the score computed by the DistMult factorization method (Section 6.3.4), that is independent from the structure of the input data source. On the other hand, Bootox performed better than SeMi in case of denormalized artifacts where semantic relations between different entities are included in the same table. The features of Bootox, that exploit lexical and structural matching, are more suitable in tackling this condition. In general, the F_1 values are quite low in the inference of semantic relations, and results underline the complexity of this task in the context of relational databases. In fact, systems such as MIRROR and Ontop never infer the correct semantic relation.

[6]The data is available on GitHub at https://github.com/giuseppefutia/semi/tree/master/data/rodi

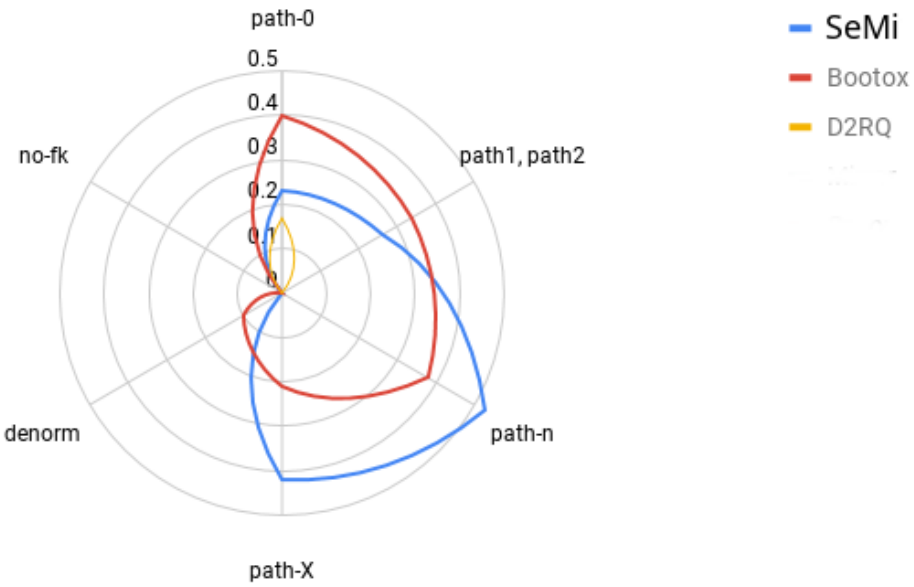

Figure 6.6: F_1 scores obtained by evaluated systems in RODI among different query categories

6.6 Evaluation in Public Procurement Scenario

This evaluation procedure investigates the results of the introduction of the GNN for the refinement of the initial semantic model in the field of public procurement domain. The analysis compares results of SPARQL queries performed on different KGs. The reference source is a KG generated through ground-truth semantic models constructed by domain experts from data of public contracts. On the other side, the test sources include the KG generated through semantic models built by the Semantic Model Builder (SMB) (see Subsection 6.2.3) and SeMi , which refines the initial semantic models produced by the SMB employing the GNN. The scoring function and the metric adopted for the comparison are described in Subsection 6.5.1. This evaluation method follows the same principles reported in Section 6.5. In fact, the aim of the evaluation process follows the main idea defined for the RODI benchmarking suite [122]. Such idea is focused on the *utility* of semantic modeling, rather than comparing semantic models directly to a reference semantic model. The idea of utility is related to the practical reasons for modeling the semantics of data sources: the main interest is not tied to analysis of semantic models, but it is focused on the evaluation of the resulting KGs, because

they contain the useful information for user purposes.

The experiment is based on the following steps: (i) KG generation (ii) SPARQL queries definition; (iii) F_1 score computation.

The input sources for the KG generation step are represented by 20 JSON files. These files are created from XML files, which are not compliant with the standard schema of Italian public procurement data (for more details see the Subsection 4.1.2). For each file, 3 different semantic models are built:

1. *GT-SM*: the ground-truth semantic model of the target source created by domain experts;
2. *SMB-SM*: the automatically-built semantic model of target source obtained from the initial semantic model, which is generated by the SMB;
3. *SeMi-SM*: the automatically-refined semantic model of the target source, which is generated by SeMi .

Exploiting these 3 semantic models built on input data sources, 3 different KGs are produced:

1. *GT-KG*: the ground-truth KG generated with the GT-SM of the 20 JSON files;
2. *SMB-KG*: the KG generated with the SMB-SM of the 20 JSON files;
3. *SeMi-KG*: the KG generated with the SeMi-SM on the 20 JSON files.

The following SPARQL queries have been performed against these 3 different KGs:

- SPARQL Query 1 (SQ1 - See Listing 4): it retrieves all business entities that have the role of contracting authority in all contracts within each KG;
- SPARQL Query 2 (SQ2 - See Listing 5): it retrieves all business entities that have the role of bidder of a tender for each contract within each KG[7];
- SPARQL Query 3 (SQ3 - See Listing 6): it retrieves all business entities that have the role of bidder of the awarded tender for all the contracts.

Table 6.11 reports F_1 values obtained comparing the information retrieved performing these 3 SPARQL queries, which allow to understand the behaviour of the systems for the inference of specific semantic relations.

[7]This SPARQL query is performed for each contract within each KG, to verify the correct assignment of the pc:bidder relationship for each contract. If all contracts are globally considered, there is the risk to obtain an high precision value also in cases where pc:bidder relation is erroneously assigned

```
PREFIX pc: <http://purl.org/procurement/public-contracts#>
SELECT ?businessEntity {
    ?contract pc:contractingAuthority ?businessEntity .
}
```

Listing 4: SPARQL query SQ1 to retrieve all contracting authorities in the KG

```
PREFIX pc: <http://purl.org/procurement/public-contracts#>
SELECT ?businessEntity {
    ?contract pc:tender ?Tender .
    ?Tender pc:bidder ?businessEntity .
}
```

Listing 5: SPARQL query SQ2 to retrieve all business entities that have the role of bidder of a tender for each contract

```
PREFIX pc: <http://purl.org/procurement/public-contracts#>
SELECT ?businessEntity {
    ?contract pc:awardedTender ?Tender .
    ?Tender pc:bidder ?businessEntity .
}
```

Listing 6: SPARQL query SQ3 to retrieve all business entities that have the role of bidder of the awarded tender for all the contracts

Table 6.11: F_1 values obtained comparing SPARQL results with the GT-KG

SPARQL	**SMB**	**SeMi**
SQ1	0.62	0.43
SQ2	0.34	**0.78**
SQ3	0.38	**0.66**

Chapter 7

Predicting New Links with GNNs: a Recommendation System Perspective

The goal of this Chapter is to measure the performance of the link prediction mechanism and understand its role for recommendation purposes.

7.1 Building the Dataset: Training, Validation, and Test Sets

The adoption of the GAE for the link prediction requires to prepare the dataset for the training and the evaluation phases, starting from the initial version of the PKG. In order to avoid noisy data that can potentially lead to wrong predictions, the first step is to remove from the PKG the facts that include entities that are characterized by ambiguous information, for instance the author keywords or the external contributors, that are identified by their name and surname. The following list includes the entities that are actually adopted to train and evaluate the model.

1. The publications, that are identified by the IRIS ID.
2. The authors, that are identified by the Polito internal ID.
3. The journals, that are identified by their ISSN.
4. The topics extracted by TMF, that are identified by the DBpedia URIs.

The Dataset Builder is implemented as a Python command-line script that assigns an integer (or index) representation to entities, classes, and properties of the KG. This step is necessary to obtain a numerical representation of the graph

data, which can be processed by the GAE. After this step, the script prepares the following input data for the Model Trainer:

1. The number of nodes that corresponds to the number of RDF entities in the graph.
2. The number of different object properties that link together the graph entities.
3. The number of node labels, where each label is referred to a class of the KG ontology.
4. A list of edges, represented as tuples composed of three elements: the node index of the subject, the node index of the object, and the index of the predicate.
5. A list of node-specific normalization constants.

To build such data structures starting from the RDF graph, the Dataset Builder leverages some look-up hash tables (implemented as Python dictionaries) that are created starting from the RDF statements and the ontology. Such tables, whose details are available below, are accessed by URIs, and allow to retrieve the corresponding entity or property index.

1. The *nodes table* is populated by assigning to each entity a unique and increasing integer index. Such index identifies the entity when building the list of edges. Only the entities that are instances of the classes that have been selected as part of the dataset are added to the table.
2. The *property table* is built starting from the PKG ontology, assigning to each property inside the graph a corresponding integer id.

The obtained list of edges is then splitted into three separate and disjoint sets that are used to train, evaluate, and test the GAE model. The splitting process has to ensure that the training set includes at least one edge for every kind of property: once such initial sampling has been done, the remaining training edges are randomly taken. This is mandatory to obtain meaningful embeddings, because the GAE model learns the vector representations of the nodes on the basis of their neighbours features. As a consequence, it is crucial to have a neighborhood structure in the training set which is similar to the one in the entire dataset. The percentage of tuples used to create the three sets is an hyperparameter that can be chosen in advance. However, an initial test showed that picking less then 70% of nodes for training does not allow to maintain a representative neighborhood for each node: the results are potentially inaccurate link predictions. To solve this issue, the edges list is splitted in approximately 90% of the tuples for train, 5% for

Table 7.1: Statistics of the dataset produced by the Dataset Builder.

