Uniqueness, Density, and Keyness: Exploring Class Hierarchies

Anja Jentzsch[1], Hannes Mühleisen[2], and Felix Naumann[1]

[1] Hasso Plattner Institute (HPI), Potsdam, Germany
{anja.jentzsch, felix.naumann}@hpi.de
[2] Centrum Wiskunde & Informatica (CWI), Amsterdam, Netherlands hannes@cwi.nl

Abstract. The Web of Data contains a large number of openly-available datasets covering a wide variety of topics. In order to benefit from this massive amount of open data, e.g., to add value to an organization's internal data, such external datasets must be analyzed and understood already at the basic level of data types, uniqueness, constraints, value patterns, etc.
For Linked Datasets and other Web data such meta information is currently quite limited or not available at all. Data profiling techniques are needed to compute respective statistics and meta information. Analyzing datasets along the vocabulary-defined taxonomic hierarchies yields further insights, such as the data distribution at different hierarchy levels, or possible mappings betweens vocabularies or datasets. In particular, key candidates for entities are difficult to find in light of the sparsity of property values on the Web of Data. To this end we introduce the concept of *keyness* and perform a comprehensive analysis of its expressiveness on multiple datasets.

1 Profiling Linked Data

Over the past years an increasingly large number of data sources has been published as part of the Web of Data[3]. At the time of writing the Web of Data comprised already roughly 1,000 datasets totaling more than 84 billion triples[4], including prominent examples, such as DBpedia [10], and LinkedGeoData [17]. Furthermore, more than 17 billion triples are available as RDFa, Microdata, and Microformats in HTML pages [13]. With the growing number of Linked Datasets on the Web of Data its heterogeneity increases. Simultaneously the number of existing schemas as well as the content and granularity of datasets increases and thus also diverges. This trend makes it increasingly difficult to find and understand relevant datasets, for instance for integration.

A Linked Dataset is represented in the Resource Description Framework (RDF). In comparison to other data models, e.g., the relational model, RDF often lacks explicit schema information that precisely defines the types of entities

[3] The Linked Open Data Cloud visualizes this trend: http://lod-cloud.net
[4] http://datahub.io/dataset?tags=lod

and their properties. Furthermore, Linked Datasets are often inconsistent and lack even basic metadata. One of the main reasons for this problem is that many of the data sources, such as DBpedia or YAGO, have been extracted from unstructured datasets and their schemata usually evolve over time. Hence it is vital to thoroughly examine and understand each dataset, its structure, and its properties before usage. Algorithms and tools are needed that profile the dataset to retrieve relevant and interesting metadata analyzing the entire dataset [14].

We focus on a specific metadata aspect, namely the identification of objects with *keys*. Keys are important in many aspects of data management, such as guiding query formulation, query optimization, indexing, etc. Furthermore, for Linked Datasets key properties allow, e.g., for defining interlinking specification rules to establish the underlying links as owl:sameAs that make the Web of Data a linked one.

In OWL 2 a collection of properties can be assigned as a key to a class using the owl:hasKey statement [6]. This means that each named instance of the class is uniquely identified by the set of values. While in OWL 2 key properties are not required to be functional or total properties, it is always possible to separately state that a key property is functional, if desired. Though OWL allows the definition of key properties, it has not yet fully arrived on the Web of Data. Glimm et al. found that in 2012 only one dataset uses owl:hasKey, while properties like owl:sameAs are already widely used on the Web of Data [5]. Thus, actually analyzing and profiling Linked Datasets to find key candidates requires manual, time-consuming inspection or the help of tools.

Specifying or finding keys in RDF data has another dimension that distinguishes it from relational data: Ontology classes usually are arranged in a taxonomic (subclass/superclass) hierarchy. The class owl:Thing is a superclass of all OWL classes and thus, classes without an explicit superclass are direct subclasses of it. While the Web of Data spans a global distributed data graph, its ontology classes build a tree with owl:Thing as its root. Analyzing datasets along the vocabulary-defined taxonomic hierarchies yields further insights, such as the data distribution at different hierarchy levels, or possible mappings between vocabularies or datasets.

