Tab2KG: Semantic Table Interpretation with Lightweight Semantic Profiles

Simon Gottschalk [a,*] and Elena Demidova [b]
[a] *L3S Research Center, Leibniz Universität Hannover, Hannover, Germany*
E-mail: gottschalk@L3S.de
[b] *Data Science & Intelligent Systems Group (DSIS), University of Bonn, Bonn, Germany*
E-mail: elena.demidova@cs.uni-bonn.de

Abstract. Tabular data plays an essential role in many data analytics and machine learning tasks. Typically, tabular data does not possess any machine-readable semantics. In this context, semantic table interpretation is crucial for making data analytics workflows more robust and explainable. This article proposes Tab2KG – a novel method to automatically infer tabular data semantics and transform such data into a semantic data graph. We introduce original lightweight semantic profiles that enrich a domain ontology's concepts and relations and represent domain and table characteristics. We propose a one-shot learning approach that relies on these profiles to map a tabular dataset containing previously unseen instances to a domain ontology. In contrast to the existing semantic table interpretation approaches, Tab2KG relies on the semantic profiles only and does not require any instance lookup. This property makes Tab2KG particularly suitable in the data analytics context, in which data tables typically contain new instances. Our experimental evaluation on several real-world datasets from different application domains demonstrates that Tab2KG outperforms state-of-the-art semantic table interpretation baselines.

Keywords: Semantic Table Interpretation, Domain Knowledge Graphs, Semantic Profiles, One-shot Learning

1. Introduction

A vast amount of data is currently published in a tabular format [1–3]. Typically, this data does not possess any machine-readable semantics. Semantic table interpretation is an essential step to make this data usable for a wide variety of applications, with data analytics workflows (DAWs) as a prominent example [4]. DAWs include data mining algorithms and sophisticated deep learning architectures and require a large amount of heterogeneous data as an input. Typically, DAWs treat tabular data as character sequences and numbers without inferring any further semantics. This practice can often lead to error-prone analytics processes and results, particularly when data analytics frameworks utilize the data from various heterogeneous sources. Therefore, DAWs can substantially profit from the semantic interpretation of the involved data tables [5, 6].

In this context, semantic table interpretation aims to transform the input data table into a semantic data graph. In this process, table columns are mapped to a domain ontology's classes and properties; table cell values are transformed into literals, forming the data graph – a network of semantic statements, typically encoded in RDF. In the context of DAWs, semantic table interpretation can bring several advantages. First, an abstraction from tabular data to semantic concepts and relations can guide domain experts in the DAW creation [4]. Second, validation options (e.g., type inference) that become available through the semantic table interpretation can increase the robustness of DAWs [7]. Third, semantic descriptions can be employed to facilitate the explainability of the data analytics results [8]. Finally, semantic table interpretation adds structure directly usable for knowledge inference [9].

The existing approaches to semantic table interpretation do not adequately support the interpretation of tabular data for DAWs. At the core of such approaches (e.g., [10–12]) is the instance lookup task, where table

*Corresponding author. E-mail: gottschalk@L3S.de.

1570-0844/0-1900/$35.00 © 0 – IOS Press and the authors. All rights reserved

cells are linked to known instances in a target knowledge graph, with DBpedia [13] being a popular cross-domain target. Subsequent steps such as property mapping are based on the results of this lookup step. However, as shown by Ritze et al. [14], only about 3% of the tables contained in the 3.5 billion HTML pages of the Common Crawl Web Corpus[1] can be matched to DBpedia. In the context of DAW, the input data typically represents new instances (e.g., sensor observations, current road traffic events, ...), and substantial overlap between the tabular data values and entities within existing knowledge graphs cannot be expected.

In this article, we present *Tab2KG* – a novel semantic table interpretation approach. *Tab2KG* aims to transform a data table into a semantic data graph. As a backbone of the data graph, *Tab2KG* relies on an existing domain ontology that defines the concepts and relations in the target domain. To facilitate the transformation, *Tab2KG* introduces original lightweight semantic profiles for domains and data tables. Domain profiles enrich ontology relations and represent domain characteristics. A domain profile associates relations with feature vectors representing data types and statistical characteristics such as value distributions. To transform a data table, *Tab2KG* first creates a data table profile. Then, *Tab2KG* uses the domain and data table profiles to transform the table into a data graph using a novel one-shot learning approach.

To create domain profiles, *Tab2KG* uses a domain ontology and a data sample that can be obtained from an existing domain knowledge graph. Lightweight semantic profiles generated by *Tab2KG* can be utilized as compact domain representations and complement and enrich existing dataset catalogs. Such profiles can be generated automatically from the existing datasets and described using the DCAT[2] and the SEAS[3] vocabularies to enable their reusability. We believe that lightweight semantic profiles presented in this article are an essential contribution that can benefit a wide range of semantic applications beyond semantic table interpretation.

In summary, our contributions presented in this article are as follows:

1. We introduce lightweight semantic domain and table profiles. Domain profiles enrich relations of domain ontologies and serve as a lightweight domain representation. Semantic table profiles summarize data tables facilitating effective semantic table interpretation.
2. We propose the *Tab2KG* approach to transform tabular data into a data graph with one-shot learning based on semantic profiles.
3. We evaluate the proposed method on several real-world datasets. Our evaluation results demonstrate that *Tab2KG* outperforms state-of-the-art semantic table interpretation baselines.
4. We make the scripts for creating lightweight semantic profiles and transforming data tables into data graphs publicly available[4].

The structure of this article is as follows: In Section 2, we introduce a running example used throughout this article. Then, in Section 3, we define the problem of semantic table interpretation, followed by the definition and creation of domain and data table profiles (Section 4). In Section 5, we describe our proposed *Tab2KG* approach and its implementation (Section 6). We present evaluation setup and results in Sections 7 and 8, followed by a discussion of our profile-based approach in Section 9. Then, we discuss related work in Section 10. Finally, we provide a conclusion in Section 11.

2. Running Example from the Weather Observation Domain

Tab2KG is a domain-independent approach that generalizes to previously unseen domains and data tables. There are no constraints on the data nature (e.g., sensor data, numbers, strings, ...), and we demonstrate in our evaluation how *Tab2KG* performs on different domains, including soccer and advertisements data. As a running example, we use the weather observation domain and a data table that provides observations of sensors measuring air conditions.

Consider the table in Fig. 1 that contains weather observation sensor data, separated by a tab character (↠). The table does not include column titles. As a human, we can observe that the first column refers to the air condition (*cloudy*, *clear*, *rain*). The second and third column may represent a time interval of the measurement (e.g., *16:30* and *17:00*). The fourth column containing the values *S2*, *S3*, and *S1* is hard to interpret without background knowledge.

[1] http://commoncrawl.org/
[2] https://www.w3.org/TR/vocab-dcat-2/
[3] https://ci.mines-stetienne.fr/seas/index.html
[4] https://github.com/sgottsch/Tab2KG

```
cloudy  ↠  16:30  ↠  17:00  ↠  S2
clear   ↠  10:00  ↠  10:30  ↠  S3
cloudy  ↠  17:00  ↠  17:30  ↠  S3
rain    ↠  16:30  ↠  17:00  ↠  S1
cloudy  ↠  17:30  ↠  18:00  ↠  S2
clear   ↠  08:30  ↠  09:00  ↠  S1
clear   ↠  09:00  ↠  09:30  ↠  S1
```

Fig. 1. Example of a data table as a tab-separated file without column titles.

To facilitate semantic table interpretation, *Tab2KG* relies on background knowledge regarding the target domain. Such background consists of two parts: a domain ontology and a domain profile.

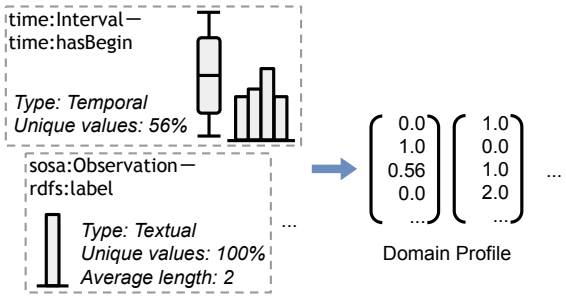

Fig. 3. An example profile of the weather observation domain. The domain profile is represented as a set of feature vectors, each containing statistical features, such as value distributions. The domain profile can also be used for visualization.

– The **domain ontology** models the specific domain of interest. In our running example, we use the Semantic Sensor Network Ontology[5] illustrated in Fig. 2. Amongst others, this ontology provides classes to model sensors, their observations, and corresponding time intervals.
– The **semantic domain profile** is a lightweight representation of typical value distributions for the ontology relations. In this example, such distributions can be obtained from prior weather observations. Fig. 3 gives an exemplary illustration of such a domain profile in the weather observation domain. Here, we illustrate statistical features of two observation properties (the beginning of the observation and the sensor label) using box plots and histograms. Such features can be represented as numerical feature vectors and included in a semantic domain description using DCAT and SEAS vocabularies.

[5] `sosa`: https://www.w3.org/TR/vocab-ssn/

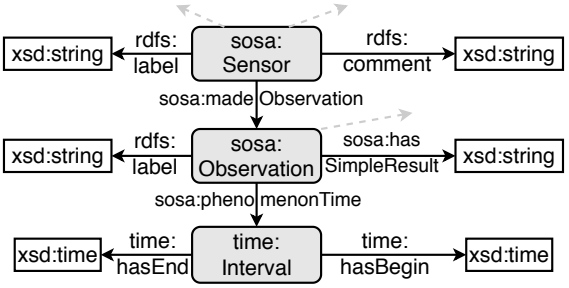

Fig. 2. Excerpt of the Semantic Sensor Network Ontology.

Tab2KG enables the semantic table interpretation through profile matching, which maps table columns to the ontology relations. Given the mapping, *Tab2KG* transforms the table into the data graph shown in Fig. 4. As we can observe, the first three columns are mapped to the observations and their time intervals. The fourth column is mapped to the sensor labels.

The transformation process is challenging and potentially error-prone. For example, Fig. 5 illustrates a wrong transformation result, with an incorrect column mapping and an erroneous graph structure. In this case, the sensor label “S3" was erroneously interpreted as an observation label. In addition, the beginning and end times are swapped. *Tab2KG* utilizes semantic profiles to avoid such interpretation errors.

3. Problem Statement

In this section, we first formally define relevant concepts. Then, we present the task of semantic table interpretation.

The entities and relations in the domain of interest can be represented in a domain knowledge graph.