Dataset			
Number of nodes	Number of classes/labels	Number of relations	Total number of edges
47,996	4	4	170,593
Number of train, evaluation and test samples			
Train edges	Evaluation edges	Test edges	
153,531	8,528	8,534	

the validation and 5% for testing. The three sets are serialized and stored, in order to mantain the same edges that are randomly picked during the training phase.

Table 7.1 summarizes the details on the dataset obtained by the Dataset Builder. Comparing the size of this dataset with the dimension of the initial PKG, whose details are reported in Table 4.4, more than half of the nodes have been removed. This nodes referred to the author keywords and to the external co-authors and contributors, which do not have unique identifiers within the IRIS system. Removing these entities prevent the inclusion of noisy data in the training process.

7.2 Training the GAE Model

The training edges are used to build a *training graph*, which is actually used for learning the node and property embeddings. The features used for the node are only based on the local neighborhood structure. Moreover, to increase the performance of the training, the node features are initialized on the basis of the node degree. Considering the size of the full graph, the training is performed in batches. At each training epoch, the Model Trainer randomly samples a subset of the training edges and, as a consequence, multiple epochs are required to train over all the nodes in the training graph. The training step is performed using the negative sampling approach. Each edge resulting from the sample process is a fact considered as a positive example. For each positive example, a certain number of negative samples is generated, corrupting the subject and the object with a random node index. This node index must not assume a value corresponding to a positive examples, in order to avoid that corrupted samples are considered positive example. This preparation phase is necessary, because the link prediction task is considered a binary classification problem, in which facts extracted from the training graph are considered as positive example, while corrupted facts are consider as negative examples for the training. The ratio defined for the number of negative samples, for each positive fact, is an hyperparameter of the GAE model. The results of this training process are the node and the property embeddings. In order to identify

the best parameters, represented by the embeddings, a separate validation set is evaluated during the training. The validation step is not performed at every epoch due to computation time constraints, and its frequency is an hyperparameter of the model. During the validation step, the best model is confirmed or replaced according to the obtained level of accuracy. At the end of the training process, the best model is loaded from memory and its accuracy is evaluated against the test set. The embeddings produced by the best model are directly used for the link prediction. Within the training process, the current bottleneck is represented by the sampling phase, that in the current implementation can not be perform on a GPU, because it is coded as single thread using Numpy [117].

7.3 The Link Evaluator

The Link Evaluator leverages the output of the trained GAE, in particular the entity and the property embeddings, in order to predict new and valuable facts that are not present in the PKG. The first step of the Link Evaluator is to generate a set of all the graph nodes, resulting by the merge of the training, the validation and the test sets. Then it creates the set of all possible facts by constructing, for each node, all the possible links to every other node in the graph. This is done by taking for each source node (subject) all the possible permutations of relations (predicates) and destination nodes (objects). However, the majority of such automatically generated facts is semantically invalid. For instance, among the permutations there are facts which connect together two publications using as predicate the relation *dc:subject*. These type of facts are is clearly meaningless, because they state that the main topic of a publication is another publication. For this reason, the facts that are not semantically valid with respect to the constraints of the ontology are immediately discarded. Once the invalid facts are removed, the remaining edges could be scored by applying the DistMult factorization. Then, the scored facts are grouped by subject and relation and sorted by their score in descending order. The facts with the highest score can be added to the PKG, in order to enrich its knowledge.

7.4 Evaluation and Results

The method adopted for the evaluation stage is performed on the test set and measure the accuracy according to which the facts are correctly scored. The metrics adopted to measure the accuracy level are the following:

1. The Reciprocal Rank (RR), which is computed for a single evaluated edge (s_i, p_i, o_i) as $1/rank_{(s_i,p_i,o_i)}$

2. The Mean Reciprocal Rank (MRR), which is the average of the RR of all the evaluation edges. The obtained value defines the accuracy of the entire model.

3. The Hits-at-N (HITS@N), which is the number of facts for whom the computed rank is between 0 and N.

To clarify this metrics, consider the following example. A triple ranked in the tenth position of the sorted list of scores obtain a RR value of 0.1. If the facts in the evaluation set are 100, with half of them ranked in first position and the other half in the tenth, the obtained MRR is $\frac{1}{100}\sum_{i=1}^{100}\frac{1}{rank_{(s_i,r_i,o_i)}} = 0.55$, while the HITS@1 is equal to 50, and the HITS@10 is equal to 100. The evaluation stage has been performed in batches.

7.4.1 Hyperparameters Validation

Table 7.2 shows the evaluation results in terms of the MRR metric of the GAE model, adopting different hyperparameters. In particular, different combinations of the learning rate and the regularization have been adopted to obtain the best MRR value. According to the results, the best values of MRR are obtained with 0.001, for the learning rate, and 0.5 for the regularization. In the round brackets of the table including results on the validation test are reported the results obtained only with DistMult. These results demonstrate that the GNNs play a fundamental role to improve the accuracy of traditional link predictions.

The high value of the regularization parameter, compared to the value that is traditionally adopted in literature for the adopted GAE model (which corresponds to 0.01), is particularly interesting. The motivation of this result is that the PKG is characterized by a small number of different relations, compared to other graphs, that are usually adopted for benchmarking tasks of GAE models. This implies a less variety in the features used to generate the embeddings. In this regard, the regularization parameter helps the model to differentiate the embeddings obtained, even if there are few relations, and limits the overfitting on common features.

Regarding the MRR value obtained for the best model, it can be interpreted as the fact that, in average, the evaluated facts are ranked in the *eleventh* position. The MRR available in literature corresponds to 0.158 over the FB15k-237 dataset, which is commonly used as benchmark for the evaluation of link prediction model. The MRR obtained in this case correspond to a ranking that, in average, localized the correct facts in the sixth or seventh position. Therefore, the obtained result for the PKG is comparable with the results obtained over the benchmarking dataset.

Moreover, considering the number of nodes in the PKG dataset, the result obtained is particularly noteworthy, given that the model is able to rank (in average) the true triple in the eleventh position, among 47,995 corrupted facts, while in FB15k-237 for each true fact only 14,951 corrupted ones are generated.

Table 7.2: Evaluation of different combinations of learning rate and regularization parameters using as benchmark the MRR value. The first table shows the MRR of the best model found during the training phase. The second table shows the MRR obtained by the best model over the test set.

(a) MRR of the best model found during training.

		Learning Rate				
	MRR	**0.05**	**0.01**	**0.005**	**0.001**	**0.0005**
Regularization	**1.0**		0.0623	0.0504	0.0823	0.0793
	0.5		0.0677	0.0718	**0.0924**	0.0865
	0.1	0.0004	0.0703	0.852	0.0764	0.0706
	0.05		0.0707	0.0763	0.0673	
	0.01		0.062	0.0725		

(b) MRR obtained when evaluating the best model over the test set.

		Learning Rate				
	MRR	**0.05**	**0.01**	**0.005**	**0.001**	**0.0005**
Regularization	**1.0**		0.0611	0.0497	0.081	0.079
	0.5		0.0662	0.0723	**0.0882**(0.008)	0.0818 (0.006)
	0.1	0.0002	0.0693	0.0827(0.001)	0.0726	0.0666
	0.05		0.0687	0.0726	0.0654	
	0.01		0.0589	0.07		

The Figure 7.1 shows how the facts within the test set are distributed over the possible ranking values. As can be saw, the GAE model was able to correctly assign an high rank to most of the correct facts.

7.4.2 Validation with Different Number of Research Topics

Large part of the entities included in the PKG are represented by research topics, which correspond to the *TMFResource* instances, connected to the publications by means of the *dc:subject* property. Table 4.4 shows that almost a third of the PKG edges connects a publication to a topic extracted by TMF. Considering the

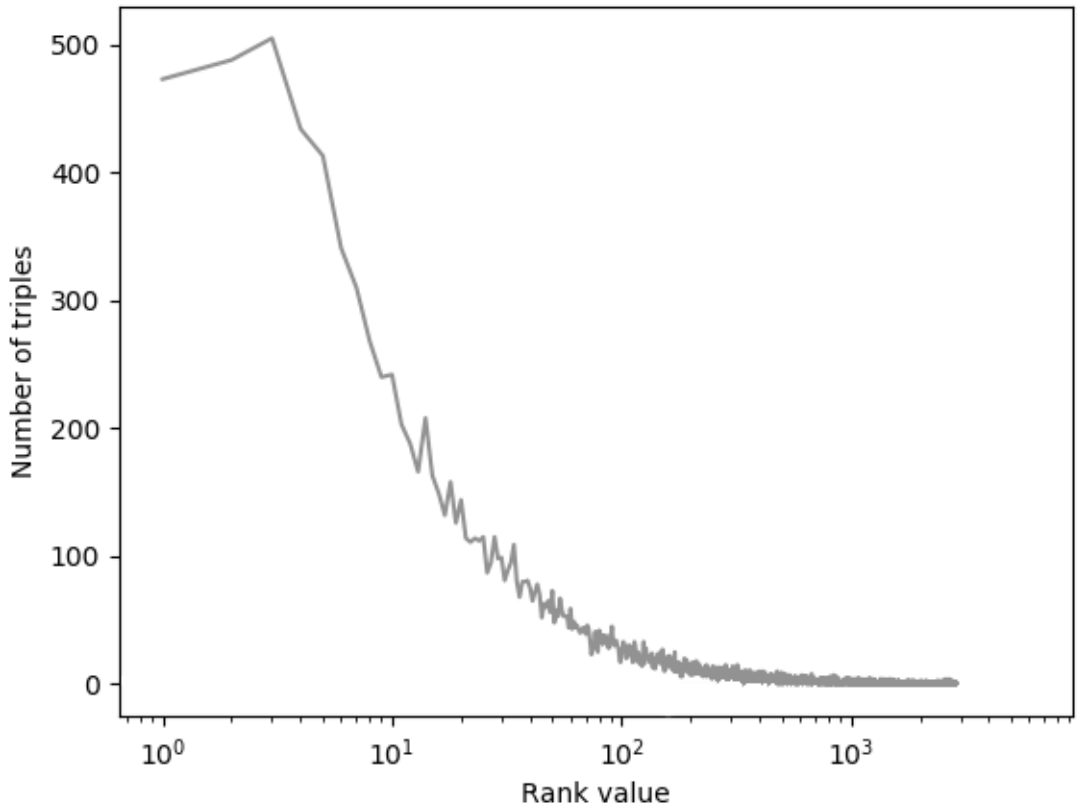

Figure 7.1: Number of facts for each ranking position obtained by evaluating the best model found over the test set. The ranking positions range from 1 to 47,995.