Given a Linked Dataset with a set of properties, a *unique property combination* is a set of one or more properties whose projection has only unique entities. Our initial approach to efficiently detect keys in a given dataset regards all property combinations to find those that together uniquely (and reliably) identify an object. Unfortunately, due to poor specification and data sparsity, such unique property combinations are rare. This insight leads us to the definition of three relaxed dimensions of an RDF property:

- *Uniqueness* is the degree to which the values of a property are unique.
- *Density* is the degree to which a property is filled with values.
- *Keyness* combines the two former dimensions: Properties that are highly unique and highly dense are good key candidates.

Each of these dimensions is determined at all levels of a class hierarchy, leading to insights about which properties best distinguish objects of a class and its sub-

classes. We evaluate the usefulness of our approach by showing various insights we gained analyzing the DBpedia and LinkedGeoData datasets.

2 Related Work

A plethora of tools for profiling Linked Datasets and gathering comprehensive statistics, most tools focus on a specific profiling task. One example for the area of schema induction is ExpLOD [8], which creates summaries for RDF graphs based on class and property usage as well as statistics on the interlinking between datasets based on owl:sameAs links. Li describes a tool that can deduce the actual schema of an RDF dataset [11]. It gathers schema-relevant statistics like cardinalities for class and property usage, and presents the induced schema in a UML-based visualization. In [9] the authors present RDFStats, which uses a SPARQL query processor to collect statistics on Linked Datasets and thus optimize queries. These statistics include histograms for subjects (URIs, blank nodes) and histograms for properties and associated ranges. In [7] authors calculate certain statistical information for the purpose of observing the dynamic changes in datasets.

Others have worked more generally on generating statistics that describe datasets on the Web of Data and thereby help understanding them. LODStats computes statistical information for datasets from the Data Hub [3]. It calculates 32 simple statistical criteria, e.g., cardinalities for different schema elements and types of literal values (e.g., languages, value data types). In [4] the authors automatically create VoID descriptions for large datasets using MapReduce. Aether [12] generates VoID statistical descriptions of RDF datasets. It also provides a Web interface to view and compare them. Roomba [2] generates and validates descriptive Linked Dataset profiles. Finally, ProLOD++ is a web-based tool for profiling and mining Linked Datasets [1]. It comprises various traditional data profiling tasks, adapted to the RDF data model. In addition, it features many specific profiling results for open data, such as schema discovery for user-generated attributes, or association rule mining to uncover synonymous properties.

While there are some RDF profiling tools already available, few tackle key discovery. KD2R allows the automatic discovery of composite key constraints in RDF datasets by deriving maximal non-keys and from these minimal keys [15]. Symeonidou et al. introduce SAKey, which extends KD2R to find "almost keys" [18], i.e., sets of properties that are not quite a key due to few exceptions. The set of almost keys is derived from the set of non-keys found in the data. Both approaches take into account that Linked Data can be erroneous or contain duplicate data but omit the missing density. ROCKER [16] uses a refinement operator that is based on key monotonicity, finds candidate sets of key properties, and assigns them with a discriminability score.

All these existing approaches do not deliver key candidates for each and every dataset. This is where our approach is superior as it calculates the keyness for all properties.

3 Uniqueness, Density, and Keyness of Data

As Linked Datasets are usually sparsely populated, minimal unique property combinations (key candidates) often consist of either multiple low-density properties or cannot be found at all. Novel property attributes, such as the uniqueness, density, and keyness of a property are needed to discover the set of properties that likely identifies an entity, the key candidates. Furthermore, since ontologies are topically clustered by their underlying ontologies, these attributes can be determined per cluster and give some detailed insights into the properties that serve as key candidates per topic.