Definition 1. *A **domain knowledge graph** is a graph $G = (N, R)$, whose nodes N represent entities and literals, and whose edges R represent relations between these nodes in the specific domain.*

A domain knowledge graph consists of two sub graphs: a domain ontology and a data graph.

Definition 2. *A **domain ontology** $G_O = (N_O, R_O)$, $N_O \subset N, R_O \subset R$, where $N_O = C \cup D \cup P$ includes a set*

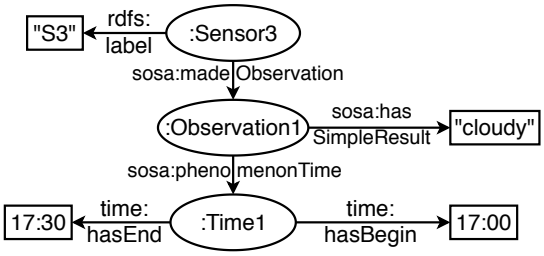

Fig. 4. Correct mapping of the third line in Fig. 1 to the ontology in Fig. 2. For brevity, we omit `rdf:type` relations.

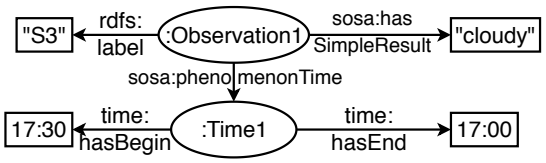

Fig. 5. Incorrect mapping of the third line in Fig. 1 to the ontology in Fig. 2. For brevity, we omit `rdf:type` relations.

of classes C, a set of data types D, and a set of properties $P = P_d \cup P_o$, where P_d are data type properties, and P_o are object properties. Data type properties relate entities to literals. Object properties relate entities to each other.

The relations represented by $R_O = R_{OC} \cup R_{OD}$ include class relations R_{OC} and data type relations R_{OD}:

- *R_{OC} is the set of class relations: $R_{OC} \subseteq C \times P_o \times C$.*
- *R_{OD} is the set of data type relations: $R_{OD} \subseteq C \times P_d \times D$.*

For example, in the excerpt of Semantic Sensor Network Ontology illustrated in Fig. 2, (`sosa:Sensor sosa:madeObservation sosa:Observation`) is a class relation and (`sosa:Sensor rdfs:label xsd:string`) is a data type relation.

Definition 3. *A **data graph** is a graph $G_D = (N_D, R_D)$, $N_D \subset N, R_D \subset R$. The nodes $N_D = C \cup D \cup E \cup L$ include classes C, data types D, entities E and literals L. Each literal $l \in L$ is assigned a data type $dt(l) \in D$. Within R_D, we distinguish between entity relations ($E \times P_o \times E$) and literal relations ($E \times P_d \times L$).*

A data table is defined as follows:

Definition 4. *A **data table** T is a $M \times N$ matrix consisting of M rows and N columns. A cell $T_{m,n}$, $m \in \{1,\ldots,M\}, n \in \{1,\ldots,N\}$, represents a data value. A row r_m is a tuple that represents a set of semantically related entities. A column c_n represent a specific characteristic of the entities in a row.*

For example, the data table illustrated in Fig. 1 contains $M = 7$ rows and $N = 4$ columns, where the columns represent the weather conditions, time intervals, and sensor labels, and each row contains three semantically related entities: an observation, a time interval, and a sensor. The column values can belong to different data types, including text, numeric, Boolean, temporal and spatial.

Semantic table interpretation is the task of transforming a data table into a data graph.

Definition 5. ***Semantic table interpretation**: Given a data table T and a domain knowledge graph G, create a data graph $G_D^T = (N_D^T, R_D^T)$ with nodes N_D^T and relations R_D^T. Its literal values $L^T \subseteq N_D^T$ are connected to the entities $E^T \subset N_D^T$ and represent the values in the literal columns of T. The entities E^T are connected via entity relations in R_D^T.*

4. Semantic Profiles

Semantic table interpretation in *Tab2KG* does not require any instance lookup in a domain knowledge graph. Instead, *Tab2KG* uses a sample domain knowledge graph to create a lightweight semantic domain profile. This domain profile, together with a domain ontology, build reusable **domain background knowledge** that is later on used to interpret the data tables semantically. Note that the entities and literals in the sample domain knowledge graph do not need to overlap with the data tables' instances to be interpreted.

Tab2KG involves the creation of two types of profiles: *domain profiles* and *data table profiles*, both represented as feature vectors and described in a semantic data catalog to facilitate their reusability. Such profiles are inspired by the dataset profiles described in [15], where statistical features are defined as an important element of a dataset profiles taxonomy. In *Tab2KG*, the primary purpose of the domain and data table profiles is to enable effective and efficient access to the domain and table statistics for semantic table interpretation.

We present domain profiles in Section 4.1 and data table profiles in Section 4.2. We discuss profile features in Section 4.3. Then, in Section 4.4, we describe

how we represent domain and data table profiles in a semantic, machine-readable way. Finally, in Section 4.5 we provide an example of a data catalog that includes semantic profiles.

4.1. Domain Profiles

For creating a domain profile, we make use of a domain knowledge graph G that contains representative values for the data type relations in the target domain. A **domain profile** is a set of data type relation profiles derived from G, where a data type relation profile is a set of statistical characteristics (features) of the literals covered by this data type relation in G's data graph G_D.

Definition 6. *The **data type relation profile** $\pi(r_D) \in \mathbb{R}^f$ of the data type relation $r_D \in R_{OD}$ is a vector that includes f features of the literal relations covered in the domain knowledge graph G.*

In brief, the profile of a data type relation r_D is a feature vector containing a set of statistics, computed using all literals corresponding to r_D.

To create a profile for the data type relation $r_D = (c, p_d, d)$, we utilize all literals $l \in G_D$, such that: $(e, p_d, l) \in R_D$, $(e, rdf:type, c) \in R_D$, and $dt(l) = d$.

In our running example, in Fig. 4, the data type relation (`sosa:Sensor rdfs:label xsd:string`) in the domain ontology corresponds to the literal relation (`:Sensor3 rdfs:label` "S3") in the data graph. Therefore, we use "S3" as one of the literals to create the data type relation profile.

4.2. Data Table Profiles

To facilitate semantic interpretation of a data table, we create a data table profile.

A **data table profile** is a set of column profiles, each representing a specific data table column. More formally, the profile of a data table T consists of a column profile $\pi(c_n), n \in \{1, \ldots, N\}$ for each table column $c_n \in T$ as defined in Definition 4.

A column profile is defined as follows:

Definition 7. *A **column profile** $\pi(c_n) \in \mathbb{R}^f$ of a column c_n is a vector of f statistical characteristics (features) of the values contained in that column.*

We create column profiles using literal values contained in the table columns.

Column profiles and data type relation profiles are created analogously and contain the same features, presented in Section 4.3.

4.3. Profile Features

Motivated by the RDF profile characteristics defined by Ellefi et al. [15], we include data types, as well as completeness and statistical features described in the following into the profiles in *Tab2KG*. The selection is motivated by the expected feature effectiveness for semantic table interpretation, i.e., matching the domain and data table profiles. We demonstrate in our evaluation that these features can facilitate an effective matching in several application domains. This feature set can be further extended to include relevant characteristics in specific domains.

- **Data type:** We represent data types as binary profile features. We include fine-granular data types to facilitate the precise matching of domain and data table profiles. The following data type taxonomy includes the most common cases observed in our evaluation domains.
  - **Text:** Categorical, URL, Email, Other
  - **Numeric:** Integer, Decimal / Sequential, Categorical, Other[6]
  - **Boolean**
  - **Temporal:** Date, Time, Date Time
  - **Spatial:** Point, Linestring, Polygon

  A data type relation or column can be assigned multiple (fine-granular) data types (e.g., integer and categorical). We provide technical details regarding the identification of fine-granular data types later in Section 6.
- **Completeness:** We include the number of non-null values as a completeness indicator.
- **Basic statistics:** We include the number of values, the number of distinct values, as well as the average length and the average number of digits in the literals.
- **Histograms:** Histograms are an effective means for RDF data summarization [17]. We create a histogram for a given number of buckets as part of the data type relation profile or column profile. As features, we add the number of literals in each bucket, in the increasing order of bucket ranges. For histogram creation, we remove the outliers detected using the interquartile range (1.5 IQR) rule.

[6] following the taxonomy defined in [16]

– **Quantiles:** We add quartiles and deciles to the profile (including minima and maxima). In addition, we add the number of outliers detected using the 1.5 IQR rule.

To derive numerical features, we transform literals into numbers. The features of textual data type relations are computed based on the textual values' lengths. Temporal values are transformed into timestamps. For spatial values, we consider the line string length or the polygon area, respectively.

4.4. Semantic Profile Representation

Domain and data table profiles can be represented as **semantic profiles** in RDF, as an extension of the Data Catalog Vocabulary (DCAT)[7] and the SEAS Statistics ontology[8].

Within the DCAT vocabulary, a data catalog (`dcat:DataCatalog`) consists of datasets (`dcat:Dataset`), where a dataset is a collection of data, published or curated by a single agent. In the context of *Tab2KG*, both the domain knowledge graph and the data tables can be represented using `dcat:Dataset`. We extend the descriptions of datasets in a *Tab2KG* data catalog to include semantic profiles. For example, we introduce an `Attribute` class representing the data table columns and data type relations. We make the definitions of this vocabulary available online[9].

Fig. 6 provides an overview of the classes involved in representing semantic profiles. A `dcat:Dataset` in a *Tab2KG* data catalog includes several attributes. In the case of a data table profile, these attributes the columns. In the case of a domain profile, these attributes are the data type relations. These attributes are assigned the profile features, as presented in Section 4.3:

1. Data types: Data type assignments follow the previously mentioned taxonomy.
2. Numeric features: The numeric profile features are represented through subclasses of `seas:Evaluation`. In the case of quartiles, deciles, and histograms, the values come with a rank. For example, we can denote the second decile using `seas:rank` 2.

In the case of domain profiles, the existing mapping to the domain ontology can be modeled by connecting attributes to their corresponding classes, and data type properties [4].