potential impact of the number of these entities in the link prediction mechanism, a second experiment has been conducted to evaluate the impact of different number of research topics within the PKG. Three different versions of the PKG have been created, which include for each publication three, seven, and fourteen topics respectively. The experiment employs the same learning rate and regularization hyperparameters, which obtain the best results in the previous experiment.

Table 7.3: Impact of the number of topics present in the RDF graph on the accuracy of the trained model.

Number of topics extracted per-abstract	**3**	**7**	**14**
Number of *TMFResource* entities in the graph	9.591	16.988	26.541
Number of *dc:subject* edges	45.423	107.093	212.226
MRR over test data	0.103	0.0882	0.0671
HITS@15	31.5%	27.9%	20.3%

Table 7.3 shows that the best results are obtained within the PKG which include only 3 topics. According to this result, decreasing the number of topics seems to obtain an improvement in the evaluation result. However, this not directly lead to a better link prediction model. In fact, the better ranking is a consequence of the less complex graph structure. This version of the PKG includes almost half of the topic entities and, as a consequence, it is an easy task for the model ranking the correct facts in a higher position. Moreover, due to the low number of topics and the features of TMF, there is a high number of publications that share the same topics. Therefore, it is not difficult for the model predicting facts that include such topics during the evaluation stage, over the test set. On the other side, increasing the number of topics within the PKG lead to a degradation of the MRR values. This behaviour is a consequence of the more complex graph structure. Moreover, increasing the number of facts (and edges) which include the predicate *dc:subject* has an impact on the neighborhood representation encoded by the GAE model. This bias in the neighborhood structure determines closer values in the embedding representation of the nodes and can conduct to more inaccurate prediction during the evaluation phase of the link prediction.

7.4.3 Results of a Sample-Based Validation

A sample-based validation confirms that the model is able characterize the graph entities and predicting meaningful facts. Consider, for instance, the publication entitled "Real-Time Tools for Situational Awareness and Emergency Management in Transport Infrastructures"[1]. The GAE model predicts a new edge, which connects this publication with the DBpedia research topic `http://dbpedia.org/resource/List_of_software_reliability_models`, through the predicate *dc:subject*. This topic is not recognized by TMF during the building of the initial version of the PKG, however it looks an appropriate topic suggestion for the publication. Moreover, the GAE model is able to predict another edge that connects this publication to a researcher, whose main interest is focused on circuits integration for aircraft application. Also in this case, the prediction can be considered valid and potentially useful for the researcher.

[1]This publication is available in the Geranium Web application at: `https://geranium.polito.it/results/paper/11583-2584393`

Chapter 8

On the Integration of Knowledge Graphs into Neural Networks for an eXplainable AI

This Chapter presents a review of the main eXplainable Artificial Intelligence (XAI) approaches existing in the literature, underlying their strengths and limitations, and propose neural-symbolic integration as a cornerstone to design an AI which is closer to non-insiders comprehension. Most of the approaches developed in this field, in fact, require very specific technical expertise to manipulate the algorithms at the roots of the modern implementation of Neural Networks (NNs). Moreover, understanding this algorithmic scaffolding is not enough to get insights into internal working models. In fact, in order to be more understandable, neural-based systems should be able to emit and manipulate symbols, enabling user explanations on how a specific result is achieved [45]. In the context of symbolic systems, Knowledge Graphs (KGs) [50] and their underlying semantic technologies are a promising solution for the issue of understandability [145]. In fact, they provide a useful backbone for several reasoning mechanisms, ranging from consistency checking [30], to causal inference [113]. These reasoning procedures are enabled by ontologies [66], which provide a formal representation of semantic entities and properties relevant to a specific sphere of knowledge. Considering these features, the implementations of symbolic systems based on semantic technologies are suitable to improve explanations for non-insiders. Input features, hidden layers and computational units, and predicted output of neural models can be mapped into entities of KGs or concepts and relationships of ontologies (*knowledge matching*). Traditionally, these ontology artifacts are the results of conceptualizations and practices adopted by experts from various disciplines, such as biology [47], finance [13], and law [29]. As a consequence, they are very comprehensible to people with expertise in a specific domain (*cross-disciplinary explanations*), even if they do not have skills in AI technologies. Moreover, in the context of semantic technologies,

KGs and ontologies are natively built to be queried and therefore they are able to provide answers to user requests (*interactive explanations*) and to provide a symbolic level to interpret the behaviour and the results of a NN. A schematic graph that summarises the role of semantic technologies for XAI is available in Figure 8.1.

This Chapter includes the following sections. Section 8.1 outlines the technical issues related to traditional methods of XAI, describing current solutions that require expertise on NN techniques. Section 8.2 provides details on the three promising research challenges (knowledge matching, cross-disciplinary explanations, and interactive explanations) enabled by the integration of KGs and ontologies into NN models.

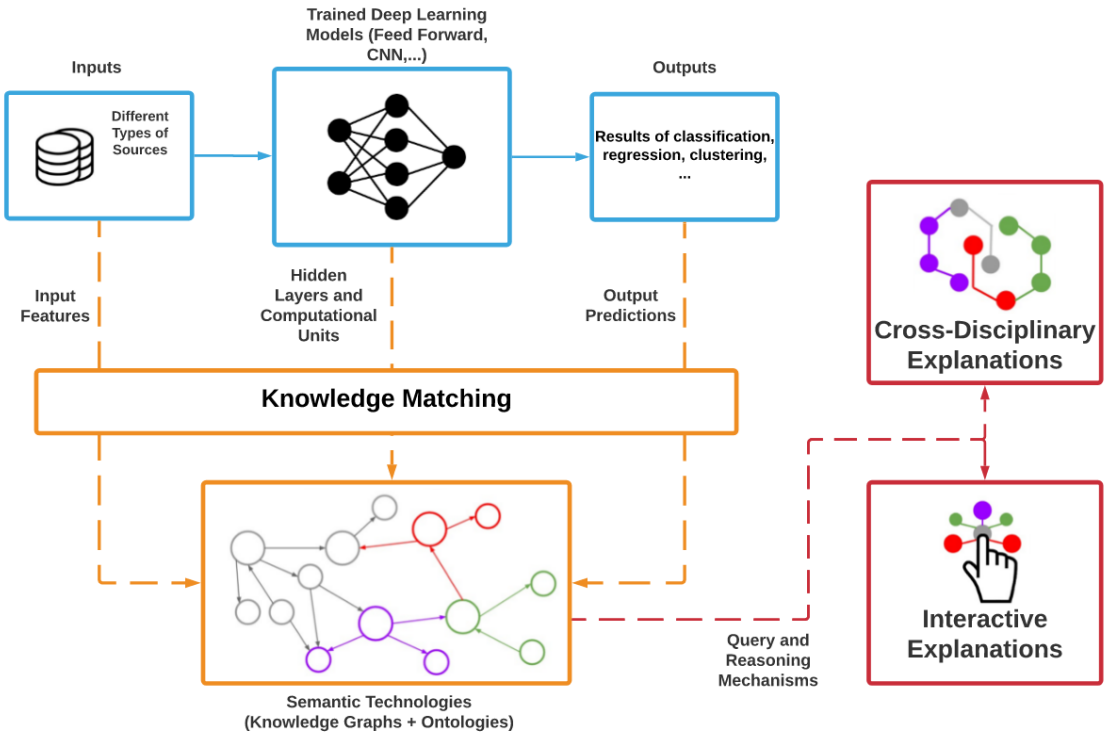

Figure 8.1: Schematic representation of a XAI system that integrates KGs and ontologies into NN models.

8.1 Explanations for AI Experts: Technical Issues and Solutions

The goal of an explainable system is to expose intelligible explanations in a human-comprehensible way, and keeping the human in the loop is a determinant aspect. Nevertheless, as clearly demonstrated in the literature survey conducted by Adadi et al. [3], the human factor impact is not adequately considered XAI.

Concordantly, Miller et al. [110] argue that most of the existing methods focus on the perspective of technicians rather than the viewpoint of the intended users.

This condition creates a gap of expectations of the layman in terms of explainable systems, because the term "explanation" has been re-purposed by the connectionist community to address specific technical issues. Instead, other disciplines collectively considered as "explanation sciences" [111], from cognitive science to philosophy, are rarely included in the current XAI research.