3.1 Statistics for class hierarchies

A Linked Dataset's class hierarchy is the taxonomy defined by its ontology and therein the rdfs:subClassOf relations between the classes. A cluster C_c for a class c consists of all the entities e that are of rdf:type c, which includes all subclasses of c.

$$C_c = \{e | e \xrightarrow{rdf:type} c\}$$

Clusters can contain entities e that are not in any of its subclusters d. We cluster these entities separately and call the resulting clusters *unspecialized clusters*, denoted as C'_c.

$$C'_c = C_c \setminus \{e \,|\, e \xrightarrow{rdf:type} d,\, d \xrightarrow{rdfs:subClassOf} c\}$$

We omit the c subscript where it is irrelevant in the context. As an additional complication, properties on the Web of Data can have multiple property values. E.g., in the DBpedia dataset we find the following four values for the property dbpedia:birthPlace for the entity of Albert Einstein:

dbpedia:Albert_Einstein dbpedia:birthPlace dbpedia:Ulm,
dbpedia:Kingdom_of_Wuerttemberg,
dbpedia:German_Empire
dbpedia:Baden-Wuerttemberg .

We denote the set of property values of an entity e and property p as $V(e, p)$. To count the number of entities in a cluster C that have at least one value for p, we define $V(C, p) = \{e \,|\, |V(e, p)| > 0, e \in C\}$. Property values of a property p and two entities $e1$ and $e2$ are equal if $V(e1, p) = V(e2, p)$, i.e., if the two sets are identical. With this definition we further define the *set of unique value sets* as $V_{uq}(C, p) = \{V(e, p) \,|\, e \in C\}$.

We are now ready to define the three attributes, uniqueness, density, and keyness, of a property. The *uniqueness* uq of a property p for a cluster C is the number of unique value sets $V_{uq}(C, p)$ per number of total value sets $V(C, p)$ for the given property.

$$\textbf{Uniqueness:} \qquad uq(C, p) = \frac{|V_{uq}(C, p)|}{|V(C, p)|} \tag{1}$$

The *density* d of a property p for a cluster C is the ratio of entities in C that have p to the overall number of entities in C.

$$\textbf{Density:} \qquad d(C,p) = \frac{|V(C,p)|}{|C|} \tag{2}$$

We call a property *full key candidate* if its density and uniqueness are both 1. For cases where they are not both 1 we define its keyness as a useful attribute. The *keyness* k of a property p for a cluster C is the harmonic mean of its uniqueness and density. The harmonic mean emphasizes that *both* parameters must be high to achieve an overall high keyness:

$$\textbf{Keyness:} \qquad k(C,p) = \frac{2 \cdot uq(C,p) \cdot d(C,p)}{uq(C,p) + d(C,p)} \tag{3}$$

We call a property *key candidate* if its keyness is above some threshold.

We investigate the three attributes of an RDF property, uniqueness, density, and keyness, for the given cluster types C, and C'. Determining uniqueness, density, and keyness for a property p in a cluster C_c requires analyzing *all* property value sets for *all* entities in the given cluster. We observe all kinds of specificities of properties for clusters and their subclusters that allow for a fine-grained, cluster-based retrieval of key candidates.

4 Evaluation

Our approach has been implemented in ProLOD++, a web-based tool for profiling and mining Linked Datasets [1][5]. It comprises various traditional data profiling tasks, adopted to the RDF data model. In addition, it features many specific profiling results for Linked Datasets, such as schema discovery for user-generated attributes, association rule discovery to uncover synonymous properties, and, in particular, key discovery along ontology hierarchies. It allows to navigate a Linked Dataset via an automatically computed topical and hierarchical clustering as well as along its ontology class tree. The latter allows the user to observe the evolution of key features along hierarchies, thus determining class-specific properties and key candidates.

In combination, having these property attributes at hand, the user can better decide on which properties serve as keys, especially on class level and gain further insight into the completeness and structure of the dataset at hand.

4.1 Datasets

We evaluated our approach using two datasets, DBpedia v.3.9 [10][6] and LinkedGeoData [17]. DBpedia is a Linked Data version of Wikipedia and thus a cross-domain dataset. It has evolved into a hub on the Web of Data with many other

[5] A ProLOD++ demo is available at `http://prolod.org`

[6] Our evaluation uses the English DBpedia, excluding the raw infobox properties.

datasets linking to it. LinkedGeoData is a spatial dataset that publishes OpenStreetMap data as Linked Data. It is one of the bigger and constantly evolving datasets on the Web of Data.