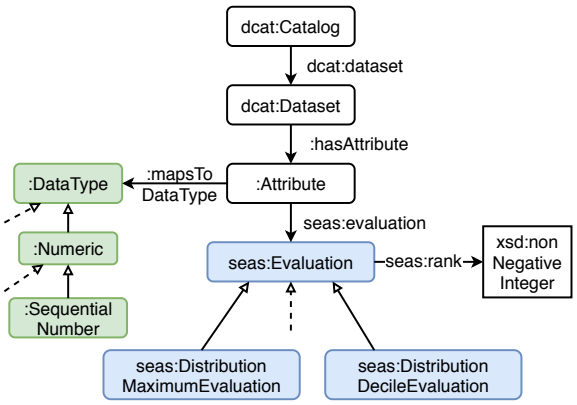

Fig. 6. Classes and properties used for describing a semantic profile. ⇾ marks `owl:subClassOf` relations. Dashed arrows indicate the existence of further classes which are not included in this excerpt. The two feature types (data types and numeric features) are grouped by different colors.

Note that such semantic profiles do not only enable semantic table interpretation but can also be used to provide lightweight dataset visualizations, e.g., through box plots (quartiles) or bar charts (histograms).

4.5. Running Example: Weather Data Catalog

An excerpt of an example data catalog for our running example from the weather observation domain introduced in Section 2 is shown in Fig. 7. The data catalog identified as `WeatherCatalog` includes two data tables (`RainData` and `AirData`). Here, the `AirData` data table has two columns, one of them with a column profile feature denoting the maximum value of the observation end time.

With *Tab2KG*, we can directly utilize this catalog for semantic table interpretation. Both example data tables can be interpreted through their data table profiles when a domain profile of the weather observation domain is provided.

5. *Tab2KG* Semantic Table Interpretation Approach

This section describes the process of semantic table interpretation.

[7] https://www.w3.org/TR/vocab-dcat-2/
[8] https://ci.mines-stetienne.fr/seas/StatisticsOntology
[9] https://github.com/sgottsch/Tab2KG

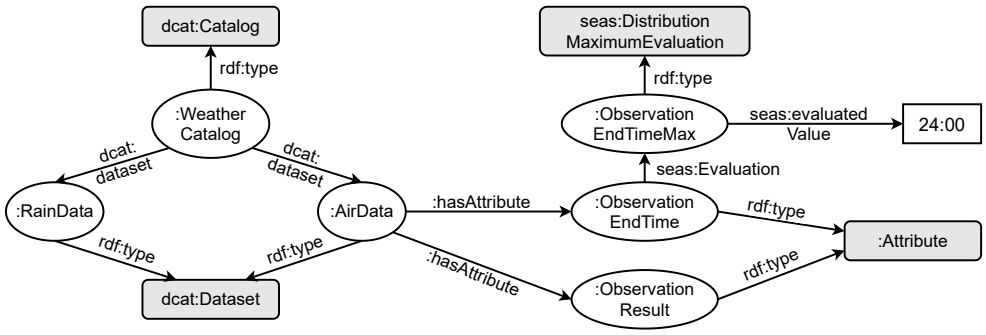

Fig. 7. Running example: Excerpt of a weather data catalog containing two data tables and an exemplified column profile feature denoting the maximum end time value (`ObservationEndTimeMax`).

5.1. Approach Overview

Fig. 8 provides an overview of our proposed *Tab2KG* approach to semantic table interpretation, where a data graph G_D^T is created from a data table T. To facilitate the interpretation, *Tab2KG* utilizes domain background knowledge that includes a domain ontology G_O and a domain profile. The domain profile is generated in a pre-processing step from a domain knowledge graph G.

In brief, the *Tab2KG* pipeline consists of the following steps:

1. **Domain Profile Creation**: In a pre-processing step, we create a domain profile from a domain knowledge graph G.
2. **Data Table Profile Creation**: We create a profile of the input data table T.
3. **Column Mapping**: We generate candidate mappings between the columns of T and the data type relations in G_O using the domain profile, the data table profile, and a one-shot learning mapping function.
4. **Data Graph Creation**: We use the candidate column mappings and the domain ontology G_O to create a data graph G_D^T representing T's content.

In the following, we describe these steps in more detail.

5.2. Domain Profile Creation

The semantic table interpretation in *Tab2KG* requires the availability of a domain profile. This profile can be inferred from a domain knowledge graph G as described in Section 4.1. The domain profile is created by computing the feature values given the literal relations in the domain knowledge graph. This profile can be used as a lightweight domain representation. The domain profile can be created in a pre-processing step and become available as part of a data catalog as described in Section 4.4.

Note that the domain profile does not contain any entities or literals from the domain knowledge graph G. The domain knowledge graph is not directly used for the semantic table interpretation.

5.3. Data Table Profile Creation

From the input data table T, we create a data table profile by computing the profile features based on the column values as described in Section 4.2.

5.4. Column Mapping

With the help of the domain profile and the data table profile, we create *column mappings*.

Definition 8. *A **column mapping** is a mapping from a column c_n in a data table T to a data type relation $r_D \in R_D$ in the domain ontology G_O: $c_n \mapsto r_D$.*

For example, we can create a mapping from column c_2 of the data table illustrated in Fig. 1 to the data type relation (`time:Interval - time:hasBegin - xsd:time`) in the ontology illustrated in Fig. 2.

Within the *Tab2KG* pipeline shown in Fig. 8, we use a mapping function that creates a set of candidate column mappings M_{c_n} for each column c_n in the data table T. Given a column profile and a data type relation profile, the mapping function returns a similarity score in the range $[0, 1]$. The mapping is created by detecting all data type relation profiles $\pi(r_D)$ similar to the respective column profile $\pi(c_n)$. In this step, we only consider mappings between columns and data type relations of the same data type (numeric, textual, temporal, spatial or Boolean).

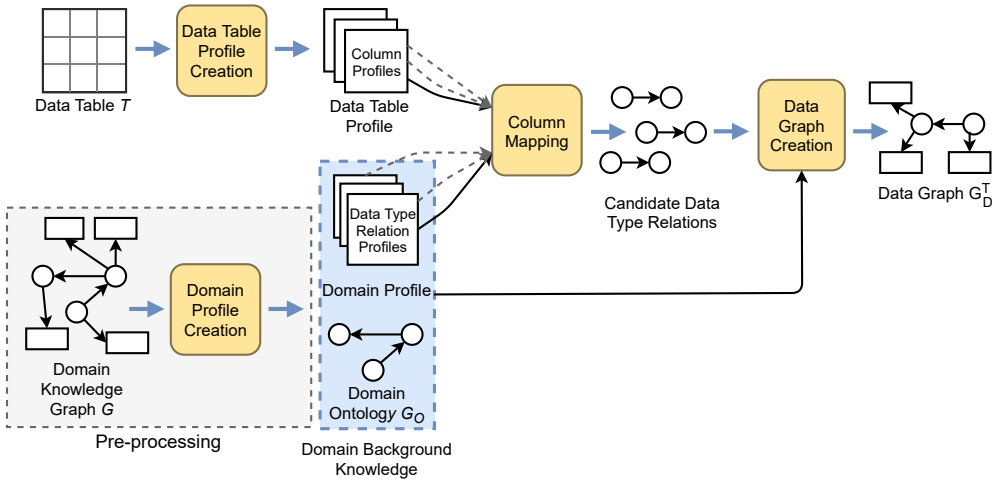

Fig. 8. An overview of semantic table interpretation with *Tab2KG*. Input is a data table T and a domain knowledge graph G. The output is a data graph G_D^T that represents the content of T as a data graph.

The architecture of the column mapping function is shown in Fig. 9. First, it takes two profiles as an input and performs a joint normalization, i.e., the features are normalized in a range between 0 and 1 concerning the sum of the values in both profiles. Then, we follow the idea used for one-shot learning for image classification [18]. Here, the task is not only to classify images into known classes (e.g., many images showing tigers) but also to generalize towards new classes (e.g., new images showing lions). That means, the underlying classifier needs to acquire features which enable the model to successfully generalize. This is done by inducing a metric that represents the domain-independent similarity between two input feature vectors (e.g., between an unknown image and a single image showing a lion).

As we cannot train a classifier on known classes (in contrast to domain-specific approaches such as Sherlock [19] and ColNet [12]), we are in a one-shot learning setting as well: We may learn how to map column profiles to known data type relations. But when facing a new domain, our classifier needs to generalize towards unseen data type relations. In *Tab2KG*, the similarity between a column and a data type relation is predicted based on the experience of the similarity of other profiles learned earlier. The score to measure such similarity is facilitated by a Siamese network that encodes both profiles using the same weights and then predicts the similarity score based on the difference between the two profile encodings. As in [18], we use Rectified Linear Units for the hidden layers and a Sigmoid output layer.

5.5. Data Graph Creation

Given a set of candidate column mappings M_{c_n} with similarity scores for each column c_n in the input data table T, we now assign each table column a data type relation in a greedy manner. First, we take the column mapping with the highest similarity score. Then, we remove all candidate column mappings with the particular column or data type relation. These two steps are repeated until all columns are assigned a data type relation.

From the chosen column mappings set, we create the data graph G_D^T that contains all data type relations resulting from the chosen mapping. G_D^T needs to adhere to the following four conditions:

1. The data graph covers all literal columns of T, and each literal column has exactly one mapping to a data type relation.
2. The set of entities in a table row is connected via entity relations.
3. G_D^T is minimal, i.e., no relation can be removed without invalidating the previous two conditions.
4. Each class relation represented by G_D^T is connected to at least one class that is part of M_{c_n}. This condition ensures semantic closeness of the data table columns and reduces the number of potential paths in the graph.

5.6. Creation of Training Instances for Column Mapping

For the computation of the column mapping function, we utilize a Siamese network trained once in a

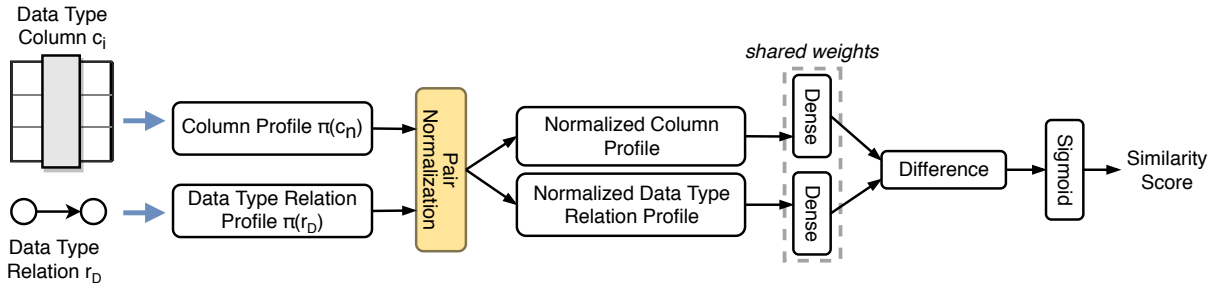

Fig. 9. Architecture of the mapping function to predict the similarity between a column c_n and a data type relation r_D.

pre-processing step. This training process requires the extraction of positive and negative training instances. Following Definition 5 (see also Fig. 8), this step requires a set of (G, T, G_D^T) triples, where G is the domain knowledge graph, T is the data table and G_D^T is the resulting data graph. For each triple (G, T, G_D^T), each pair of data type relations in G and a column in T is a positive training instance. We select the remaining (data type relation, column) pairs from the same knowledge graph G as negative instances.