The following subsections discuss the technical issues related to XAI, and provide a brief overview of the typical approaches adopted by AI experts for the explanation task.

8.1.1 Technical Issues in a Connectionist Perspective

This subsection synthesize the technical issue related to the explainability as follows: (i) complexity; (ii) multiplicity; (iii) opacity degree.

Complexity: NN techniques are difficult to examine, because of their structure and the way they are working. As reported by Adadi [3], since NN algorithms are based on high-degree interactions between input features, the disaggregation of such functions in a human understandable form and with human meaning is inevitably more difficult.

Multiplicity: considering the complex network architecture of NN techniques, these techniques might produce multiple accurate models from the same training data. Hall and Gill [68] define this issue as "multiplicity of good models". In fact, the internal paths through the network are very similar, but not the same, and consequently the related explanations can change across different models. This issue clearly emerges in case of *knowledge extraction* techniques [85, 136]. The explanation in this case consists in the knowledge acquired and encoded as the internal representation of the network during the training.

Opacity Degree: a well-known problem in the AI research field is the trade off between interpretability and accuracy. As underlined by Breiman [25], achieving high accuracy often requires more complex prediction methods, and vice versa simple and interpretable functions do not provide accurate predictors. Considering this issue, the traditional approach is to construct complex models to reach a high accuracy and then adopt reverse engineering techniques to extract explanations, thus without the necessity of knowing in details the inner works of the original model [3].

8.1.2 Explainable Systems for AI Experts

Considering the technical issues mentioned in the previous subsection, two main explanation methods from literature are identified: transparency and post-hoc [111]

explanations. Techniques included in these methods can support AI experts in sensitive conditions.

Transparency Explanations: these types of methods are focused on how a model work internally and, as underlined by Lepri et al. [97], the explanation can be rendered at three different levels: (i) the entire model, (ii) the individual components (e.g., parameters) of the model, (iii) the specific training algorithm. For each level, Mittelstadt et al. [111] recognize respectively the following notions in terms of explanation:

1. **Simulatability**: checking through a heuristic approach whether a human reaches the mechanistic understanding of how the model functions, and consequently if he is able to simulate the decision process. In this context, within a user study [55] that involved thousand participants, Friedler et al. measured human performance in operations that mimic the definition of simulatability, using as evaluation metric the runtime operation count.

2. **Decomposability**: in this case each component of the model, including a single input, parameter, and computation has to be clearly interpretable. In a recent work, Assaf et al. [10] introduce a Convolutional Neural Network (CNN) to predict multivariate time series, in the domain of renewable energy. The goal is to produce saliency maps [4] to provide two different types of explanation on the predictions: (i) which features are the most important in a specific interval of time; (ii) in which time intervals the joint contribution of the features has the greatest impact.

3. **Algorithmic transparency**: for techniques such as linear models there is a margin of confidence that the training will converge to a unique solution, so the model might behave in an online setting in an expected way. At the opposite, NN models cannot provide guarantees that they will work in the same way on new problems. Datta et al. [38] designed a set of Quantitative Input Influence (QII) for capturing the joint influence of the inputs on the outputs of an AI system, with the goal to produce transparency reports.

Post-hoc Explanations: these methods do not seek to reveal how a model works, but they are focused on how it behaved and why. Lipton [100] detects different post-hoc approaches that include natural language explanations, interactive visualizations, local explanations, and case-based explanations. Natural language explanations are based on qualitative artifacts that describe the relationships between the features of the input data and the outputs (e.g., predictions or classifications) of the model [129]. Interactive visualizations show relative influence of features or provide graphical user interfaces to explore visual explanations [159]. Local explanations intend to identify the behavior of a NN model on a particular prediction in two different ways—a simple and local fitting around a particular

decision [129], and variables perturbations to understand the changes in the prediction [5]. Case-based explanations consist in the exploitation of the trained model to identify which samples of the training data are the most similar to the prediction or the decision to be explained [87].

Despite this variety of approaches to address technical issues related to the explainability, these methodologies are not intended to capture the full behavior of the system, but rather to provide approximations of the system behavior. As stressed by Mittelstadt et al. [111], these approximations can be offered to AI experts both for "pedagogical purposes" and to provide reliable predictions on the system behavior over a restricted domain. Nevertheless, technical approximations are not enough and can be misleading when presented as an explanation to the non-insiders on how the model works.

8.2 Explanations for Non-insiders: Three Research Challenges with Symbolic Systems

In his prominent work, Miller [110] defines explanations as social conversation and interaction for transfer knowledge. A fruitful exchange implies that who explains must be able to recognize the mental model of who receives the information. To enable this process, a representation of the world through symbols is a crucial requirement. Furthermore, Miller articulates that this social exchange can be enabled in the XAI only if human sciences, such as philosophy, psychology, and cognitive sciences are injected within the development of new XAI approaches. The final aim is in fact is to produce explanations that allow "affected parties, regulators and more broadly non-insiders to understand, discuss, and potentially contest decisions provided by black-box models" [3]. Considering these requirements, symbolic systems open opportunities in three different human-centric challenges: (i) knowledge matching, (ii) cross-disciplinary explanations and (iii) interactive explanations.

The identification of these three challenges is based on the analysis, synthesis and elaboration of the most recent advances in the field of incorporating KGs features into NN models. This corpus of evidence is revised in accordance with the aim of building AI systems able to provide comprehensible explanations for non-insiders, through manipulation of symbols. For each challenge, recent research works representing the most promising references are reported. Therefore, further work along these tracks should be encouraged and supported.

Knowledge Matching: Seeliger et al. [145] identify the matching of input features or internal neurons of the models to classes of an ontology or entities of a KG as an important challenge to be addressed by the research community of XAI. Interesting works in this sense have been proposed by Sarker et al. [142], in which objects within images are mapped to the classes of the Suggested Upper Match Ontology. On the basis of the classification output of the NN, a description logic (DL)

learner is adopted to create class expressions that work as explanations. In a recent work, Angelov et al. [8] introduce a novel neural architecture for image classification, that includes an hidden semantic layer, called "Prototype layer", to provide a clear explanation of the model. This intermediate semantic layer receives training data samples, defined as *prototypes*, that are characterized by local peaks in the data distribution. Each prototype is then exploited to generate logical rules to provide natural language explanations. From a different perspective, Selvaraju et al. propose a method to learn a map between neurons weight to semantic domain knowledge [146]. In a work more focused on unsupervised learning, Batet et al. [14] exploit WordNet [109] and its taxonomic knowledge to compute semantic similarities that conduct to more interpretable clusters. In the context of transfer learning, Geng et al. [62] exploit two external KGs in a neural architecture for the following purposes: (i) provide explanations to understand the transferability of features learned by a CNN between different domains; (ii) justify new classes predicted by a Graph Convolutional Network (GCN), that were unseen by the CNN.

Cross-disciplinary Explanations: ontologies and KGs are able to represent domains by means of symbols, whose manipulation produces transparent inferences of new information. Both implementations are able to outline different areas of human knowledge according to characteristics and expertise of the related users. Moreover, it is worth mentioning that the concept of ontology is adopted by information science and philosophy. The stretch in common within the two different disciplines is the attempt to define ideas and entities within a precise system of categories, that explicit interdependent properties and relationships. Therefore, applied ontologies can be considered a technical application to prior work in philosophy. In a work entitled “The Knowledge Graph as the Default Data Model for Machine Learning”, Wilcke et al. [175] describe how decades of work have been devoted to the development of vast and distributed KGs to represent various domains of knowledge. Potentially, the usage of this form of interlinked and structured data enables the training of NN models from different domains and from the perspective of different disciplines. In this context, Sopchoke et al. [148] developed a method for explainable rules in recommendation systems related to different domains, using relational learning techniques on semantic statements.

Interactive Explanations: in a human-centered vision of explainable tools, AI systems should be able to offer user interaction features rather than static explanations. Wang et al. [172] developed a NN that extracts image contents as KG facts that are interlinked with the DBpedia repository [96], and questions provided by the user are translated in SPARQL (SPARQL Protocol and RDF Query Language) queries that are run over this enriched knowledge base. Liao et al. [98] propose a recommendation system that enable user-feedback on human-interpretable domain concepts. More in general, in recommendation systems the ontology is able to provide information that is implicit in the data used to perform inference and consequently to create rules which limit the number of plausible recommendations.

For future developments, Sarker et al. [142] envision their explanation tool for image classification to be used in an interactive system where a human can monitor and fix algorithmic decisions based on the given explanations.

Chapter 9

Conclusions and Future Work

Knowledge Graphs (KGs) are labeled and directed multigraphs that encode information in the form of entities and relations relevant to a specific domain or organization. KGs are practical tools for capturing and organizing a large amount of structured and multi-relational data to explore employing query mechanisms. Considering these features, KGs are becoming the backbone of Web and legacy information systems in different research fields and industrial applications. The structured knowledge that shapes KG can be supported by deductive and inductive reasoning approaches. Deductive methods employ simple statements and quantified statements to derive additional knowledge. Inductive techniques involve simple and quantified statements to create further knowledge and discover and generalize patterns available in the KG. These patterns can be inferred by applying statistical learning methods on multi-relational data, which are less interpretable than deductive approaches. However, they are capable of exploiting latent factors in the KG that are not directly extracted as quantities in the data, but whose variations influence every single piece of information we are able to observe. Graph Neural Networks (GNNs) are representation learning techniques, which are natively-built for encoding graph structures and are enabling cutting-edge research on graph data.