For DBpedia we analyzed the Person cluster and several subclusters including Athlete and its subclusters, and Scientist (see Table 1). We deliberately omit the artificial class Agent, subclass of owl:Thing and superclass of dbpedia:Person, due to its main function to define properties that are also needed for the Organization and Family class on the same class hierarchy level as Person.

Table 1 lists some subclasses of the DBpedia Person class along with the number of entities and number of properties in the respective cluster. 35% of the entities in DBpedia are persons due to Wikipedia mainly covering persons, places, and sports topics. The Person cluster is a diverse one with 255 properties being used, half of them (127) also occurring on the Athlete subclass. The Athlete class has several subclasses from which we chose representative ones. Furthermore, we included the Scientist class as a comparison in our evaluation, which has no further subclasses and only 40 properties.

Class	**# Entities**	**# Properties**
owl:Thing	3,221,405	1,376
Person	831,558	255
Person'	110,726	56
Athlete	185,081	127
Athlete'	16,224	18
AmericanFootballPlayer	11,884	44
BaseballPlayer	19,807	44
BasketballPlayer	6,487	33
Cyclist	3,828	18
IceHockeyPlayer	11,535	31
RugbyPlayer	11,098	32
SoccerPlayer	89,078	37
Scientist	14,894	40

Table 1. Number of entities and properties for some classes of the DBpedia Person class hierarchy.

This analysis shows that having some basic profiling results on property usage at hand can already be useful. While there are 2,333 properties defined in the DBpedia 3.9 ontology, only 1,376 are used by at least one entity. When browsing through Linked Dataset class hierarchies, the information on which properties are actually being used, compared to the defined ones, can help to narrow down the properties of interest for tasks like key discovery.

For LinkedGeoData we analyzed the 2013 version and therein the Amenity cluster and the subclusters shown in Table 2. Due to its automatic conversion from OpenStreetMaps, which is publicly curated by a large number of people, the number of properties to describe amenities is very high.

Class	# Entities	# Properties
owl:Thing	49,355,161	16,278
Amenity	6,824,892	12,371
Amenity'	5,543,014	10,467
Shop	1,130,204	3,790
Shop'	1,008,344	3,530
Butcher	20,432	294
Bakery	57,204	544
IceCream	2,643	140

Table 2. Number of entities and properties for some classes of the LinkedGeoData Shop class hierarchy.

4.2 Density, uniqueness, and keyness

Figure 1 plots the uniqueness and density for all properties in the DBpedia Person, Athlete, and SoccerPlayer cluster, highlighting selected properties. Overall, the property densities are noticeably low, while the uniqueness is distributed over the entire x-axis. The DBpedia Person cluster has only two properties (foaf:name and rdfs:label) with a high uniqueness (above 0.9) and density (roughly 0.7). For the subclasses there are a few more properties with a high density and uniqueness. We can already see that there is different behaviour in property uniqueness and density along the class hierarchy. While the uniqueness for dbpedia:birthPlace stays approximately equal for Person, Athlete, and SoccerPlayer, dbpedia:team becomes more unique for Athlete than for Person and even more unique for SoccerPlayer. The density of both properties increases.

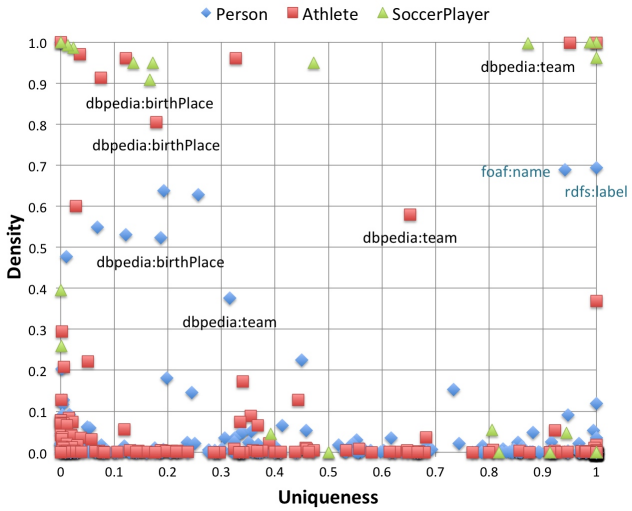

Fig. 1. Uniqueness and density for properties of the DBpedia Person, Athlete, and SoccerPlayer clusters.