In the first step, we synthetically create a set of (G, T, G_D^T) triples for the model training, intending to have a large dataset of positive and negative examples derived from existing data tables and knowledge graphs. Such data is difficult to obtain, except for manually created, task-specific research datasets [20], which are not large enough for training deep neural networks and do not provide enough topical and structural diversity. Therefore, we utilize existing knowledge graphs to create training data.

Given a set of knowledge graphs, we create a new dataset of triples (G_1, T, G_2^T). Each input knowledge graph G is disjunctly split into two KGs: G_1 and G_2. G_1 represents the domain knowledge graph, while G_2 is transformed into a data table T. The transformation of G_2 into T is based on a set of domain ontology templates. A template is a directed tree with up to k nodes, where k is a parameter. The nodes and edges of these trees are placeholders for classes and properties. A set of trees is transformed into domain ontologies by replacing these placeholders with the classes and properties of G_2. From the knowledge graph G_2, a data graph G_2^T is extracted and transformed into a data table T to create the triple (G_1, T, G_2^T).

We aim to retrieve a set of heterogeneous data tables that represent the original knowledge graph characteristics. Therefore, the data table creation process incorporates several stochastic decisions in proportion to the knowledge graph statistics:

- Entities and entity relations (and consequently, the literal relations) in G are split at a random ratio between 25% and 75% into G_1 and G_2, whereas their domain ontologies remain the same.
- During the template creation, classes, and properties are assigned randomly to a domain ontology template, proportionally to their occurrence rate in G.
- Data type relations are added in the same manner, under the condition that each leaf node has to be connected with at least one data type relation. After adding the minimal required number of data type relations, we add data type relations as long as any of them are left and if a randomly generated number between 0 and 1 exceeds a predefined threshold δ.

5.7. Running Example: Data Table Creation

For our running example introduced in Section 2, Fig. 10 illustrates the transformation of a domain knowledge graph G and a domain ontology τ into a data table T with two rows and two columns. In this specific minimal example, the data graph G_D^T is identical with the input knowledge graph G, as τ equals the domain ontology of G_2.

6. Implementation

Tab2KG is implemented in Java 1.8. The Siamese network is trained and applied using Keras in Python 3.7. We load knowledge graphs using Apache Jena[10]. We represent the column mappings inferred by *Tab2KG* in the RDF Mapping Language (RML) [21]. RML definitions are then utilized to materialize the data

[10]https://jena.apache.org/

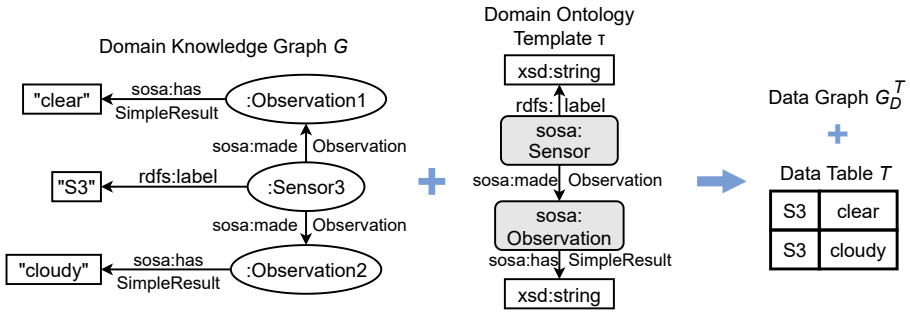

Fig. 10. Creation of a data table T and a data graph G_D^T from a domain knowledge graph G and a domain ontology template τ. For brevity, we omit type relations. In this particular example, G_T^D equals G.

graph. Data tables are provided as CSV files; knowledge graphs and data graphs as Turtle (*.ttl*) files.

6.1. Data Table Pre-Processing

Each data table interpreted in *Tab2KG* runs through three pre-processing steps:

1. **Data type identification**: For each table column, we identify its data type(s) by trying to parse more than 90% of the values as numeric, Boolean, spatial (Well-Known Text or Well-Known Binary format [22]) or temporal. If that is not possible, we assign the column to the text data type. For the more fine-grained data types mentioned in Section 4, we utilize regular expressions (text: URL or email), follow the algorithms proposed by Alobaid et al. [16] (numeric: sequential or categorical) or analyze the parsed objects (temporal: date, time, date-time; spatial: point, line string, polygon). We follow the algorithm in [16], using the threshold of not more than 20 different categories to detect categorical text values.
2. **Key column detection**: When we transform a data table into a data graph based on the RML mapping, we create new entities. In RDF, each entity is identified by a Unique Resource Identifier (URI). It is important to understand how to create these URIs, as we need to re-use URIs referring to the same entity: For example, each row in our running example in Fig. 1 forms a new instance of `sosa:Observation`, together with a new URI (e.g., `sosa:Observation7`), but there should only be three different `sosa:Sensor` URIs: `sosa:SensorS1`, `sosa:SensorS2`, `sosa:SensorS3`, created using the literal values assigned to the data type property `rdfs:label`. To create URIs, we detect data type relations representing the unique literal values of an entity as follows: (i) the data type relation is used on all instances of the class exactly once, and the literal values are unique across their instances. Currently, we do not consider the combinations of literal relations as identifiers [23]; we leave such combinations for future work.
3. **Identifier generation**: RML transformation requires referenceable columns and instances in data tables. Therefore, we automatically generate identifiers for each column and row of the data table. If available, column names are used as part of the column identifier.

6.1.1. Mapping Representation in RML

We utilize RML for storing column mappings inferred by *Tab2KG* in a machine-readable format such as Turtle. The RML defines subject maps that specify how to generate subjects for each row of the data table and predicate-object maps to create predicate-object pairs. In *Tab2KG*, the inferred column mappings are translated into the RML definitions according to the following four steps:

1. We create one instance of `rml:source` and `csvw:Table` each, denoting relevant characteristics for parsing the data table T (file location, delimiter, ...).
2. For each class part of the data graph G_D^T, we create a new instance of `rr:TriplesMap`, together with a `rr:subjectMap` that denotes the class as well as the target node URIs.
3. For each column mapping $c_n \mapsto r_D$, we create a `rr:predicateObjectMap` denoting the source column c_n, the data type property and a reference to the data type relation r_D.

4. For each class relation $r \in R_{OC}$ in the domain ontology G_O, we create a `rr:predicateObjectMap` connecting the respective entities and the object property.

6.1.2. Running Example: RML Mapping Definitions

Listing 1 in the Appendix provides an example of the RML definitions that were automatically generated for our running example introduced in Section 2 – without the time intervals, for brevity. Instances of `sosa:Sensor` and `sosa:Observation` are created alongside their relations. The sensor labels in the third column were detected as identifiers, i.e., we create node URIs as *https://www.w3.org/TR/vocab-ssn/Sensor{col3}*[11].

Listing 2 in the Appendix provides the resulting Turtle file representing the knowledge graph inferred from the input data table. The correct mapping of the third line shown in Fig. 4 is contained here.

7. Evaluation Setup

The goal of the evaluation is to assess the performance of *Tab2KG* concerning the semantic table interpretation effectiveness. In this section, we describe the datasets utilized for the evaluation and the baselines.

7.1. Datasets

For training and evaluating *Tab2KG*, we use several datasets.

The Siamese network training requires a dataset that spans over multiple domains and domain knowledge graphs, respectively, to ensure generalization. As the datasets typically used for semantic table interpretation only target a single domain or a cross-domain knowledge graph such as DBpedia, we created a new synthetic dataset automatically extracted from GitHub repositories dealing with knowledge graphs.

For testing, we consider the GitHub dataset and well-established datasets for semantic table interpretation that target specific domains (soccer and weapon advertisements), and DBpedia as a cross-domain knowledge graph.

[11] The RML template definitions do not allow to use prefixes.

7.1.1. Synthetic GitHub Dataset

To gain a dataset that covers a large variety of domains and schemas, we collect knowledge graphs from GitHub. The GitHub advanced code search[12] provides access to millions of data repositories. We selected files larger than 5KB with the specific file extensions[13] that contain the text "`xsd`" or "`XSDSchema`". To ensure the heterogeneity of our dataset, we limited the number of files per GitHub repository to three. Each file that was successfully parsed as a knowledge graph with more than 50 statements including at least 25 literal relations was added to our dataset. This way, we obtained $3,922$ files.

We set the parameter for maximum tree size $k = 3$, and the parameter for adding data type relations $\delta = 0.2$. The knowledge graphs set was split into a training set (90%) and a test set (10%). Knowledge graphs from the same repository were not included in the same set.

7.1.2. Test Datasets

We use the following datasets for evaluating our approach:

- **GitHub (GH):** The test split of the synthetic GitHub dataset, without a restriction on the used vocabularies.
- **Soccer (So):** 12 data sources regarding soccer players and their teams, annotated with the *schema.org* vocabulary [20].
- **Weapon Ads (WA):** 15 data sources about weapon advertisements, annotated with the *schema.org* vocabulary [24].
- **SemTab (ST):** A collection of data tables extracted from the T2Dv2 web table corpus, Wikipedia and others, annotated with the DBpedia ontology [25].
- **SemTab Easy (SE):** A subset of ST, including only data tables whose columns are mapped to one class only. Only classes appearing in the *T2KMatch* corpus [26] are included.

For all data tables contained in these datasets, we set the following constraints:

1. The input table file is parseable as a CSV file without errors.
2. There are no classes that are instantiated multiple times in the same row. This condition is to avoid cyclic structures. We discuss limitations regarding cyclic structures in Section 9.1.