The thesis's primary goal is to investigate the role of neural architectures, particularly the GNNs, to support the publication of KGs. More precisely, this thesis intends to address open research problems (RP) in the KG research field: (**RP1**) the automatic mapping of data source schemas to reference ontologies. The mapping step is framed into the generation of semantic models, which includes the detection of semantic types and the semantic relation inference between these annotated attributes; (**RP2**) the automatic completion of existing KGs by inferencing soft, but consistent knowledge in terms of new edges (or links). Such new edges are hard to encode into deductive and logic-based reasoning, but they are beneficial to develop tools on top of KGs, e.g., recommendation systems. Considering these open problems, the contributions reported in this thesis addressed the following research questions:

- **RQ1** - Which is the contribution of NN to improve the accuracy of automatically-inferred semantic relations between the source attributes?
- **RQ2** - Which is the impact of GNNs in the prediction of new edges within a KG, in the context of recommendation systems?

For answering RQ1, the thesis illustrated two main contributions. The first contribution is an initial study [59] that applies a simple, but efficient language model such as Word2Vec, on SPARQL queries performed on different KGs. This study aims to learn the latent representations of SPARQL variables, which are included in specific triple patterns within the query. Variables with a similar latent representation are then labeled with the most common relations available in the triple patterns set. Then, the syntactic closeness between such labeled variables and the attributes is exploited to create a mapping between these elements. Consequently, the correct semantic relations between the data source attributes are assigned. The main limitation is that such an approach does not take full advantage of the graph structure for the learning process: SPARQL queries include a limited number of graph patterns and Word2Vec treats these patterns as plain text. Considering these limits, a more in-depth investigation has been conducted, developing a tool called SeMi (SEmantic Modeling machIne) [57], which employs a novel method based on a Graph Auto-Encoder (GAE), trained on available multi-relational data repositories. The GAE's encoder component is a GNN, while the decoder component is a factorization technique for KG embedding (KGE) generation. The achieved results show that SeMi outperforms the state-of-the-art approach [157] based on manually-selected features within the SRI task. Furthermore, the study shows that the GNN plays a fundamental role in this task because it outperforms the results obtained only with the KGE techniques. SeMi has been adopted to support the publication of large-scale KGs in the context of public procurement domain. The results achieved in this scenario show that the integration of public contracts data enables to detect and address consistency issues within the information released by public administrations. The most relevant inconsistencies are related to: (i) business entities with more than one business name; (ii) CIGs that identify more than one contract; (iii) incoherent payments among different versions of an ongoing contract. These issues can be fixed by exploiting the peculiar feature of KGs, overcoming the fragmentation which characterizes the public procurement information. For answering RQ2, the thesis illustrated a contribution that analyzes the GAE's adoption to predict new links, which can be useful in the context of recommendation systems in the field of academic publications. This approach supported the development of Geranium, a semantic platform to collect and organize the scientific knowledge of the Politecnico di Torino (Polito). The research achievements obtained in this frame are the following: (i) a novel academic KG that semantically connects information on researchers and publications of Polito; (ii) a semantic search engine that aggregates such information and enables advanced features for the content exploration;

(iii) a recommendation system which suggests collaboration opportunities between researchers of different departments.

Future developments for the contributions presented in this thesis are related to implementing new encoder/decoder architecture within the GAE, exploiting more complex GNNs as encoders and other types of KGE techniques as decoders. Further work will be dedicated to the computation of semantic models or the prediction of new links in the context of constant evolution of the KG. This feature can be particularly useful in real-world situations, in which new data sources are included in a continuous integration and enrichment process. A further indication for future development concerns developing an interactive user interface that enables users to intervene in each step of the semantic model construction or the link prediction. For this issue, a valuable solution is presenting a list of candidates as output of each block, instead of the most plausible output from the implemented algorithms. In regards to developing new research trajectories, this thesis analyzed the opportunities in combining deductive and inductive techniques for emergent research fields, such as the eXplainable AI (XAI) [56]. Indeed, the application domains such as those analyzed in this thesis — public procurement and academic publications — are contexts where the impact of neural architectures is relevant: the interpretability of the results is not only a desirable property, but it is a fundamental requirement for the involved stakeholders. Nevertheless, most of the available approaches to implement XAI focus on technical solutions usable only by experts, which are able to understand and manipulate the computational architectures of NNs. A complementary approach could incorporate deductive methods, which can exploit the symbolic representation of KG for inference new logic-based knowledge. Within such a general direction, the thesis proposes three specific challenges for future research—knowledge matching, cross-disciplinary explanations, and interactive explanations. Further work along these tracks should be encouraged and supported to make explanations of AI systems outputs more inclusive and effective.

Bibliography

[1] Ahmed Abd el Moneim Kamal, Yasser Omar, and Atef Zaki Ghalwash. "Simultaneous mapping of multi RDB to RDF". In: *2016 7th International Conference on Computer Science and Information Technology (CSIT)*. IEEE. 2016, pp. 1–5.

[2] Manel Achichi et al. "Results of the ontology alignment evaluation initiative 2016". In: *OM: Ontology matching*. No commercial editor. 2016, pp. 73–129.

[3] Amina Adadi and Mohammed Berrada. "Peeking inside the black-box: A survey on Explainable Artificial Intelligence (XAI)". In: *IEEE Access* 6 (2018), pp. 52138–52160.

[4] Julius Adebayo et al. "Sanity checks for saliency maps". In: *Advances in Neural Information Processing Systems*. 2018, pp. 9505–9515.

[5] Philip Adler et al. "Auditing black-box models by obscuring features". In: *arXiv preprint arXiv:1602.07043* (2016).

[6] Jose María Álvarez et al. "Query Expansion Methods and Performance Evaluation for Reusing Linking Open Data of the European Public Procurement Notices". In: *Advances in Artificial Intelligence: 14th Conference of the Spanish Association for Artificial Intelligence, CAEPIA 2011, La Laguna, Spain, November 7-11, 2011. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 494–503. ISBN: 978-3-642-25274-7. DOI: 10.1007/978-3-642-25274-7_50.

[7] Waleed Ammar et al. "Construction of the literature graph in semantic scholar". In: *arXiv preprint arXiv:1805.02262* (2018).

[8] Plamen Angelov and Eduardo Soares. "Towards Explainable Deep Neural Networks (xDNN)". In: *arXiv preprint arXiv:1912.02523* (2019).

[9] Simone Angioni et al. "AIDA: a Knowledge Graph about Research Dynamics in Academia and Industry". In: *Submitted to International Semantic Web Conference*. 2020.

[10] Roy Assaf and Anika Schumann. "Explainable deep neural networks for multivariate time series predictions". In: *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. AAAI Press. 2019, pp. 6488–6490.

[11] Ram G Athreya, Axel-Cyrille Ngonga Ngomo, and Ricardo Usbeck. "Enhancing Community Interactions with Data-Driven Chatbots–The DBpedia Chatbot". In: *Companion Proceedings of the The Web Conference 2018*. International World Wide Web Conferences Steering Committee. 2018, pp. 143–146.

[12] Ivana Balažević, Carl Allen, and Timothy M Hospedales. "Tucker: Tensor factorization for knowledge graph completion". In: *arXiv preprint arXiv:1901.09590* (2019).

[13] S Banerjee. "A Semantic Web Based Ontology in the Financial Domain". In: *Proceedings of World Academy of Science, Engineering and Technology*. 78. World Academy of Science, Engineering and Technology (WASET). 2013, p. 1663.

[14] Montserrat Batet et al. "Semantic Clustering Using Multiple Ontologies". In: *CCIA*. 2010, pp. 207–216.

[15] Dave Beckett and Brian McBride. "RDF/XML syntax specification (revised)". In: *W3C recommendation* 10.2.3 (2004).

[16] David Beckett. "Rdf 1.1 n-triples". In: *URL: https://www. w3. org/TR/n-triples* (2014).

[17] Yoshua Bengio, Aaron Courville, and Pascal Vincent. "Representation learning: A review and new perspectives". In: *IEEE transactions on pattern analysis and machine intelligence* 35.8 (2013), pp. 1798–1828.

[18] Tim Berners-Lee and Dan Connolly. "Notation3 (N3): A readable RDF syntax". In: *W3C team submission* 28 (2011).

[19] Tim Berners-Lee, Roy Fielding, Larry Masinter, et al. *Uniform resource identifiers (URI): Generic syntax*. 1998.

[20] Tim Berners-Lee, James Hendler, and Ora Lassila. "The semantic web". In: *Scientific american* 284.5 (2001), pp. 34–43.

[21] Stefan Bischof et al. "Mapping between RDF and XML with XSPARQL". In: *Journal on Data Semantics* 1.3 (2012), pp. 147–185.

[22] Christian Bizer, Tom Heath, and Tim Berners-Lee. "Linked data: The story so far". In: *Semantic services, interoperability and web applications: emerging concepts*. IGI Global, 2011, pp. 205–227.

[23] Christian Bizer and Andy Seaborne. "D2RQ-treating non-RDF databases as virtual RDF graphs". In: *Proceedings of the 3rd international semantic web conference (ISWC2004)*. Vol. 2004. Proceedings of ISWC2004. 2004.

[24] Antoine Bordes et al. "Translating embeddings for modeling multi-relational data". In: *Advances in neural information processing systems*. 2013, pp. 2787–2795.