Ontology Property	Uniq.	Density	Keyness
foaf:name	0.94	0.69	0.80
rdfs:label	1.00	0.69	0.59
foaf:surname	0.26	0.63	0.36
dbpedia:team	0.32	0.38	0.34
dbpedia:currentMember	0.45	0.23	0.30
foaf:givenName	0.19	0.64	0.30
dbpedia:birthPlace	0.19	0.52	0.28
dbpedia:occupation	0.73	0.15	0.25
dbpedia:careerStation	1.00	0.12	0.21
...			
rdf:type	0.00	1.00	0.00
...			
dbpedia:pseudonym	0.99	0.00	0.00
...			
dbpedia:espnId	1.00	0.00	0.00

Table 3. Uniqueness, density, and keyness for the DBpedia Person cluster.

Table 3 shows some of the 255 properties in the Person cluster along with their uniqueness, density, and keyness, ordered by keyness. What can already be observed from this selection are some typical characteristics of Linked Datasets. They often contain properties with only few property values but a high uniqueness (nearly 1), e.g., dbpedia:pseudonym. Many properties have a high uniqueness but their density is not 1, e.g., foaf:name, and rdfs:label. The density rarely reaches 0.5 (for only eight properties) and only in one of 255 properties (rdf:type) reaches 1. 86 % of the properties have only up to 5 % values available.

For example out of the 185,081 athletes in DBpedia, only 36 have a dbpedia:espnId value, yet all of these values are unique. This would identify dbpedia:espnId as a key candidate for athletes using traditional key discovery approaches. This observation emphasizes the need to take into account further details like the keyness of a property when choosing key candidates.

Figure 2 shows the uniqueness and density for all properties in the LinkedGeoData Amenity, Shop, and Bakery clusters, again highlighting selected properties. wgs84:lat is the latitude of an amenity's position. Uniqueness and density stay approximately equal for the classes along the hierarchy. The same can be observed for the label rdfs:label of entities in LinkedGeoData. Positions and labels are the two most used properties in LinkedGeoData. The overall property density is again low, which is due to the enormous mostly manual effort to add metadata for all the 49,355,161 entities in OpenStreetMaps/LinkedGeoData.

Even more than DBpedia, LinkedGeoData is sparsely populated with property values. Table 4 shows some of the 12,371 properties in the Amenity cluster along with their uniqueness, density, and keyness, ordered by keyness. Only five properties have a keyness above 0.8 and as we have already observed in Figure 2, the overall property density is low. Only ten properties have a density above 0.5, and two of them, dcterms:modified and lgd:changeset, are metadata

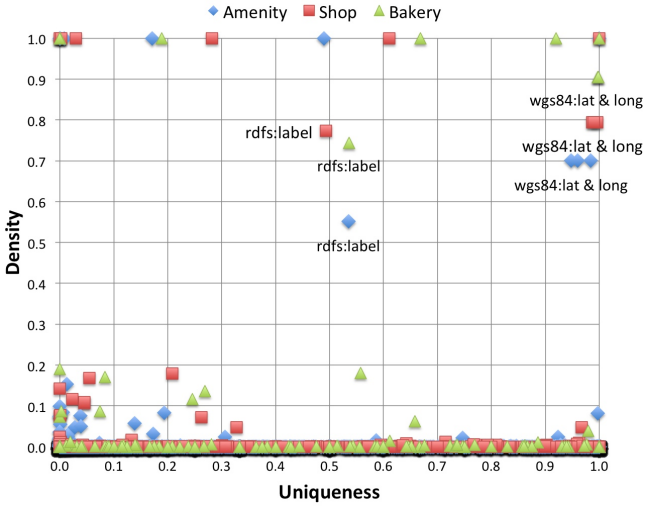

Fig. 2. Uniqueness and density for properties of the LinkedGeoData Amenity, Shop, and Bakery clusters.

