Discovering Types in RDF Datasets

Kenza Kellou-Menouer(✉) and Zoubida Kedad(✉)

PRISM - University of Versailles Saint-Quentin-en-Yvelines, Versailles, France
{kenza.menouer,zoubida.kedad}@prism.uvsq.fr

Abstract. An increasing number of linked datasets is published on the Web, expressed in RDF(S)/OWL. Interlinking, matching or querying these datasets require some knowledge about the types and properties they contain. This work presents an approach, relying on a clustering algorithm, which provides the types describing a dataset when this information is incomplete or missing.

Keywords: Type extraction · Clustering · Semantic web · Linked data

1 Introduction

An increasing number of linked datasets is published on the Web. Understanding these datasets is crucial in order to exploit them. Having some knowledge about the content of a dataset, such as the types it contains, is crucial for users and applications as it will enable many tasks, such as creating links between datasets or querying them. Linked datasets are not always complete with respect to type information. Even when they are automatically extracted from a controlled source, type information can be missing: in DBpedia (extracted from Wikipedia), 63.7 % of type information is provided [8].

Our goal is to infer the types describing an RDF(S)/OWL dataset. Our main contribution is a deterministic and automatic approach relying on a clustering algorithm to extract types, where several types can be assigned to an entity. Our approach does not require any schema related information in the dataset. We have implemented our algorithms and we present some experimental evaluation results to demonstrate the effectiveness of the approach.

2 Type Discovery

In order to infer the types from a dataset, our approach relies on grouping entities according to their similarity. A group of similar entities corresponds to a type definition. The similarity between two given entities is evaluated considering their respective sets of both incoming and outgoing properties.

Our main requirements are the following: (i) the number of types is not known in advance, (ii) an entity can have several types, and (iii) the datasets may contain noise. The most suitable grouping approach is density-based clustering,

F. Gandon et al. (Eds.): ESWC 2015, LNCS 9341, pp. 77–81, 2015.
DOI: 10.1007/978-3-319-25639-9_15

introduced by [2], because it is robust to noise, deterministic and it finds classes of arbitrary shape. In addition, unlike the algorithms based on k-means and k-medoid, the number of classes is not required.

Our density-based algorithm has two parameters: the maximum radius of neighborhood ε and the minimum number of neighbors for an entity $MinPts$. ε represents the minimum similarity value for two entities to be considered as neighbors. We use Jaccard similarity to measure the closeness between two property sets describing two entities. $MinPts$ is the minimum number of similar entities required to form a core [2]: an entity is not assigned to a class if it is considered as noise, i.e. if it is neither a core itself nor the neighbor of a core.

In order to speed up the clustering process, we perform once and for all the calculation of the nearest neighbors of each entity [4]. We store a neighborhood matrix containing for each entity the ordered list of its neighbors. It is then straightforward to find the nearest neighbors for an entity at a distance lower than ε, with a linear complexity o(n) [5].

Each type is described by a profile, which is a property vector where each property is associated to a probability. The profile corresponding to a type T_i is denoted $TP_i =< (p_{i1}, \alpha_{i1}), ..., (p_{in}, \alpha_{in}) >$, where each p_{ij} represents a property and where each α_{ij} represents the probability for an entity of T_i to have the property p_{ij}. The type profile represents the canonical structure of type T_i.

An important aspect of RDF(S)/OWL datasets is that an entity may have several types [8]. We provide overlapping types by analyzing the type profiles: intuitively, if an entity e is described by properties characterizing several types, then these types could be assigned to e. However, the properties have different levels of confidence, which has to be considered. Indeed, if σ represents a threshold above which the probability associated to a property is considered as high, and if all the properties p of the type T_i having a probability $\alpha_i > \sigma$ are also properties of another type T_k, then T_i is also a type for the entities in T_k.