[25] Leo Breiman et al. "Statistical modeling: The two cultures (with comments and a rejoinder by the author)". In: *Statistical science* 16.3 (2001), pp. 199–231.

[26] Dan Brickley, Ramanathan V Guha, and Andrew Layman. "Resource description framework (RDF) schema specification". In: (1999).

[27] Davide Buscaldi et al. "Mining scholarly data for fine-grained knowledge graph construction". In: *CEUR Workshop Proceedings*. Vol. 2377. 2019, pp. 21–30.

[28] Diego Calvanese et al. "Ontop: Answering SPARQL queries over relational databases". In: *Semantic Web* 8.3 (2017), pp. 471–487.

[29] Pompeu Casanovas et al. "Semantic web for the legal domain: the next step". In: *Semantic Web* 7.3 (2016), pp. 213–227.

[30] Melisachew Wudage Chekol et al. "Marrying uncertainty and time in knowledge graphs". In: *Thirty-First AAAI Conference on Artificial Intelligence*. 2017.

[31] Jiaoyan Chen et al. "Colnet: Embedding the semantics of web tables for column type prediction". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. 2019, pp. 29–36.

[32] Paolo Cifariello, Paolo Ferragina, and Marco Ponza. "Wiser: A semantic approach for expert finding in academia based on entity linking". In: *Information Systems* 82 (2019), pp. 1–16. URL: https://arxiv.org/pdf/1805.03947.pdf.

[33] Nick Craswell. "Mean Reciprocal Rank." In: *Encyclopedia of database systems* 1703 (2009).

[34] Marco Cremaschi et al. "A fully automated approach to a complete Semantic Table Interpretation". In: *Future Generation Computer Systems* (2020).

[35] Richard Cyganiak. *Tarql (sparql for tables): Turn csv into rdf using sparql syntax*. Tech. rep. Technical report, Jan. 2015. http://tarql. github. io, 2015.

[36] Jeffrey Dalton, Laura Dietz, and James Allan. "Entity query feature expansion using knowledge base links". In: *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. ACM. 2014, pp. 365–374.

[37] Souripriya Das, Seema Sundara, and Richard Cyganiak. *R2RML: RDB to RDF Mapping Language. W3C Recommendation (2012)*. 2016.

[38] Anupam Datta, Shayak Sen, and Yair Zick. "Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems". In: *2016 IEEE symposium on security and privacy (SP)*. IEEE. 2016, pp. 598–617.

[39] Tim Dettmers et al. "Convolutional 2d knowledge graph embeddings". In: *arXiv preprint arXiv:1707.01476* (2017).

[40] Anastasia Dimou et al. "RML: A Generic Language for Integrated RDF Mappings of Heterogeneous Data." In: *Ldow* 1184 (2014).

[41] Li Ding et al. "TWC data-gov corpus: incrementally generating linked government data from data. gov". In: *Proceedings of the 19th international conference on World wide web*. ACM. 2010, pp. 1383–1386.

[42] Isabella Distinto, Mathieu d'Aquin, and Enrico Motta. "LOTED2: An Ontology of European Public Procurement Notices". In: *Semantic Web* 7.3 (2016), pp. 267–293.

[43] AnHai Doan, Alon Halevy, and Zachary Ives. *Principles of data integration*. Elsevier, 2012.

[44] Li Dong et al. "Question answering over freebase with multi-column convolutional neural networks". In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Vol. 1. 2015, pp. 260–269.

[45] Derek Doran, Sarah Schulz, and Tarek R Besold. "What does explainable AI really mean? A new conceptualization of perspectives". In: *arXiv preprint arXiv:1710.00794* (2017).

[46] Bob DuCharme. *Learning SPARQL: querying and updating with SPARQL 1.1*. " O'Reilly Media, Inc.", 2013.

[47] Michel Dumontier et al. "Bio2RDF release 3: a larger connected network of linked data for the life sciences". In: *Proceedings of the 2014 International Conference on Posters & Demonstrations Track*. Vol. 1272. 2014, pp. 401–404.

[48] Vijay Prakash Dwivedi et al. "Benchmarking graph neural networks". In: *arXiv preprint arXiv:2003.00982* (2020).

[49] Vasilis Efthymiou et al. "Matching web tables with knowledge base entities: from entity lookups to entity embeddings". In: *International Semantic Web Conference*. Springer. 2017, pp. 260–277.

[50] Lisa Ehrlinger and Wolfram Wöß. "Towards a Definition of Knowledge Graphs." In: *SEMANTiCS (Posters, Demos, SuCCESS)* 48 (2016).

[51] Orri Erling and Ivan Mikhailov. "RDF Support in the Virtuoso DBMS". In: *Networked Knowledge-Networked Media*. Springer, 2009, pp. 7–24.

[52] Said Fathalla, Sören Auer, and Christoph Lange. "Towards the semantic formalization of science". In: *Proceedings of the 35th Annual ACM Symposium on Applied Computing*. 2020, pp. 2057–2059.

[53] Dieter Fensel et al. *Knowledge Graphs*. Springer, 2020.

[54] Roy Fielding et al. *Hypertext transfer protocol–HTTP/1.1*. 1999.

[55] Sorelle A Friedler et al. "Assessing the local interpretability of machine learning models". In: *arXiv preprint arXiv:1902.03501* (2019).

[56] Giuseppe Futia and Antonio Vetrò. "On the Integration of Knowledge Graphs into Deep Learning Models for a More Comprehensible AI—Three Challenges for Future Research". In: *Information* 11.2 (2020), p. 122.

[57] Giuseppe Futia, Antonio Vetrò, and Juan Carlos De Martin. "SeMi: A SEmantic Modeling machIne to build Knowledge Graphs with graph neural networks". In: *SoftwareX* 12 (2020), p. 100516.

[58] Giuseppe Futia et al. "Removing barriers to transparency: A case study on the use of semantic technologies to tackle procurement data inconsistency". In: *European Semantic Web Conference*. Springer. 2017, pp. 623–637.

[59] Giuseppe Futia et al. "Training Neural Language Models with SPARQL queries for Semi-Automatic Semantic Mapping". In: *Procedia Computer Science* 137 (2018), pp. 187–198.

[60] Kata Gábor et al. "Semeval-2018 Task 7: Semantic relation extraction and classification in scientific papers". In: *Proceedings of The 12th International Workshop on Semantic Evaluation*. 2018, pp. 679–688.

[61] Aldo Gangemi et al. "Semantic web machine reading with FRED". In: *Semantic Web* 8.6 (2017), pp. 873–893.

[62] Yuxia Geng et al. "Human-centric transfer learning explanation via knowledge graph". In: *arXiv preprint arXiv:1901.08547* (2019).

[63] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[64] Clinton Gormley and Zachary Tong. *Elasticsearch: the definitive guide: a distributed real-time search and analytics engine*. " O'Reilly Media, Inc.", 2015.

[65] Mike Graves, Adam Constabaris, and Dan Brickley. "FOAF: Connecting People on the Semantic Web". In: *Cataloging & Classification Quarterly* 43.3-4 (2007), pp. 191–202. DOI: 10.1300/J104v43n03_10. eprint: https://doi.org/10.1300/J104v43n03_10. URL: https://doi.org/10.1300/J104v43n03_10.

[66] Thomas R Gruber. "Toward principles for the design of ontologies used for knowledge sharing?" In: *International journal of human-computer studies* 43.5-6 (1995), pp. 907–928.

[67] Ramanathan V Guha, Dan Brickley, and Steve Macbeth. "Schema.org: evolution of structured data on the web". In: *Communications of the ACM* 59.2 (2016), pp. 44–51.

[68] Patrick Hall and Navdeep Gill. *An Introduction to Machine Learning Interpretability-Dataiku Version*. O'Reilly Media, Incorporated, 2018.

[69] Steve Harris and Andy Seaborne. "sparql 1.1 query language. Recommendation". In: *World Wide Web Consortium, March* (2013), pp. 89–99.

[70] Ali Hasnain et al. "Linked biomedical dataspace: lessons learned integrating data for drug discovery". In: *International Semantic Web Conference*. Springer. 2014, pp. 114–130.

[71] John Haugeland. *Artificial intelligence: The very idea*. MIT press, 1989.

[72] Mohamed AG Hazber et al. "A survey: Transformation for integrating relational database with semantic Web". In: *Proceedings of the 2019 3rd International Conference on Management Engineering, Software Engineering and Service Sciences*. 2019, pp. 66–73.

[73] Tom Heath and Christian Bizer. "Linked data: Evolving the web into a global data space". In: *Synthesis lectures on the semantic web: theory and technology* 1.1 (2011), pp. 1–136.

[74] Pieter Heyvaert et al. "Ontology-based data access mapping generation using data, schema, query, and mapping knowledge". In: *European Semantic Web Conference*. Springer. 2017, pp. 205–215.

[75] Geoffrey E Hinton, James L McClelland, David E Rumelhart, et al. *Distributed representations*. Carnegie-Mellon University Pittsburgh, PA, 1984.

[76] Ben Hixon, Peter Clark, and Hannaneh Hajishirzi. "Learning knowledge graphs for question answering through conversational dialog". In: *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2015, pp. 851–861.

[77] Robert Hoehndorf, Núria Queralt-Rosinach, et al. "Data science and symbolic AI: Synergies, challenges and opportunities". In: *Data Science* 1.1-2 (2017), pp. 27–38.