[12] https://github.com/search/advanced

[13] `ttl`, `rdf`, `nt`, `nq`, `trix`, `n3`, `owl`

3. There is no pair of columns with identical values. This condition is to avoid randomness during the evaluation.

It is important to emphasize the difference in the evaluation setting compared to typical evaluation using the previously mentioned datasets such as ST: In the experiments conducted by [20, 27, 28], target general-purpose knowledge graphs such as DBpedia or Wikidata are given. Each data table in the test set is then mapped to the nodes in such a knowledge graph. In our evaluation setting, we assume that no data instances are given, i.e., an instance lookup is not possible. Instead, a domain profile and a domain ontology are provided. For evaluation, we select data table pairs, such that one data table mimics the domain knowledge graph from which we can derive a domain profile.

Following our setting defined in Definition 5 and illustrated in Fig. 8, the datasets are transformed into a set of triples (G, T, G_D^T), consisting of a data table T, a domain knowledge graph G and the mapping definition which transforms data table T into the data graph G_D^T. Technically, we extract a set of test instances, consisting of (i) a *.ttl* file representing the domain knowledge graph G, and (ii) a *.csv* file representing the data table T and a *.rml* file representing the mapping from T to G's domain ontology. To transform the datasets into such test instances, we identify pairs of data tables where the columns of the first data table are a subset of the second data table's columns. Then, the second data table represents the domain knowledge. Table 1 provides an overview of the datasets used during training and evaluation under these conditions.

We make the scripts to extract such evaluation datasets and the code for training the Siamese network publicly available[14].

7.2. Baselines

We compare *Tab2KG* against the following three semantic table interpretation baselines:

- ***DSL***: The Domain-independent Semantic Labeler [20] uses logistic regression on a set of hand-crafted features; some of them compare value pairs at the instance-level. It has been shown to outperform previous approaches such as the SemanticTyper [29]. We train *DSL* on the same GitHub training data set as *Tab2KG*.
- ***DSL****: The Domain-independent Semantic Labeler without using value similarity at the instance level. In contrast to *Tab2KG* and the other baselines, *DSL* utilizes a domain-specific data graph. As *Tab2KG* is solely based on the domain profile, *DSL* is in an advantageous setting that does not entirely reflect our setting. Therefore, we remove features at the instance-level for the *DSL** baseline.
- ***T2KMatch***: *T2KMatch* [26] performs semantic table interpretation on the instance level. In contrast to other approaches [10, 11, 28] that rely on a costly knowledge graph lookup at runtime, *T2KMatch* creates an index over the instances of frequently used DBpedia classes and is thus commonly used as a baseline for semantic table interpretation approaches [12, 30, 31]. It combines a ranking of entities found in the lookup phase for column type identification and data type-specific similarity measures (Levenshtein distance for strings, deviation for numbers, and deviation of years for dates) for property identification. *T2KMatch* assumes that a table only describes one entity class at a time and thus does not consider class relations.

8. Evaluation Results

In this section, we describe the training performance and the evaluation results based on the evaluation setup described before.

8.1. Accuracy of the Column Mapping Function

We train our Siamese network for $1,000$ epochs with a batch rate of 100 and a learning rate of 0.00006, following Hsiao et al.'s approach for one-shot image classification [32]. We use 256 dimensions for the hidden layer. 10% of the training dataset are used for validation. The feature vectors contain histograms with 10 buckets.

After training for all epochs on the synthetic training dataset, the Siamese network has an accuracy of 0.84 and a loss (binary cross-entropy) of 0.48 on the validation set. On the test set, it achieves an accuracy of 0.76, when treating scores of greater than 0.5 as candidates for column mapping.

[14] https://github.com/sgottsch/Tab2KG

Table 1

Datasets used in the evaluation. # is the number of (G, T, G_D^T) triples. The other columns contain average values. For example, the tables in ST dataset have about four columns on average.

		Tables	Domain Knowledge Graphs	
Dataset	**#**	**Columns**	**Data Type Relations**	**Class Relations**
GitHub (Training)	867	2.18	3.03	1.35
GH: GitHub (Test)	98	2.24	3.16	1.44
So: Soccer	15	6.25	9.38	2.75
WA: Weapon-Ads	16	10.57	11.2	4.4
ST: SemTab	233	4.09	5.54	1.26
SE: SemTab Easy	125	4.2	5.53	0.0

Table 2

Semantic table interpretation performance of *Tab2KG*, compared to the baselines on five datasets. We report the accuracy, i.e. the percentage of correctly identified data type relations (R_{OD}) and class relations (R_{OC}) in the datasets.

	GH	**So**	**WA**	**ST**	**SE**	**Average**
DSL	0.89	0.65	0.38	0.62	0.61	0.67
DSL*	0.87	0.43	0.44	0.66	0.71	0.70
T2KMatch	-	-	-	-	-	0.44
Tab2KG	0.88	0.64	0.48	0.78	0.78	**0.79**

8.2. Semantic Table Interpretation Results

We evaluate the performance of the semantic table interpretation achieved by *Tab2KG* compared to the baselines. Table 2 shows how the approaches perform on the different datasets, measured using accuracy, i.e., the percentage of the columns correctly mapped to data type relations and correctly identified class relations. We do not evaluate the performance of *T2KMatch* on other datasets than SE, as *T2KMatch* assumes one entity class per table only.

As we can observe in Table 2, the accuracy of the approaches varies considerably across the datasets, which can be explained by the different dataset characteristics shown in Table 1. In all cases except for the GH and So datasets, where *Tab2KG* and *DSL* show similar performance, *Tab2KG* achieves higher accuracy than the baselines concerning column mapping. Even though *Tab2KG* utilizes less information than *DSL*, *Tab2KG* performs better by 12 percentage points on average on this task.

Surprisingly, *DSL* is also outperformed by *DSL** on three of the five datasets (WA, ST, SE). To explain this behavior, we have computed the percentage of table values that also appear in the mapped data table relations: GH (33.38%), So (16.92%), WA (2.65%), ST (14.67%), SE (10.63%). This observation shows that profile-based semantic table interpretation can outperform instance-level approaches when the overlap between the data table and the instances in the domain knowledge graph is low.

Second, we assess the results of the column mapping (percentage of correctly mapped columns to the data type relations R_{OD}) and the graph creation (correctly identified class relations R_{OC}) in isolation. We report the results achieved by *Tab2KG* in comparison to the baselines in Table 3. In the case of column mapping, *Tab2KG* performs best on average, outperforming *DSL* by 10 percentage points. In general, the class relation mapping results in less accuracy than the column mapping. One reason is that errors propagate along the pipeline, i.e., a wrongly mapped data type relation invokes an erroneous class relation mapping.

8.3. Error Analysis

By inspection of the results, we have identified two typical sources of erroneous results in *Tab2KG*: (i) Value formatting: For example, the soccer dataset has data tables with column values such as “Germany", whereas the domain knowledge graphs had “GER" as a country label. Thus, the respective profile features

Table 3

Semantic table interpretation performance of *Tab2KG* in detail, compared to the baselines on five datasets, reported as the accuracy of class relations (R_{OC}) and data type relations (R_{OD}). Averages are computed in relation to the number of class relations and data type relations in the datasets, respectively.

	GH		**So**		**WA**		**ST**		**SE**	**Avg**	
	R_{OC}	R_{OD}	R_{OC}	R_{OD}	R_{OC}	R_{OD}	R_{OC}	R_{OD}	R_{OC}	R_{OC}	R_{OD}
DSL	0.93	0.62	0.42	0.46	0.45	0.40	0.65	0.70	0.71	0.67	0.57
DSL*	0.94	0.71	0.60	0.81	0.33	0.47	0.60	0.87	0.61	0.62	**0.71**
T2KMatch	-	-	-	-	-	-	-	-	0.44	0.44	-
Tab2KG	0.92	0.71	0.61	0.73	0.59	0.24	0.77	0.89	0.78	**0.77**	0.64

were highly different. In the case of high-quality domain knowledge graphs that, for example, distinguish between labels and abbreviations, the error rate should be lower. (ii) Data type differences: For example, the SemTab dataset has elevations of mountains both denoted via integer values ("5291") and as text ("2,858 ft (871 m)"). Again, the proper use of an ontology and its `rdfs:range` constraints on properties should alleviate this problem.

Overall, our evaluation results demonstrate that domain profiles in combination with zero-shot learning adopted by *Tab2KG* are an effective method for semantic table interpretation. This method does not require any instance lookup and achieves the highest accuracy on several datasets compared to the baselines.

9. Discussion

In this article, we presented *Tab2KG* – an approach for tabular data semantification. *Tab2KG* relies on domain profiles that enrich the relations in a domain ontology and serve as a lightweight domain representation. *Tab2KG* matches these profiles with tabular data using one-shot learning. Our evaluation shows that *Tab2KG* outperforms the baselines for semantic table interpretation of five real-world datasets. In future work, we plan to consider integrating user feedback into the *Tab2KG* pipeline to support an extension of the domain ontology in cases where tabular data contains previously unseen relations.

The representation of data tables as data graphs opens up several possibilities for making the lives of data scientists easier – with benefits including increased efficiency and robustness of data analytics workflows. Automating semantic table interpretation, *Tab2KG* takes an essential step in assisting domain experts and adds an important layer of abstraction to DAWs.

9.1. Limitations

We identified few limitations of *Tab2KG*, which can be attributed to the idea of using semantic lightweight dataset profiles, without requiring knowledge about particular data instances.

9.1.1. Column Mapping without Knowing the Dataset Instances

In contrast to approaches that perform semantic table interpretation at the instance level, i.e., with the help of the instance lookup in the domain knowledge graph, *Tab2KG* derives column mappings from statistical features in the domain profile. We have identified two limitations to this approach:

- Cyclic class relations: Currently, we do not address cyclic relations in the domain ontology, as for example (`dbo`[15]`:Event dbo:nextEvent dbo:Event`). Consider Fig. 11, where the second column provides the follow-up event of the event in the first column. Even if *Tab2KG* identifies the correct mapping for the third column to the property `dbo:locationCity`, we cannot tell if the third column maps to the location of the entity in the first or the second column.
- Class relations connecting the same classes: We do not have a decision criterion for distinguishing between class relations that connect the same subject and object classes. For example, consider the two object properties `dbo:leader` and `dbo:viceLeader` mapped to the second column in Fig. 12, both connecting countries to politicians. Only via statistical features extracted from the data table (which may only include the politician's name) it does not appear possible to decide

[15] http://dbpedia.org/ontology/

if Kamala Harris is the president or the vice president of the United States.

```
Olympics 2004 ↠ Olympics 2008 ↠ Athens
Olympics 2012 ↠ Olympics 2016 ↠ London
```

Fig. 11. Cyclic class relation: Did the London Olympic Games happen in 2012 or in 2016?

```
USA    ↠ Kamala Harris
Russia ↠ Michail Mischustin
```

Fig. 12. Class relations connecting the same classes: Is Kamala Harris the president or the vice president of the US?

