Statistical Knowledge Patterns for Characterising Linked Data

Eva Blomqvist[1], Ziqi Zhang[2], Anna Lisa Gentile[2], Isabelle Augenstein[2], and Fabio Ciravegna[2]

[1] Department of Computer and Information Science, Linköping University, Sweden
[2] Department of Computer Science, University of Sheffield, UK
eva.blomqvist@liu.se,
{z.zhang,a.l.gentile,i.augenstein,f.ciravegna}@dcs.shef.ac.uk

Abstract. Knowledge Patterns (KPs), and even more specifically Ontology Design Patterns (ODPs), are no longer only generated in a top-down fashion, rather patterns are being extracted in a bottom-up fashion from online ontologies and data sources, such as Linked Data. These KPs can assist in tasks such as making sense of datasets and formulating queries over data, including performing query expansion to manage the diversity of properties used in datasets. This paper presents an extraction method for generating what we call Statistical Knowledge Patterns (SKPs) from Linked Data. SKPs describe and characterise classes from any reference ontology, by presenting their most frequent properties and property characteristics, all based on analysis of the underlying data. SKPs are stored as small OWL ontologies but can be continuously updated in a completely automated fashion. In the paper we exemplify this method by applying it to the classes of the DBpedia ontology, and in particular we evaluate our method for extracting range axioms from data. Results show that by setting appropriate thresholds, SKPs can be generated that cover (i.e. allow us to query, using the properties of the SKP) over 94% of the triples about individuals of that class, while only needing to care about 27% of the total number of distinct properties that are used in the data.

1 Introduction

Originally, the notion of Ontology Design Patterns (ODPs) referred only to a top-down view on modelling best practices, and constituted manually designed patterns representing those best practices. More recently, however, Knowledge Patterns (KPs), as a generalisation of ODPs and other patterns, have also been created in a bottom-up fashion, i.e., representing the way information on the Web or Linked Data is actually represented, rather than how it "should" be represented according to some best practice. This paper follows the more recent tradition and presents what we call Statistical Knowledge Patterns (SKPs), which aim to characterise concepts that exist within Linked Data based on a statistical analysis of those data. Since the SKPs are wholly based on the characteristics of data itself, their construction is a completely automatic process, which means that they can be kept up-to-date with respect to data without any manual effort.

In a related paper [15] we have presented the details of the initial steps of the SKP generation method, with specific focus on discovering relations that are (to some extent) synonymous, and evaluating that part of the extraction in the context of query expansion. In this paper we instead focus on the pattern extraction method as a whole, and the resulting resource, i.e., the pattern catalogue, and in particular discuss the parts of the method not covered by the previous paper. In Section 2 we first present some related work on ODP generation from different sources. We then briefly present our SKP extraction method in Section 3, and exemplify the resulting SKPs in Section 4. In Section 5 we show through some empirical findings that the SKPs fulfill their purpose, i.e., characterise and provide access to the underlying data, but in particular we study and evaluate the range extraction method. Finally, in Section 6 we discuss some general implications of this research, and in Section 7 we provide conclusions and outline future work.

2 Related Work

Ontology Design Patterns (ODPs) were originally conceived for the task of ontology engineering, and in particular were intended to encode general best practices and modelling principles in a top-down fashion [5,6]. Since then several kinds of patterns have been proposed, such as *Content Ontology Design Patterns* (CPs) [7]. CPs focus on domain-specific modelling problems and can be represented as small, reusable pieces of ontologies. CPs are similar to the SKPs presented in this paper, in the way that they also represent concepts with their most distinguishing characteristics. Unlike SKPs however, CPs are usually created manually, and since they are abstract patterns intended for being used as “templates” in ontology engineering they usually lack any direct connection to data and cannot directly (without manual specialisation) be used for querying Linked Data. Since CPs represent an abstract top-down view, they additionally do not consider aspects such as diversity and synonymy among properties, which is one of the benefits that our proposed SKPs display.