[78] Konrad Höffner, Michael Martin, and Jens Lehmann. "LinkedSpending: openspending becomes linked open data". In: *Semantic Web* 7.1 (2016), pp. 95–104.

[79] Aidan Hogan et al. "Knowledge graphs". In: *arXiv preprint arXiv:2003.02320* (2020).

[80] Bryan Hooi et al. "Fraudar: Bounding graph fraud in the face of camouflage". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. 2016, pp. 895–904.

[81] Frank K Hwang and Dana S Richards. "Steiner tree problems". In: *Networks* 22.1 (1992), pp. 55–89.

[82] Marijn Janssen and Jeroen van den Hoven. "Big and Open Linked Data (BOLD) in government: A challenge to transparency and privacy?" In: *Government Information Quarterly* 32.4 (2015), pp. 363–368. ISSN: 0740624X. DOI: 10.1016/j.giq.2015.11.007.

[83] Guoliang Ji et al. "Knowledge graph embedding via dynamic mapping matrix". In: *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers)*. 2015, pp. 687–696.

[84] Ernesto Jiménez-Ruiz et al. "BootOX: Practical mapping of RDBs to OWL 2". In: (2015), pp. 113–132.

[85] Ulf Johansson, Rikard König, and Lars Niklasson. "The Truth is In There-Rule Extraction from Opaque Models Using Genetic Programming." In: *FLAIRS Conference*. Miami Beach, FL. 2004, pp. 658–663.

[86] Seyed Mehran Kazemi and David Poole. "Simple embedding for link prediction in knowledge graphs". In: *Advances in neural information processing systems*. 2018, pp. 4284–4295.

[87] Been Kim, Cynthia Rudin, and Julie A Shah. "The bayesian case model: A generative approach for case-based reasoning and prototype classification". In: *Advances in Neural Information Processing Systems*. 2014, pp. 1952–1960.

[88] Thomas N Kipf and Max Welling. "Semi-supervised classification with graph convolutional networks". In: *arXiv preprint arXiv:1609.02907* (2016).

[89] Craig A Knoblock et al. "Semi-automatically mapping structured sources into the semantic web". In: *Extended Semantic Web Conference*. Springer. 2012, pp. 375–390.

[90] Petr Knoth and Zdenek Zdrahal. "CORE: three access levels to underpin open access". In: *D-Lib Magazine* 18.11/12 (2012), pp. 1–13.

[91] L Kou, George Markowsky, and Leonard Berman. "A fast algorithm for Steiner trees". In: *Acta informatica* 15.2 (1981), pp. 141–145.

[92] Benno Kruit, Peter Boncz, and Jacopo Urbani. "Extracting Novel Facts from Tables for Knowledge Graph Completion". In: *International Semantic Web Conference*. Springer. 2019, pp. 364–381.

[93] Ora Lassila and Ralph R Swick. *Resource description framework (RDF) model and syntax specification*. Tech. rep. W3C recommendation, 1999. URL: http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/.

[94] Ora Lassila, Ralph R Swick, et al. "Resource description framework (RDF) model and syntax specification". In: (1998).

[95] Freddy Lecue. "On the role of knowledge graphs in explainable AI". In: *Semantic Web Journal (Forthcoming). http://www. semantic-web-journal. net/content/role-knowledge-graphs-explainable-ai* (2019).

[96] Jens Lehmann et al. "DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia". In: *Semantic Web* 6.2 (2015), pp. 167–195.

[97] Bruno Lepri et al. "Fair, transparent, and accountable algorithmic decision-making processes". In: *Philosophy & Technology* 31.4 (2018), pp. 611–627.

[98] Lizi Liao et al. "Interpretable multimodal retrieval for fashion products". In: *2018 ACM Multimedia Conference on Multimedia Conference*. ACM. 2018, pp. 1571–1579.

[99] Yankai Lin et al. "Learning Entity and Relation Embeddings for Knowledge Graph Completion, 2015". In: *Google Scholar Google Scholar Digital Library Digital Library* ().

[100] Zachary C Lipton. "The mythos of model interpretability". In: *arXiv preprint arXiv:1606.03490* (2016).

[101] Yi Luan et al. "Multi-Task Identification of Entities, Relations, and Coreference for Scientific Knowledge Graph Construction". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 2018, pp. 3219–3232.

[102] Kaleem Razzaq Malik and Tauqir Ahmad. "Technique for transformation of data from RDB to XML then to RDF". In: *Web Semantics for Textual and Visual Information Retrieval*. IGI Global, 2017, pp. 70–91.

[103] Michael Martin et al. "Increasing the Financial Transparency of European Commission Project Funding". In: *Semantic Web Journal* Special Call for Linked Dataset descriptions.2 (2013), pp. 157–164.

[104] Deborah L McGuinness, Frank Van Harmelen, et al. "OWL web ontology language overview". In: *W3C recommendation* 10.10 (2004), p. 2004.

[105] Luciano Frontino de Medeiros, Freddy Priyatna, and Oscar Corcho. "MIRROR: Automatic R2RML mapping generation from relational databases". In: *International Conference on Web Engineering*. Springer. 2015, pp. 326–343.

[106] Marios Meimaris Michail Vafolopoulos and Giannis Xidias et al. "Publicspending. gr: interconnecting and visualizing Greek public expenditure following Linked Open Data directives". In: *Using Open Data: policy modeling, citizen empowerment, data journalism. W3C, The European Commission*. 2012.

[107] Rada Mihalcea, Courtney Corley, Carlo Strapparava, et al. "Corpus-based and knowledge-based measures of text semantic similarity". In: *Aaai*. Vol. 6. 2006. 2006, pp. 775–780.

[108] Tomas Mikolov et al. "Efficient estimation of word representations in vector space". In: *arXiv preprint arXiv:1301.3781* (2013).

[109] George A Miller. "WordNet: a lexical database for English". In: *Communications of the ACM* 38.11 (1995), pp. 39–41.

[110] Tim Miller, Piers Howe, and Liz Sonenberg. "Explainable AI: Beware of inmates running the asylum or: How I learnt to stop worrying and love the social and behavioural sciences". In: *arXiv preprint arXiv:1712.00547* (2017).

[111] Brent Mittelstadt, Chris Russell, and Sandra Wachter. "Explaining explanations in AI". In: *Proceedings of the conference on fairness, accountability, and transparency*. ACM. 2019, pp. 279–288.

[112] Varish Mulwad, Tim Finin, and Anupam Joshi. "Semantic message passing for generating linked data from tables". In: *International Semantic Web Conference*. Springer. 2013, pp. 363–378.

[113] Mélanie Munch et al. "Interactive Causal Discovery in Knowledge Graphs". In: 2019.

[114] Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. "Holographic embeddings of knowledge graphs". In: *arXiv preprint arXiv:1510.04935* (2015).

[115] Maximilian Nickel and Volker Tresp. "Tensor factorization for multi-relational learning". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2013, pp. 617–621.

[116] Natasha Noy et al. "Industry-scale knowledge graphs: lessons and challenges". In: *Queue* 17.2 (2019), pp. 48–75.

[117] Travis E Oliphant. *A guide to NumPy*. Vol. 1. Trelgol Publishing USA, 2006.

[118] Yasser MK Omar, Ahmed Abd el Moneim, and Kamal Mohamed. "Building a Framework for Mapping RDBMS to RDF With Semantic Query Capabilities". In: *2019 60th International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS)*. IEEE. 2019, pp. 1–5.

[119] Heiko Paulheim. "Knowledge graph refinement: A survey of approaches and evaluation methods". In: *Semantic web* 8.3 (2017), pp. 489–508.

[120] Silvio Peroni and David Shotton. "OpenCitations, an infrastructure organization for open scholarship". In: *Quantitative Science Studies* 1.1 (2020), pp. 428–444.

[121] Silvio Peroni and David Shotton. "The SPAR ontologies". In: *International Semantic Web Conference*. Springer. 2018, pp. 119–136.

[122] Christoph Pinkel et al. "RODI: Benchmarking relational-to-ontology mapping generation quality". In: *Semantic Web* Preprint (2016), pp. 1–28.

[123] Christoph Pinkel et al. ": A benchmark for automatic mapping generation in relational-to-ontology data integration". In: *European Semantic Web Conference*. Springer. 2015, pp. 21–37.

[124] Christoph Pinkel et al. "IncMap: pay as you go matching of relational schemata to OWL ontologies." In: *OM*. 2013, pp. 37–48.

[125] André Pomp et al. "Enhancing Knowledge Graphs with Data Representatives". In: (2019).

[126] André Pomp et al. "Semantic Concept Recommendation for Continuously Evolving Knowledge Graphs". In: *International Conference on Enterprise Information Systems*. Springer. 2019, pp. 361–385.

[127] Irfan Rafique et al. "Information quality evaluation framework: Extending ISO 25012 data quality model". In: *World Academy of Science, Engineering and Technology* 65 (2012), pp. 523–528.

[128] S Krishnamurthy Ramnandan et al. "Assigning semantic labels to data sources". In: *European Semantic Web Conference*. Springer. 2015, pp. 403–417.

[129] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why should i trust you?: Explaining the predictions of any classifier". In: *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. ACM. 2016, pp. 1135–1144.

[130] Petar Ristoski and Heiko Paulheim. "Rdf2vec: Rdf graph embeddings for data mining". In: *International Semantic Web Conference*. Springer. 2016, pp. 498–514.