9.1.2. *Correlations between Columns and Data Type Relations*

Our data table profiles consist of column profiles, i.e., the features of the single columns are computed in isolation (the same applies to domain profiles and data type relations). Such column profiles can be efficiently computed and added to the dataset profile. However, the dependencies between columns may hold additional knowledge. Consider the running example in Fig. 1 where the time in the second column (begin time) does always precede the time in the next column (end time), i.e., there is a correlation between the values in these two columns.

We have decided against the inclusion of correlation features into the domain and data table profiles because of the following reasons: First, correlations are often implicitly captured by the column profiles (e.g., in Fig. 1, the second column's mean value is less than the third column's mean value). Second, the variety of data types requires different correlation and dependency measures that are hard to compare. Third, we observed that the number of column pairs heavily exceeds the number of potentially meaningful correlations in our datasets. For example, consider the football dataset where the length of the first names may be compared to the length of last names, team names, the number of goals, ..., potentially leading to correlations by chance. Fourth, the computation and semantic representation of all possible column combinations are impractical due to the quadratic number of pairwise comparisons.

9.1.3. *Asymmetry between Domain Profiles and Data Table Profiles*

The domain profile and the data table profile can vary, even though they represent the same knowledge. Consider our running example of weather observations. In the table shown in Fig. 1, three rows refer to the sensor labeled "S1" but only two rows refer to the other sensors. When modeled as a knowledge graph following the mapping shown in Fig. 4, each sensor is modeled precisely as one node in the knowledge graph. Consequently, the statistical characteristics related to the sensors vary between the data table profile and the domain profile.

9.2. *Lightweight Semantic Dataset Profiles*

Lightweight semantic profiles generated by *Tab2KG* can be utilized as a compact domain and dataset representation to complement and enrich existing dataset catalogs. Such profiles can be generated automatically from the existing datasets and described using the DCAT[16] and the SEAS[17] vocabularies to facilitate their reusability. We believe that lightweight semantic profiles presented in this article are an essential contribution that can benefit a wide range of semantic applications beyond semantic table interpretation.

10. Related Work

This section provides an overview of related approaches in the areas of dataset profiling and semantic table interpretation.

Given the growth of data available on the Web and in industrial data lakes, there is a high demand for dataset profiling, e.g., for creating data catalogs [33]. The profile features typically belong to several categories, including statistical observations at the instance and schema level [15, 34]. Such features are not restricted to the initially defined schemas. For example, Neumaier et al. demonstrate how user interaction and search functionalities profit from the inclusion of spatio-temporal features into a dataset profile [35]. For tabular data, other approaches for dataset profile enrichment include the generation of table titles [36] and schema labels [37]. The inferred relation-specific rules and observations can further verify the data quality and become part of dataset profiles [38, 39].

Recently, approaches to annotate tabular data with concepts from a knowledge base to predict column types gained increased attention. In the following, we introduce approaches for semantic table interpretation and the methods they use.

[16] https://www.w3.org/TR/vocab-dcat-2/
[17] https://ci.mines-stetienne.fr/seas/index.html

Instance-level lookup. Most semantic table interpretation tools require access to a target knowledge graph, as they link data table cells to its resources [2]. Such approaches on the instance-level have recently been driven by the SemTab Challenge [25, 40], which explicitly postulates a cell-entity annotation (CEA) task, where labels in data table cells are linked to entities in a target knowledge graph. The subsequent steps of column-type annotation (CTA) and columns-property annotation typically build upon the CEA results.

Several approaches are based on entity lookup (ColNet [12], MantisTable [11], LinkingPark [27], DAGOBAH [41, 42], MTab [10], T2KMatch [26], CSV2KG [43], TableMiner+ [28], and the work by Zhang et al. [30]), with different (combined) query strategies, including URL matching [43], (partial) string lookup [11, 27, 42], string similarity [10, 26, 30, 42], spelling correction [27] or the use of named entity linking tools [11]. After linking data table cells to resources in the knowledge graphs, the CTA is typically decided through voting or counting [10, 12, 27], ranking [11, 28, 43] or clustering [42]. ColNet and TableMiner+ apply learning strategies to reduce the number of lookup tasks required for detecting the class of the entities represented by a column. MantisTable and CSV2KG utilize concept graphs in their ranking to identify the most-specific sub classes. Also, identifying properties represented by the data tables typically relies on the CEA and additional knowledge graph lookups. For example, MTab does pairwise queries between entities identified in different columns to identify potential entity relations. To identify literal relations, MTab and TableMiner+ row-by-row compute data-type specific similarities between the literals in the target knowledge graph and the cell values. CSV2KG also involves the target ontology in this step. Sherlock [19] is a system that performs CTA and does not rely on CEA. However, it extracts column features for training a neural network, which is solely trained on DBpedia and explicitly predicts one of the selected DBpedia classes.

The reliance on entity linking with the target knowledge graph makes these approaches unsuitable in settings where data tables only contain previously unseen data, which is a common issue [14]. Even if the data instances in the data table are not entirely unknown, these approaches do not perform well when the number of matching entities drops [12]. Another thing these approaches have in common is the reliance on a large underlying knowledge base such as DBpedia and stable lookup services. In contrast, *Tab2KG* does not require access to the target knowledge graph after the domain profile has been created.

Subject column detection. Several approaches [11, 28] for semantic table interpretation assume the existence of a subject column, i.e., the main column of the data table where every other column is directly connected to. The subject column detection is typically identified through a set of statistic features. Approaches relying on a subject column do not consider the involvement of any classes which are not directly represented in the data table (for example, consider Fig. 1, but without the first column). *Tab2KG* utilizes a graph-based approach where such class relations can be found.

Column titles. Some data tables come with column titles, which may indicate respective classes or properties. Efthymiou et al. [31] propose a method based on ontology matching, where one column title defines the class label, and other column titles represent property labels. Domain-independent Semantic Labeler [20], DAGOBAH [42] and TableMiner+ [28] exploit column titles as one of their features. *Tab2KG* does not require any column titles. This way, we ensure the generalizability for data tables without headers and language-independence.

Data type restrictions. Data tables contain data of various types, and thus there are approaches specific to some of them. For example, EmbNum+ [44] transforms data table columns with numeric values into embedding vectors. Alobaid et al. demonstrate that using more fine-grained numeric data types increases semantic table interpretation performance for numeric column values [16]. For the interpretation of cross-lingual textual values, Luo et al. propose using several translation tools [45]. *Tab2KG* aims at the interpretation of data tables as a whole without restricting to particular data types or languages and thus establishes profiles that do not depend on particular data types or languages.

User feedback. Instead of relying on fully-automated approaches for semantic table interpretation, which may be error-prone due to the challenges involved in this task, manual or semi-automated approaches rely on user feedback. Karma [46], Odalic [47], and ASIA [48] are interactive tools that let users decide on the correctness of suggested table annotations and thus achieve high precision, but demand both time and expertise from the user. *Tab2KG* is a fully-automated approach for semantic table interpretation that does not require user interventions.

Domain-independent semantic table interpretation. Domain-independent approaches are not restricted to specific target knowledge graphs. Instead, they learn domain-independent similarity features to generate the mapping. The SemanticTyper [29] scores similarity between columns and data type relations based on handcrafted features for numeric and textual values. Based on similar features, the Domain-independent Semantic Labeler [20] adopts machine learning and handcrafted features to predict the similarity between a column and a class in the domain knowledge graph. Taheriyan et al. [49] generate a ranked list of potential column mappings learned from a sample of the domain ontology, which is then presented to the user. While both approaches are flexible concerning the target domain, *Tab2KG* aims to use only features present in the dataset profiles.

11. Conclusion

This article presented *Tab2KG* - an approach for semantic table interpretation based on lightweight semantic profiles. *Tab2KG* relies on domain profiles that enrich the relations in a domain ontology and serve as a semantic domain representation. *Tab2KG* matches these profiles with the tabular data profiles using one-shot learning approach. Our evaluation on five real-world datasets shows that our approach outperforms the baselines for semantic table interpretation. In future work, we plan to consider integrating user feedback into the *Tab2KG* pipeline to support an extension of the domain ontology in cases where tabular data contains previously unseen relations.

Acknowledgements

This work is partially funded by the DFG, German Research Foundation (“WorldKG", DE 2299/2-1, 424985896), the Federal Ministry of Education and Research (BMBF), Germany (“Simple-ML", 01IS18054) and the European Commission (EU H2020, “smashHit", grant ID 871477).

References

[1] M.J. Cafarella, A.Y. Halevy, H. Lee, J. Madhavan, C. Yu, D.Z. Wang and E. Wu, Ten Years of WebTables, *Proc. VLDB Endow.* **11**(12) (2018), 2140–2149, DOI: https://doi.org/10.14778/3229863.3240492.

[2] D. Ritze and C. Bizer, Matching Web Tables to DBpedia - a Feature Utility Study (2017), 210–221, DOI: https://doi.org/10.5441/002/edbt.2017.20.

[3] J. Mitlöhner, S. Neumaier, J. Umbrich and A. Polleres, Characteristics of Open Data CSV Files, in: *2nd International Conference on Open and Big Data, OBD 2016, Vienna, Austria, August 22-24, 2016*, I. Awan and M. Younas, eds, IEEE Computer Society, 2016, pp. 72–79, DOI: https://doi.org/10.1109/OBD.2016.18.

[4] S. Gottschalk, N. Tempelmeier, G. Kniesel, V. Iosifidis, B. Fetahu and E. Demidova, Simple-ML: Towards a Framework for Semantic Data Analytics Workflows, in: *Semantic Systems. The Power of AI and Knowledge Graphs - 15th International Conference, SEMANTiCS 2019, Karlsruhe, Germany, September 9-12, 2019, Proceedings*, M. Acosta, P. Cudré-Mauroux, M. Maleshkova, T. Pellegrini, H. Sack and Y. Sure-Vetter, eds, Lecture Notes in Computer Science, Vol. 11702, Springer, 2019, pp. 359–366, DOI: https://doi.org/10.1007/978-3-030-33220-4_26.

[5] M. Garda, A Semantics-Enabled Approach for Data Lake Exploration Services, in: *2019 IEEE World Congress on Services, SERVICES 2019, Milan, Italy, July 8-13, 2019*, C.K. Chang, P. Chen, M. Goul, K. Oyama, S. Reiff-Marganiec, Y. Sun, S. Wang and Z. Wang, eds, IEEE, 2019, pp. 327–330, DOI: https://doi.org/10.1109/SERVICES.2019.00091.