3 Related Works

Type inference from structureless and semi-structured data has been addressed by some works in the literature. In [10], an approximate DataGuide based on an incremental hierarchical clustering (COBWEB) is proposed in order to group similar nodes, i.e. the ones having the same incoming/outgoing edges. The approach considers both types of edges in the same way, which could be a problem if applied to RDF datasets as it will not differentiate between the domain and the range of properties. The resulting classes are disjoints, and the approach is not deterministic as it is based on COBWEB. The approach presented in [6] uses bottom-up grouping providing a set of disjoint classes. A similarity threshold has to be set, as well as the number of clusters, unlike in our approach. In [1] standard ascending hierarchical clustering is used to build structural summaries of linked data. Each instance is represented by its outgoing properties only and the property set of a class is the union of the properties of all its entities, unlike our approach where the probability of each property is computed

for a type. The algorithm provides disjoint classes and it is costly, in addition, the method explores the hierarchical clustering tree to assess the best cutoff level. SDType [8] enriches an entity by several types using inference rules, and computes the confidence of a type for an entity. The focus of the approach is more on the evaluation of the relevance of the inferred types rather than finding these types. In addition, *rdfs:domain*, *rdfs:range* and *rdfs:subClassOf* properties are required. The approach does not introduce new types, but considers instead the ones already assigned to an entity in the dataset. Works in [3,7] infer types for the DBpedia dataset only: [7] uses K-NN and [3] finds the most appropriate type of an entity in DBpedia based on descriptions from Wikipedia and links with WordNet and Dolce ontologies. A Statistical Schema Induction approach [9] enriches a RDF dataset with the RDFS/OWL primitives, however the classes must be pre-defined and expressed as *rdf:type* declarations.

4 Evaluation

We have used the Conference[1] dataset, which exposes data for several semantic web related conferences and workshops. We have also used a dataset extracted from DBpedia considering the following types: Politician, SoccerPlayer, Museum, Movie, Book and Country.

We have extracted the existing type definitions from our dataset and considered them as a gold standard. We have then run our algorithm on the dataset without the type definitions and evaluated the precision and recall for the inferred types. We have annotated each inferred class C_i with the most frequent type label associated to its entities. For each type label L_j corresponding to type T_j in the dataset and each class C_i inferred by our algorithm, such that L_j is the label of C_i, we have evaluated the precision $P(T_j, C_i) = |T_j \cap C_i|/|C_i|$ and recall $R(T_j, C_i) = |T_j \cap C_i|/|T_j|$. We have set $\varepsilon = 0.5$ and $MinPts = 1$ so that an entity is considered as noise if it is completely isolated.

The resulting values of the metrics are shown in Fig. 1. For the Conference dataset, our approach gives good results and detects types which were not declared in the dataset, annotated as follows: classes 7, 8, 9 and 10 are labeled 'AuthorList', 'PublicationPage', 'HomePage' and 'City' respectively. In

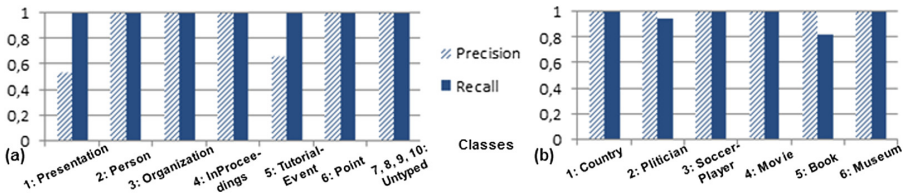

Fig. 1. Quality evaluation on the conference (a) and DBpedia (b) datasets.

[1] Conference: data.semanticweb.org/dumps/conferences/dc-2010-complete.rdf.

some cases, types have been inferred relying on incoming properties only. Indeed, for containers, such as 'AuthorList', it is necessary to consider these properties as they do not have any outgoing property. Classes 1 and 5 do not have a good precision because they contain entities with different types in the dataset: class 1, annotated 'Presentation', corresponds to three types with the same properties in the dataset: 'Presentation', 'Tutorial' and 'ProgrammeCommitteeMember'.