Ontology Property	Uniq.	Density	Keyness
geovocab:geometry	1.00	1.00	1.00
owl:sameAs	0.98	0.70	0.82
wgs84:long	0.96	0.70	0.81
wgs84:lat	0.95	0.70	0.81
dcterms:modified	0.49	1.00	0.66
rdfs:label	0.53	0.55	0.54
lgd:changeset	0.17	1.00	0.29
lgd:gnis:feature_id	1.00	0.08	0.15
lgd:addr:street	0.19	0.08	0.12
lgd:operator	0.14	0.06	0.08
...			
rdf:type	0.00	1.00	0.00
...			
lgd:housenumber	1.00	0.00	0.00

Table 4. Uniqueness, density, and keyness for the LinkedGeoData Amenity cluster.

properties. The average property uniqueness is 0.64, the average property density and keyness are 0.00. These numbers reflect the enormous number of places in LinkedGeoData and the according effort to add metadata for all of them.

4.3 Class hierarchies

Furthermore, we evaluated the uniqueness, density, and keyness of selected properties *along the class hierarchy*. Table 5 shows the uniqueness, density, and keyness for the properties dbpedia:birthDate and dbpedia:team along the Person class hierarchy. As already observed in Figure 1, the keyness for dbpedia:team increases for Athlete and subclasses of Athlete as it is specific to the sports theme but not existent for the Scientist cluster at all. The dbpedia:birthDate property has a high

density for all persons but its uniqueness is naturally not very high. The coverage of athletes' birth dates starts only in the 17th century: the oldest athlete on DBpedia is a cricketer called William Bedle, born 1679.

In the Person' and Athlete' clusters only few properties are used, which explains the missing dbpedia:team property. Generally, these clusters contain entities that could not be further classified. When identifying key candidates, we observe that the keyness for `dbpedia:birthDate` is an outlier compared with the main clusters. Thus the keyness of Person and Athlete are of higher confidence. Table 6 in the appendix shows the analogous analysis for selected LinkedGeoData properties.

To summarize our findings, we identified three types of property keyness along the class hierarchy:

- *Less specific*, i.e., keyness decreases per class level in the class hierarchy. An example is dbpedia:deathPlace whose keyness for Person is 0.18, for Athlete 0.14, and for SoccerPlayer 0.08.
- *Generic*, i.e., keyness stays approximately equal throughout the class hierarchy. An example is dbpedia:birthPlace, whose keyness for Person is 0.28, for Athlete 0.29, and for SoccerPlayer 0.28.
- *More specific*, i.e., keyness increases per class level in the class hierarchy. An example is dbpedia:team which keyness for Person is 0.34, for Athlete 0.61, and for SoccerPlayer 0.93.

Figures 1 and 2 already depict these types of property keyness. In DBpedia we can find all three types of property keyness: the keyness for dbpedia:birthPlace stays approximately equal for Person, Athlete, and SoccerPlayer *(generic)*. For dbpedia:team the keyness increases significantly from Person to Athlete and finally to SoccerPlayer *(more specific)*. The keyness for dbpedia:deathPlace decreases

Ontology Class	**Entities**	**dbpedia:birthDate**			**dbpedia:team**		
		Uniq.	**Dens.**	**Keyn.**	**Uniq.**	**Dens.**	**Keyn.**
owl:Thing	3,221,405	0.05	0.21	0.08	0.12	0.28	0.17
Person	831,558	0.07	0.55	0.12	0.32	0.38	0.34
Person'	110,726	0.28	0.75	0.41	—	—	—
Athlete	232,082	0.07	0.91	0.14	0.65	0.25	0.61
Athlete'	16,224	0.56	0.90	0.69	—	—	—
Am.FootballPlayer	11,884	0.55	0.98	0.70	0.22	0.70	0.34
BaseballPlayer	19,807	0.67	0.93	0.78	0.31	0.94	0.46
BasketballPlayer	6,487	0.69	0.98	0.81	0.11	0.41	0.17
Cyclist	3,828	0.83	0.98	0.90	0.59	0.04	0.08
IceHockeyPlayer	11,535	0.55	0.96	0.70	0.04	0.54	0.08
RugbyPlayer	11,098	0.63	0.71	0.67	0.11	0.05	0.07
SoccerPlayer	89,078	0.14	0.95	0.24	0.87	1.00	0.93
Scientist	14,894	0.84	0.74	0.79	—	—	—