[6] A. Pomp, A. Paulus, A. Kirmse, V. Kraus and T. Meisen, Applying Semantics to Reduce the Time to Analytics within Complex Heterogeneous Infrastructures, *Technologies* **6**(3) (2018), 86, DOI: https://doi.org/10.3390/technologies6030086.

[7] C. Hartenfels, M. Leinberger, R. Lämmel and S. Staab, Type-Safe Programming with OWL in Semantics4J., in: *Proceedings of the ISWC 2017 Posters & Demonstrations and Industry Tracks co-located with 16th International Semantic Web Conference (ISWC 2017), Vienna, Austria, October 23rd - to - 25th, 2017*, N. Nikitina, D. Song, A. Fokoue and P. Haase, eds, CEUR Workshop Proceedings, Vol. 1963, CEUR-WS.org, 2017. http://ceur-ws.org/Vol-1963/paper549.pdf.

[8] F. Lécué, On the Role of Knowledge Graphs in Explainable AI, *Semantic Web* **11**(1) (2020), 41–51, DOI: https://doi.org/10.3233/SW-190374. https://doi.org/10.3233/SW-190374.

[9] J. Lehmann, G. Sejdiu, L. Bühmann, P. Westphal, C. Stadler, I. Ermilov, S. Bin, N. Chakraborty, M. Saleem, A.N. Ngomo and H. Jabeen, Distributed Semantic Analytics using the SANSA Stack, in: *The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part II*, C. d'Amato, M. Fernández, V.A.M. Tamma, F. Lécué, P. Cudré-Mauroux, J.F. Sequeda, C. Lange and J. Heflin, eds, Lecture Notes in Computer Science, Vol. 10588, Springer, 2017, pp. 147–155, DOI: https://doi.org/10.1007/978-3-319-68204-4_15.

[10] P. Nguyen, N. Kertkeidkachorn, R. Ichise and H. Takeda, MTab: Matching Tabular Data to Knowledge Graph using Probability Models, in: *Proceedings of the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching co-located with the 18th International Semantic Web Conference, SemTab@ISWC 2019, Auckland, New Zealand, October 30, 2019*, E. Jiménez-Ruiz, O. Hassanzadeh, K. Srinivas, V. Efthymiou and J. Chen, eds, CEUR Workshop Proceedings, Vol. 2553, CEUR-WS.org, 2019, pp. 7–14. http://ceur-ws.org/Vol-2553/paper2.pdf.

[11] M. Cremaschi, F.D. Paoli, A. Rula and B. Spahiu, A Fully Automated Approach to a Complete Semantic Table Interpretation, *Future Gener. Comput. Syst.* **112** (2020), 478–500, DOI: https://doi.org/10.1016/j.future.2020.05.019.

[12] J. Chen, E. Jiménez-Ruiz, I. Horrocks and C. Sutton, ColNet: Embedding the Semantics of Web Tables for Column Type Prediction, in: *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, AAAI Press, 2019, pp. 29–36, DOI: https://doi.org/10.1609/aaai.v33i01.330129.

[13] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak and Z.G. Ives, DBpedia: A Nucleus for a Web of Open Data, in: *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007*, K. Aberer, K. Choi, N.F. Noy, D. Allemang, K. Lee, L.J.B. Nixon, J. Golbeck, P. Mika, D. Maynard, R. Mizoguchi, G. Schreiber and P. Cudré-Mauroux, eds, Lecture Notes in Computer Science, Vol. 4825, Springer, 2007, pp. 722–735, DOI: https://doi.org/10.1007/978-3-540-76298-0_52.

[14] D. Ritze, O. Lehmberg, Y. Oulabi and C. Bizer, Profiling the Potential of Web Tables for Augmenting Cross-domain Knowledge Bases, in: *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11 - 15, 2016*, J. Bourdeau, J. Hendler, R. Nkambou, I. Horrocks and B.Y. Zhao, eds, ACM, 2016, pp. 251–261, DOI: https://doi.org/10.1145/2872427.2883017.

[15] M.B. Ellefi, Z. Bellahsene, J.G. Breslin, E. Demidova, S. Dietze, J. Szymanski and K. Todorov, RDF Dataset Profiling–a Survey of Features, Methods, Vocabularies and Applications, *Semantic Web* **9**(5) (2018), 677–705, DOI: https://doi.org/10.3233/SW-180294.

[16] A. Alobaid, E. Kacprzak and Ó. Corcho, Typology-based Semantic Labeling of Numeric Tabular Data, *Semantic Web* **12**(1) (2021), 5–20, DOI: https://doi.org/10.3233/SW-200397.

[17] A. Harth, K. Hose, M. Karnstedt, A. Polleres, K. Sattler and J. Umbrich, Data Summaries for On-demand Queries over Linked Data, in: *Proceedings of the 19th International Conference on World Wide Web, WWW 2010, Raleigh, North Carolina, USA, April 26-30, 2010*, M. Rappa, P. Jones, J. Freire and S. Chakrabarti, eds, ACM, 2010, pp. 411–420, DOI: https://doi.org/10.1145/1772690.1772733.

[18] G. Koch, R. Zemel and R. Salakhutdinov, Siamese Neural Networks for One-shot Image Recognition, in: *Deep Learning Workshop, International Conference on Machine Learning '15*, Vol. 2, 2015.

[19] M. Hulsebos, K.Z. Hu, M.A. Bakker, E. Zgraggen, A. Satyanarayan, T. Kraska, Ç. Demiralp and C.A. Hidalgo, Sherlock: A Deep Learning Approach to Semantic Data Type Detection, in: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, A. Teredesai, V. Kumar, Y. Li, R. Rosales, E. Terzi and G. Karypis, eds, ACM, 2019, pp. 1500–1508, DOI: https://doi.org/10.1145/3292500.3330993.

[20] M. Pham, S. Alse, C.A. Knoblock and P.A. Szekely, Semantic Labeling: A Domain-Independent Approach, in: *The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part I*, P. Groth, E. Simperl, A.J.G. Gray, M. Sabou, M. Krötzsch, F. Lécué, F. Flöck and Y. Gil, eds, Lecture Notes in Computer Science, Vol. 9981, 2016, pp. 446–462, DOI: https://doi.org/10.1007/978-3-319-46523-4_27.

[21] A. Dimou, M.V. Sande, P. Colpaert, R. Verborgh, E. Mannens and R.V. de Walle, RML: A Generic Language for Integrated RDF Mappings of Heterogeneous Data, in: *Proceedings of the Workshop on Linked Data on the Web co-located with the 23rd International World Wide Web Conference (WWW 2014), Seoul, Korea, April 8, 2014*, C. Bizer, T. Heath, S. Auer and T. Berners-Lee, eds, CEUR Workshop Proceedings, Vol. 1184, CEUR-WS.org, 2014. http://ceur-ws.org/Vol-1184/ldow2014_paper_01.pdf.

[22] Information Technology – Database Languages – SQL Multimedia and Application Packages – Part 3: Spatial, Standard, International Organization for Standardization, 2016. https://www.iso.org/standard/60343.html.

[23] A. Heise, J. Quiané-Ruiz, Z. Abedjan, A. Jentzsch and F. Naumann, Scalable Discovery of Unique Column Combinations, *Proc. VLDB Endow.* **7**(4) (2013), 301–312, DOI: https://doi.org/10.14778/2732240.2732248. http://www.vldb.org/pvldb/vol7/p301-heise.pdf.

[24] M. Taheriyan, C.A. Knoblock, P.A. Szekely and J.L. Ambite, Leveraging Linked Data to Discover Semantic Relations Within Data Sources, in: *The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part I*, P. Groth, E. Simperl, A.J.G. Gray, M. Sabou, M. Krötzsch, F. Lécué, F. Flöck and Y. Gil, eds, Lecture Notes in Computer Science, Vol. 9981, 2016, pp. 549–565, DOI: 10.1007/978-3-319-46523-4_33. https://doi.org/10.1007/978-3-319-46523-4_33.

[25] E. Jiménez-Ruiz, O. Hassanzadeh, V. Efthymiou, J. Chen and K. Srinivas, SemTab 2019: Resources to Benchmark Tabular Data to Knowledge Graph Matching Systems **12123** (2020), 514–530, DOI: https://doi.org/10.1007/978-3-030-49461-2_30. https://doi.org/10.1007/978-3-030-49461-2_30.

[26] D. Ritze, O. Lehmberg and C. Bizer, Matching HTML Tables to DBpedia, in: *Proceedings of the 5th International Conference on Web Intelligence, Mining and Semantics, WIMS 2015, Larnaca, Cyprus, July 13-15, 2015*, R. Akerkar, M.D. Dikaiakos, A. Achilleos and T. Omitola, eds, ACM, 2015, pp. 10:1–10:6, DOI: https://doi.org/10.1145/2797115.2797118.

[27] S. Chen, A. Karaoglu, C. Negreanu, T. Ma, J. Yao, J. Williams, A. Gordon and C. Lin, LinkingPark: An Integrated Approach for Semantic Table Interpretation **2775** (2020), 65–74. http://ceur-ws.org/Vol-2775/paper7.pdf.

[28] Z. Zhang, Effective and efficient Semantic Table Interpretation using TableMiner$^+$, *Semantic Web* **8**(6) (2017), 921–957, DOI: https://doi.org/10.3233/SW-160242.

[29] S.K. Ramnandan, A. Mittal, C.A. Knoblock and P.A. Szekely, Assigning Semantic Labels to Data Sources, in: *The Semantic Web. Latest Advances and New Domains - 12th European Semantic Web Conference, ESWC 2015, Portoroz, Slovenia, May 31 - June 4, 2015. Proceedings*, F. Gandon, M. Sabou, H. Sack, C. d'Amato, P. Cudré-Mauroux and A. Zimmermann, eds, Lecture Notes in Computer Science, Vol. 9088, Springer, 2015, pp. 403–417, DOI: 10.1007/978-3-319-18818-8_25.

[30] S. Zhang, E. Meij, K. Balog and R. Reinanda, Novel Entity Discovery from Web Tables, in: *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, Y. Huang, I. King, T. Liu and M. van Steen, eds, ACM / IW3C2, 2020, pp. 1298–1308, DOI: https://doi.org/10.1145/3366423.3380205.