The results for the DBpedia dataset (see Fig. 1 (b)) show that the assignment of types to entities has achieved good precision and recall. The recall for types 'Book' and 'Politician' is not maximum because noisy instances were detected. Entities of the two types 'Politician' and 'SoccerPlayer' have not been grouped together despite having similar property sets, as it is shown by the corresponding type profiles generated by our algorithm, and presented below.

- Politician: $<$ ($\vec{name}$, 1), ($\vec{party}$, 0.73), ($\vec{children}$, 0.21), ($\vec{birthDate}$, 0.94), ($\vec{nationality}$, 0.15), ($\overleftarrow{successor}$, 0.78), ($\vec{deathDate}$, 0.68), ...$>$.
- SoccerPlayer: $<$ ($\vec{name}$, 1), ($\vec{height}$, 0.46), ($\vec{birthDate}$, 1), ($\vec{nationalteam}$, 0.86), ($\overleftarrow{currentMember}$, 0.8), ($\vec{surname}$, 0.93), ($\vec{deathDate}$, 0.06), ...$>$.

5 Future Works

In addition to type discovery, it is also useful to find the semantic links between them and the labels which best capture the semantics of this cluster. One of our perspectives is to tackle this issue, and provide a support for meaningful cluster annotation.

References

1. Christodoulou, K., Paton, N.W., Fernandes, A.A.: Structure inference for linked data sources using clustering. In: EDBT/ICDT (2013)
2. Ester, M., Kriegel, H.-P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: Kdd (1996)
3. Gangemi, A., Nuzzolese, A.G., Presutti, V., Draicchio, F., Musetti, A., Ciancarini, P.: Automatic typing of DBpedia entities. In: Cudré-Mauroux, P., Heflin, J., Sirin, E., Tudorache, T., Euzenat, J., Hauswirth, M., Parreira, J.X., Hendler, J., Schreiber, G., Bernstein, A., Blomqvist, E. (eds.) ISWC 2012, Part I. LNCS, vol. 7649, pp. 65–81. Springer, Heidelberg (2012)
4. Kellou-Menouer, K., Kedad, Z.: A clustering based approach for type discovery in RDF data sources. Revue des Nouvelles Technologies de l'Information, EGC (2015)
5. Kellou-Menouer, K., Kedad, Z.: Using clustering for type discovery in the semantic web. Fouille de Donnees Complexes (2015, to appear)
6. Nestorov, S., Abiteboul, S., Motwani, R.: Extracting schema from semistructured data. In: ACM SIGMOD Record (1998)
7. Nuzzolese, A.G., Gangemi, A., Presutti, V., Ciancarini, P.: Type inference through the analysis of Wikipedia links. In: LDOW (2012)

8. Paulheim, H., Bizer, C.: Type inference on noisy RDF data. In: Alani, H., Kagal, L., Fokoue, A., Groth, P., Biemann, C., Parreira, J.X., Aroyo, L., Noy, N., Welty, C., Janowicz, K. (eds.) ISWC 2013, Part I. LNCS, vol. 8218, pp. 510–525. Springer, Heidelberg (2013)
9. Völker, J., Niepert, M.: Statistical schema induction. In: Antoniou, G., Grobelnik, M., Simperl, E., Parsia, B., Plexousakis, D., De Leenheer, P., Pan, J. (eds.) ESWC 2011, Part I. LNCS, vol. 6643, pp. 124–138. Springer, Heidelberg (2011)
10. Wang, Q.Y., Yu, J.X., Wong, K.-F.: Approximate graph schema extraction for semi-structured data. In: Zaniolo, C., Grust, T., Scholl, M.H., Lockemann, P.C. (eds.) EDBT 2000. LNCS, vol. 1777, pp. 302–316. Springer, Heidelberg (2000)