Table 5. Uniqueness, density, and keyness for DBpedia properties dbpedia:team and dbpedia:birthDate along exemplary classes of the Person class hierarchy.

along the class hierarchy to SoccerPlayer *(less specific)*. This can be explained with the fact that DBpedia is an ever-evolving knowledge base and at the time of writing most death places covered for soccer players were in a certain town in England, namely Stoke-on-Trent. In LinkedGeoData the properties' keyness is mostly *generic*, like rdfs:label and wgs84:lat, but also sometimes gets *more specific* on the lower class hierarchy levels.

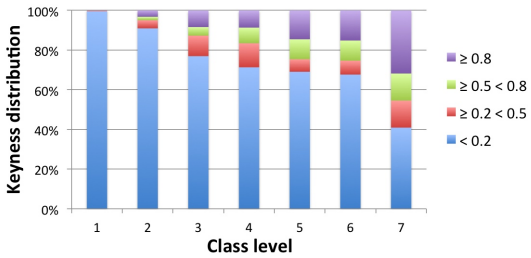

Fig. 3. Keyness distribution for properties along the DBpedia class hierarchy.

Finally, Figure 3 shows the keyness distribution for all properties along the DBpedia class hierarchy. Class level 1 contains all properties of owl:Thing, level 2 all properties of subclasses of owl:Thing and so forth. Overall, the keyness is increasing per class level. For the root class level 1 out of 1,376 properties only one has a keyness higher than 0.8, for class level 7 this increases to 7 out of 22 properties. This observation leads to the conclusion that class level-based keys are a better choice than high-level keys. Key features for athletes might be too high-level for soccer players and should be redefined on that level.

Our evaluation shows that the property keyness can help discovering key candidates for Linked Datasets. It also highlights the advantages of analyzing the class hierarchy in order to observe property behaviour for classes along it and make better choices when identifying key candidates for specific classes.

5 Conclusion and Future Work

We have introduced the concept of keyness (and therein uniqueness and density) of a property to address the sparsity on the Web of Data and thus create the possibility to find key candidates where traditional approaches fail.

Our approach has been implemented in ProLOD++ and provides users with the uniqueness, density, and keyness for all properties. Having these profiling results at hand helps users in finding key candidates and analyzing the relevance of properties along class hierarchies in Linked Datasets.

While the keyness of a property is already useful for discovering key candidates, the keyness values rarely reach 0.8 due to dataset sparsity. Thus we plan to extend the keyness concept to *sets* of properties. Because minimal unique property combinations rely only on a high combined density, they are not the ideal approach here. It seems reasonable, for instance, to consider the amount of overlap of properties in combination with the properties' keyness.