[31] V. Efthymiou, O. Hassanzadeh, M. Rodriguez-Muro and V. Christophides, Matching Web Tables with Knowledge Base Entities: From Entity Lookups to Entity Embeddings, in: *The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part I*, C. d'Amato, M. Fernández, V.A.M. Tamma, F. Lécué, P. Cudré-Mauroux, J.F. Sequeda, C. Lange and J. Heflin, eds, Lecture Notes in Computer Science, Vol. 10587, Springer, 2017, pp. 260–277, DOI: 10.1007/978-3-319-68288-4_16.

[32] S. Hsiao, D. Kao, Z. Liu and R. Tso, Malware Image Classification using One-shot Learning with Siamese Networks **159** (2019), 1863–1871, DOI: https://doi.org/10.1016/j.procs.2019.09.358.

[33] S. Neumaier, J. Umbrich and A. Polleres, Automated Quality Assessment of Metadata across Open Data Portals, *ACM J. Data Inf. Qual.* **8**(1) (2016), 2:1–2:29, DOI: https://doi.org/10.1145/2964909. https://doi.org/10.1145/2964909.

[34] Z. Abedjan, L. Golab and F. Naumann, Profiling Relational Data: a Survey, *VLDB J.* **24**(4) (2015), 557–581, DOI: https://doi.org/10.1007/s00778-015-0389-y.

[35] S. Neumaier and A. Polleres, Enabling Spatio-Temporal Search in Open Data, *J. Web Semant.* **55** (2019), 21–36, DOI: https://doi.org/10.1016/j.websem.2018.12.007.

[36] B. Hancock, H. Lee and C. Yu, Generating Titles for Web Tables, in: *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, L. Liu, R.W. White, A. Mantrach, F. Silvestri, J.J. McAuley, R. Baeza-Yates and L. Zia, eds, ACM, 2019, pp. 638–647, DOI: https://doi.org/10.1145/3308558.3313399.

[37] Z. Chen, H. Jia, J. Heflin and B.D. Davison, Generating Schema Labels through Dataset Content Analysis, in: *Companion of the The Web Conference 2018 on The Web Conference 2018, WWW 2018, Lyon , France, April 23-27, 2018*, P. Champin, F.L. Gandon, M. Lalmas and P.G. Ipeirotis, eds, ACM, 2018, pp. 1515–1522, DOI: https://doi.org/10.1145/3184558.3191601.

[38] S. Schelter, D. Lange, P. Schmidt, M. Celikel, F. Bießmann and A. Grafberger, Automating Large-scale Data Quality Verification, *Proc. VLDB Endow.* **11**(12) (2018), 1781–1794, DOI: https://doi.org/10.14778/3229863.3229867.

[39] G. Sejdiu, A. Rula, J. Lehmann and H. Jabeen, A Scalable Framework for Quality Assessment of RDF Datasets, in: *The Semantic Web - ISWC 2019 - 18th International Semantic Web Conference, Auckland, New Zealand, October 26-30, 2019, Proceedings, Part II*, C. Ghidini, O. Hartig, M. Maleshkova, V. Svátek, I.F. Cruz, A. Hogan, J. Song, M. Lefrançois and F. Gandon, eds, Lecture Notes in Computer Science, Vol. 11779, Springer, 2019, pp. 261–276, DOI: https://doi.org/10.1007/978-3-030-30796-7_17.

[40] E. Jiménez-Ruiz, O. Hassanzadeh, V. Efthymiou, J. Chen, K. Srinivas and V. Cutrona, Results of SemTab 2020, in: *Proceedings of the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching (SemTab 2020) co-located with the 19th International Semantic Web Conference (ISWC 2020), Virtual conference (originally planned to be in Athens, Greece), November 5, 2020*, E. Jiménez-Ruiz, O. Hassanzadeh, V. Efthymiou, J. Chen, K. Srinivas and V. Cutrona, eds, CEUR Workshop Proceedings, Vol. 2775, CEUR-WS.org, 2020, pp. 1–8. http://ceur-ws.org/Vol-2775/paper0.pdf.

[41] Y. Chabot, T. Labbé, J. Liu and R. Troncy, DAGOBAH: An End-to-End Context-Free Tabular Data Semantic Annotation System **2553** (2019), 41–48. http://ceur-ws.org/Vol-2553/paper6.pdf.

[42] V. Huynh, J. Liu, Y. Chabot, T. Labbé, P. Monnin and R. Troncy, DAGOBAH: Enhanced Scoring Algorithms for Scalable Annotations of Tabular Data **2775** (2020), 27–39. http://ceur-ws.org/Vol-2775/paper3.pdf.

[43] B. Steenwinckel, G. Vandewiele, F. De Turck and F. Ongenae, CSV2KG: Transforming Tabular Data into Semantic Knowledge, in: *Proceedings of the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching co-located with the 18th International Semantic Web Conference (ISWC 2019)*, 2019. http://ceur-ws.org/Vol-2553/paper5.pdf.

[44] P. Nguyen, K. Nguyen, R. Ichise and H. Takeda, EmbNum+: Effective, Efficient, and Robust Semantic Labeling for Numerical Values, *New Gener. Comput.* **37**(4) (2019), 393–427, DOI: https://doi.org/10.1007/s00354-019-00076-w. https://doi.org/10.1007/s00354-019-00076-w.

[45] X. Luo, K. Luo, X. Chen and K.Q. Zhu, Cross-Lingual Entity Linking for Web Tables, in: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, S.A. McIlraith and K.Q. Weinberger, eds, AAAI Press, 2018, pp. 362–369.

[46] C.A. Knoblock, P.A. Szekely, J.L. Ambite, A. Goel, S. Gupta, K. Lerman, M. Muslea, M. Taheriyan and P. Mallick, Semi-automatically Mapping Structured Sources into the Semantic Web, in: *The Semantic Web: Research and Applications - 9th Extended Semantic Web Conference, ESWC 2012, Heraklion, Crete, Greece, May 27-31, 2012. Proceedings*, E. Simperl, P. Cimiano, A. Polleres, Ó. Corcho and V. Presutti, eds, Lecture Notes in Computer Science, Vol. 7295, Springer, 2012, pp. 375–390, DOI: https://doi.org/10.1007/978-3-642-30284-8_32.

[47] T. Knap, Towards Odalic, a Semantic Table Interpretation Tool in the ADEQUATe Project, in: *Proceedings of the 5th International Workshop on Linked Data for Information Extraction co-located with the 16th International Semantic Web Conference (ISWC 2017), Vienna, Austria, October 22, 2017*, A.L. Gentile, A.G. Nuzzolese and Z. Zhang, eds, CEUR Workshop Proceedings, Vol. 1946, CEUR-WS.org, 2017, pp. 26–37. http://ceur-ws.org/Vol-1946/paper-04.pdf.

[48] V. Cutrona, M. Ciavotta, F.D. Paoli and M. Palmonari, ASIA: a Tool for Assisted Semantic Interpretation and Annotation of Tabular Data **2456** (2019), 209–212. http://ceur-ws.org/Vol-2456/paper54.pdf.

[49] M. Taheriyan, C.A. Knoblock, P.A. Szekely and J.L. Ambite, Learning the Semantics of Structured Data Sources, *J. Web Semant.* **37-38** (2016), 152–169, DOI: https://doi.org/10.1016/j.websem.2015.12.003.

Appendix A. Running Example: RML Definitions

```
@prefix rr: <http://www.w3.org/ns/r2rml#>.
@prefix rml: <http://semweb.mmlab.be/ns/rml#>.
@prefix ex: <http://example.com/resource/>.
@prefix csvw: <http://www.w3.org/ns/csvw#>.

ex:File a rml:source ;
 rml:source ex:FileSource ;
 rml:referenceFormulation <http://semweb.mmlab.be/ns/ql#CSV> .

ex:FileSource a csvw:Table;
 csvw:url "sky_sensors.tsv" ;
 csvw:dialect [
  a csvw:Dialect;
  csvw:delimiter "↠";
 ] .

ex:Mapping0 a rr:TriplesMap ;
 rml:logicalSource ex:File ;
 rr:subjectMap [
  rr:class sosa:Sensor ;
  rr:template "https://www.w3.org/TR/vocab-ssn/Sensor{col3}" ;
 ];

 rr:predicateObjectMap [
  rr:predicate rdfs:label ;
  rr:objectMap [
   rml:reference "col3";
  ]
 ] ;

 rr:predicateObjectMap [
  rr:predicate sosa:madeObservation ;
  rr:objectMap [
   rr:template "https://www.w3.org/TR/vocab-ssn/Observation{rowNumber}";
  ]
 ] .

ex:Mapping1 a rr:TriplesMap ;
 rml:logicalSource ex:File ;
 rr:subjectMap [
  rr:class sosa:Observation ;
  rr:template "https://www.w3.org/TR/vocab-ssn/Observation{rowNumber}" ;
 ] .
```

Listing 1: A working example of an RML file transforming the data table given in Fig. 1 (“sky_sensors.tsv”) into the data graph indicated in Fig. 4. To perform the transformation, the table needs to be pre-processed: the column titles “col0”,...,“col3” were added, and a “rowNumber” column. For brevity, we skip the mapping of the second and third column to `time:Interval`.

```
<https://www.w3.org/TR/vocab-ssn/Observation0>
  a <http://www.w3.org/ns/sosa/Observation> .

<https://www.w3.org/TR/vocab-ssn/Observation1>
  a <http://www.w3.org/ns/sosa/Observation> .

<https://www.w3.org/TR/vocab-ssn/Observation2>
  a <http://www.w3.org/ns/sosa/Observation> .

<https://www.w3.org/TR/vocab-ssn/SensorS2>
  a <http://www.w3.org/ns/sosa/Sensor>;
  <http://www.w3.org/2000/01/rdf-schema#label> "S2";
  <http://www.w3.org/ns/sosa/madeObservation>
    <https://www.w3.org/TR/vocab-ssn/Observation0> .

<https://www.w3.org/TR/vocab-ssn/SensorS3>
  a <http://www.w3.org/ns/sosa/Sensor>;
  <http://www.w3.org/2000/01/rdf-schema#label> "S3";
  <http://www.w3.org/ns/sosa/madeObservation>
      <https://www.w3.org/TR/vocab-ssn/Observation1>,
      <https://www.w3.org/TR/vocab-ssn/Observation2> .
```

Listing 2: The Turtle file resulting from running the RML file in Listing 1 to transform the table given in Fig. 1 into a data graph. For brevity, we only consider the first three lines of the data table.