[131] Petar Ristoski et al. "RDF2Vec: RDF graph embeddings and their applications". In: *Semantic Web* 10.4 (2019), pp. 721–752.

[132] O. R. Rocha et al. "Semantic Annotation and Classification in Practice". In: *IT Professional* 17.2 (Mar. 2015), pp. 33–39. DOI: 10.1109/MITP.2015.29.

[133] Natalia Ruemmele, Yuriy Tyshetskiy, and Alex Collins. "Evaluating approaches for supervised semantic labeling". In: *arXiv preprint arXiv:1801.09788* (2018).

[134] David E Rumelhart. "Parallel distributed processing: Explorations in the microstructure of cognition". In: *Learning internal representations by error propagation* 1 (1986), pp. 318–362.

[135] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. "Learning representations by back-propagating errors". In: *nature* 323.6088 (1986), pp. 533–536.

[136] Peter Sadowski et al. "Deep learning, dark knowledge, and dark matter". In: *NIPS 2014 Workshop on High-energy Physics and Machine Learning*. 2015, pp. 81–87.

[137] Satya S Sahoo et al. "A survey of current approaches for mapping of relational databases to RDF". In: *W3C RDB2RDF Incubator Group Report* 1 (2009), pp. 113–130.

[138] Fatiha Saıs. "Knowledge Graph Refinement: Link Detection, Link Invalidation, Key Discovery and Data Enrichment". PhD thesis. Université Paris Sud, 2019.

[139] Angelo A Salatino et al. "The computer science ontology: A comprehensive automatically-generated taxonomy of research areas". In: *Data Intelligence* 2.3 (2020), pp. 379–416.

[140] Angelo Salatino et al. "The CSO Classifier: Ontology-Driven Detection of Research Topics in Scholarly Articles". In: *TPDL 2019: 23rd International Conference on Theory and Practice of Digital Libraries* (2019). URL: http://oro.open.ac.uk/62026/.

[141] Wojciech Samek et al. *Explainable AI: interpreting, explaining and visualizing deep learning*. Vol. 11700. Springer Nature, 2019.

[142] Md Kamruzzaman Sarker et al. "Explaining trained neural networks with semantic web technologies: First steps". In: *arXiv preprint arXiv:1710.04324* (2017).

[143] Luisa Schiavone et al. "Library data integration: the CoBiS linked open data project and portal". In: *Italian Research Conference on Digital Libraries*. Springer. 2018, pp. 15–22.

[144] Michael Schlichtkrull et al. “Modeling relational data with graph convolutional networks”. In: *European Semantic Web Conference*. Springer. 2018, pp. 593–607.

[145] Arne Seeliger, Matthias Pfaff, and Helmut Krcmar. “Semantic Web Technologies for Explainable Machine Learning Models: A Literature Review”. In: *PROFILES 2019* (2019), p. 30.

[146] Ramprasaath R Selvaraju et al. “Choose Your Neuron: Incorporating Domain Knowledge through Neuron-Importance”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 526–541.

[147] Richard Socher et al. “Reasoning with neural tensor networks for knowledge base completion”. In: *Advances in neural information processing systems*. 2013, pp. 926–934.

[148] Sirawit Sopchoke, Ken-ichi Fukui, and Masayuki Numao. “Explainable cross-domain recommendations through relational learning”. In: *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018.

[149] Dimitrios-Emmanuel Spanos, Periklis Stavrou, and Nikolas Mitrou. “Bringing relational databases into the semantic web: A survey”. In: *Semantic Web* 3.2 (2012), pp. 169–209.

[150] Manu Sporny et al. “JSON-LD 1.1–a JSON-based serialization for Linked Data”. PhD thesis. W3C, 2019.

[151] Ondrej Šváb et al. “Ontofarm: Towards an experimental collection of parallel ontologies”. In: *Poster Track of ISWC* 2005 (2005).

[152] Vojtěch Svátek et al. “Linked Open Data for Public Procurement”. In: *Linked Open Data – Creating Knowledge Out of Interlinked Data: Results of the LOD2 Project*. Cham: Springer International Publishing, 2014, pp. 196–213. ISBN: 978-3-319-09846-3. DOI: 10.1007/978-3-319-09846-3_10.

[153] Zareen Syed et al. “Exploiting a web of semantic data for interpreting tables”. In: *Proceedings of the Second Web Science Conference*. 2010.

[154] Zareen Syed, Tim Finin, Anupam Joshi, et al. “Wikitology: Wikipedia as an ontology”. In: *Proceedings of the Grace Hopper Celebration of Women in Computing Conference*. 2008.

[155] Mohsen Taheriyan et al. “A graph-based approach to learn semantic descriptions of data sources”. In: *International Semantic Web Conference*. Springer. 2013, pp. 607–623.

[156] Mohsen Taheriyan et al. “Learning the semantics of structured data sources”. In: *Web Semantics: Science, Services and Agents on the World Wide Web* 37 (2016), pp. 152–169.

[157] Mohsen Taheriyan et al. “Leveraging linked data to discover semantic relations within data sources”. In: *International Semantic Web Conference*. Springer. 2016, pp. 549–565.

[158] Kunihiro Takeoka et al. “Meimei: An efficient probabilistic approach for semantically annotating tables”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. 2019, pp. 281–288.

[159] Paolo Tamagnini et al. “Interpreting black-box classifiers using instance-level visual explanations”. In: *Proceedings of the 2nd Workshop on Human-In-the-Loop Data Analytics*. ACM. 2017, p. 6.

[160] Jeremy Tandy, Ivan Herman, and Gregg Kellogg. “Generating RDF from tabular data on the web”. In: (2015).

[161] Henry S Thompson et al. “W3C XML schema definition language (XSD) 1.1 part 1: Structures”. In: *The World Wide Web Consortium (W3C), W3C Working Draft Dec* 3 (2009).

[162] Kristina Toutanova and Danqi Chen. “Observed versus latent features for knowledge base and text inference”. In: *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*. 2015, pp. 57–66.

[163] Théo Trouillon et al. “Complex embeddings for simple link prediction”. In: International Conference on Machine Learning (ICML). 2016.

[164] Francesco Valle et al. “LOTED: Exploiting Linked Data in Analyzing European Procurement Notices”. In: *Proceedings of the 1st Workshop on Knowledge Injection into and Extraction from Linked Data - KIELD 2010*. 2010.

[165] Petros Venetis et al. “Recovering semantics of tables on the web”. In: (2011).

[166] Ruben Verborgh and Max De Wilde. *Using OpenRefine*. Packt Publishing Ltd, 2013.

[167] Antonio Vetrò, Juan Carlos De Martin, and Giovanni Garifo. “Deep Learning on Academic Knowledge Graphs”. In: ().

[168] Binh Vu, Craig Knoblock, and Jay Pujara. “Learning Semantic Models of Data Sources Using Probabilistic Graphical Models”. In: *The World Wide Web Conference*. ACM. 2019, pp. 1944–1953.

[169] Hongwei Wang et al. “Multi-task feature learning for knowledge graph enhanced recommendation”. In: *The World Wide Web Conference*. 2019, pp. 2000–2010.

[170] Hongwei Wang et al. “Ripplenet: Propagating user preferences on the knowledge graph for recommender systems”. In: *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 2018, pp. 417–426.

[171] Minjie Wang et al. “Deep graph library, 2018”. In: *URL http://dgl. ai* ().

[172] Peng Wang et al. “Explicit knowledge-based reasoning for visual question answering”. In: *arXiv preprint arXiv:1511.02570* (2015).

[173] Zhen Wang et al. “Knowledge graph embedding by translating on hyperplanes.” In: *Aaai*. Vol. 14. 2014. Citeseer. 2014, pp. 1112–1119.

[174] Stuart Weibel et al. “Dublin core metadata for resource discovery”. In: *Internet Engineering Task Force RFC* 2413.222 (1998), p. 132. URL: http://www.hjp.at/doc/rfc/rfc2413.html.

[175] Xander Wilcke, Peter Bloem, and Victor de Boer. “The Knowledge Graph as the Default Data Model for Machine Learning”. In: ().

[176] Guohui Xiao et al. “Ontology-based data access: A survey”. In: IJCAI Organization, 2018.

[177] Chenyan Xiong and Jamie Callan. “Esdrank: Connecting query and documents through external semi-structured data”. In: *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM. 2015, pp. 951–960.

[178] Bishan Yang et al. “Embedding entities and relations for learning and inference in knowledge bases”. In: *arXiv preprint arXiv:1412.6575* (2014).

[179] Bin Yu et al. “Research and implementation of data fusion method based on RDF”. In: *2018 IEEE 3rd International Conference on Big Data Analysis (ICBDA)*. IEEE. 2018, pp. 87–91.

[180] Li Yujian and Liu Bo. “A normalized Levenshtein distance metric”. In: *IEEE transactions on pattern analysis and machine intelligence* 29.6 (2007), pp. 1091–1095.

[181] Ziqi Zhang. “Effective and efficient semantic table interpretation using tableminer+”. In: *Semantic Web* 8.6 (2017), pp. 921–957.

This Ph.D. thesis has been typeset by means of the TEX-system facilities. The typesetting engine was pdfLATEX. The document class was `toptesi`, by Claudio Beccari, with option `tipotesi=scudo`. This class is available in every up-to-date and complete TEX-system installation.