References

1. Z. Abedjan, T. Grütze, A. Jentzsch, and F. Naumann. Mining and Profiling RDF Data with ProLOD++. In *Proceedings of the International Conference on Data Engineering (ICDE)*, pages 1198–1201, 2014. Demo.
2. A. Assaf, R. Troncy, and A. Senart. Roomba: An extensible framework to validate and build dataset profiles. In *ESWC International Workshop on Dataset Profiling & Federated Search for Linked Data (PROFILES)*, 2015.
3. S. Auer, J. Demter, M. Martin, and J. Lehmann. LODStats – an extensible framework for high-performance dataset analytics. In *Proceedings of the Int. Conf. on Knowledge Engineering and Knowledge Management (EKAW)*, 2012.
4. C. Böhm, J. Lorey, and F. Naumann. Creating VoiD descriptions for web-scale data. *Journal of Web Semantics*, 9(3):339–345, 2011.
5. B. Glimm, A. Hogan, M. Krötzsch, and A. Polleres. OWL: Yet to arrive on the web of data? In *WWW Workshop on Linked Data on the Web (LDOW)*, 2012.
6. P. Hitzler, M. Krötzsch, B. Parsia, P. F. Patel-Schneider, and S. Rudolph, editors. *OWL 2 Web Ontology Language: Primer*. W3C Recommendation, 2009.
7. T. Käfer, A. Abdelrahman, J. Umbrich, P. O'Byrne, and A. Hogan. Observing linked data dynamics. In *Proceedings of the Extended Semantic Web Conference (ESWC)*, volume 7882 of *LNCS*, pages 213–227. Springer, 2013.
8. S. Khatchadourian and M. P. Consens. ExpLOD: Summary-based exploration of interlinking and RDF usage in the linked open data cloud. In *Proceedings of the Extended Semantic Web Conference (ESWC)*, pages 272–287, 2010.
9. A. Langegger and W. Wöß. RDFStats – an extensible RDF statistics generator and library. In *Proceedings of the International Workshop on Database and Expert Systems Applications (DEXA)*, pages 79–83, 2009.
10. J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer. DBpedia – a large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web Journal*, 6(2):167–195, 2015.
11. H. Li. Data Profiling for Semantic Web Data. In *Proceedings of the International Conference on Web Information Systems and Mining (WISM)*, pages 472–479, 2012.
12. E. Mäkelä. Aether – generating and viewing extended VoID statistical descriptions of RDF datasets. In *ESWC (Satellite Events)*, pages 429–433, 2014.
13. R. Meusel, P. Petrovski, and C. Bizer. The webdatacommons microdata, rdfa and microformat dataset series. In *Proceedings of the International Semantic Web Conference (ISWC)*, pages 277–292, 2014.
14. F. Naumann. Data profiling revisited. *SIGMOD Record*, 42(4):40–49, 2013.
15. N. Pernelle, F. Saïs, and D. Symeonidou. An automatic key discovery approach for data linking. *Web Semantics: Science, Services and Agents on the World Wide Web*, 23:16–30, 2013.
16. T. Soru, E. Marx, and A.-C. Ngonga Ngomo. ROCKER – a refinement operator for key discovery. In *Proceedings of the International World Wide Web Conference (WWW)*, pages 1025–1033, 2015.
17. C. Stadler, J. Lehmann, K. Höffner, and S. Auer. LinkedGeoData: A core for a web of spatial open data. *Semantic Web Journal*, 3(4):333–354, 2012.
18. D. Symeonidou, V. Armant, N. Pernelle, and F. Saïs. SAKey: Scalable almost key discovery in RDF data. In *Proceedings of the International Semantic Web Conference (ISWC)*, pages 33–49, 2014.

A Keyness analysis for selected LinkedGeoData properties

In analogy to Table 5 for persons, Table 6 shows the uniqueness, density, and keyness for the selected LinkedGeoData properties wgs84:lat and rdfs:label along the Amenity class hierarchy. For the position properties it is noticeable that the keyness stays high (above 0.79) for all the classes along the hierarchy. The fewer the entities in the class cluster (butchers and ice cream shops), the higher the property keyness. For the label the keyness also is quite stable along the class hierarchy (around 0.6) but is especially high in bakeries due to the uniqueness in labels and a high density.

Ontology Class		**wgs84:lat**			**rdfs:label**		
	Entities	**Uniq.**	**Dens.**	**Keyn.**	**Uniq.**	**Dens.**	**Keyn.**
owl:Thing	49,355,161	0.93	0.16	0.27	0.53	0.14	0.22
Amenity	6,824,892	0.95	0.70	0.81	0.53	0.55	0.54
Amenity'	5,543,014	0.95	0.68	0.79	0.55	0.51	0.53
Shop	1,130,204	0.99	0.79	0.88	0.49	0.77	0.60
Shop'	1,008,344	0.99	0.78	0.87	0.79	0.61	
Butcher	20,432	1.00	0.89	0.94	0.71	0.73	0.72
Bakery	57,204	0.54	0.74	0.62	1.00	0.90	0.95
IceCream	2,643	1.00	0.89	0.94	0.70	0.78	0.74

Table 6. Uniqueness, density, and keyness for LinkedGeoData properties wgs84:lat and rdfs:label along exemplary classes of the Amenity class hierarchy.