The approach closest to our SKPs is the *Encyclopedic Knowledge Patterns* (EKPs) [11], which were intended mainly for use in an exploratory search application [9,10]. The EKP generation process exploits statistics about link-usage from Wikipedia[3] to determine which relations are the most representative for each concept. The assumption is that if instances of a target concept `A` frequently link to instances of concept `B`, then concept `B` is an important concept for describing instances of `A`. This information is then formalised and stored as small OWL ontologies (the EKPs), each having one main class as their focus and all its significantly frequent relations (based on the wiki-link counts) to other classes represented as object properties of the main class. The main purpose of these EKPs is presenting relevant information to a human user, e.g., the ability to filter out irrelevant data when presenting information about DBpedia entities, while the ability to query for data is not a primary concern. This is reflected by the fact that EKPs mainly contain abstractions of relevant properties, such as “linksToClassB”, where `linksToClassB` expresses the fact that the pages in Wikipedia representing instances of concept `A` (the class in focus of the EKP) commonly links to pages in Wikipedia representing instances of concept `B` (links which could in many cases in turn be represented by various DBpedia properties, but not necessarily). This is however not sufficient for our case, since our main goal is to use our SKPs to characterise

[3] `http://en.wikipedia.org`

and give effective access to actual data. In such a use case one needs to be able to distinguish between, for instance, different properties that link instances of the same classes but have different meaning (e.g., birth place and death place, which both link a person to a location). Hence, we propose an extension of the existing EKPs, which also include a sufficient coverage of actual properties of the underlying datasets, together with additional features we attach to each of those properties, such as range axioms.

There exist other approaches aiming to statistically characterise datasets, such as the one by Basse et al. [3], which also exploits statistics from a specific dataset to produce topic frames of that dataset. In contrast to Nuzzolese et al. [11] they do not produce a pattern for each class but rather generate clusters of classes (up to 15 classes each) that reflect main topics of the dataset. For giving access to data (querying), however, the main focus needs to be on the properties of the classes, rather than the classes themselves. Also Atencia et al. [1] perform statistical analyses on datasets, but for the purpose of detecting key properties (i.e., to be expressed through the OWL2 notion of "key") rather than characterising the complete property landscape of a class. A related approach is also the LODStat framework [2], which has the broader scope of extracting an publishing many kinds of interesting statistics about datasets. While that framework also takes into account statistics on property usage, and declaratively represents the statistics, the approach is focused on per-dataset statistics, rather than per-class, and does not induce new information (e.g., synonymity or new range axioms) from the extracted statistics.

Looking at patterns from a more general perspective, however, Knowledge Patterns (KPs) have been defined as general templates or structures used to organise knowledge [8], which can encompass both the "traditional" view of ODPs and more recent effort such as EKPs and our SKPs. In the Semantic Web scenario they are used both for constructing ontologies [4,7,13] and for using and exploring them [3,9,10,11,12]. Presutti et al. [12] explore the challenges of capturing Knowledge Patterns in a scenario where explicit knowledge of datasets is neither sufficient nor straight-forward, which is the case for Linked Data. They propose a dataset analysis approach to capture KPs and support datasets querying. Our SKPs expand on this work as not only do we capture direct statistical information from the underlying datasets, but also further characterise relevant properties with additional features (e.g., synonymous properties and range axioms), which is highly beneficial for querying the datasets.

3 SKP Construction Method

A Statistical Knowledge Pattern (SKP) is an ontological view over a class that summarises the usage of the class (hereafter called the *main class* of the SKP) in data. The main class of an SKP can be seen as the "focus", or the context, of that SKP, hence, each SKP has exactly one main class. The term "statistical" refers to that the pattern is constructed based on statistical measures on data. Each SKP contains: (1) properties and axioms involving the main class that relates it to other classes, derived from a reference ontology or from a pre-existing EKP characterising that class; (2) properties and axioms involving the main class that are not formally expressed in the reference ontology, but which can be induced from statistical measures on statements published as

Linked Data. The information from (1) and (2) is consolidated in the form of an SKP, which is represented and stored as a small OWL ontology.

More formally, let the main class of an SKP be c_{main}, which is a class of the selected reference ontology – in fact, this is the only thing we need from the reference ontology, hence, the ontology can simply consist of one or more class URIs if nothing else is available, as long as there is some data using that class. The main class is the starting point for extracting an SKP, hence, it is selected before the construction process begins, and normally one would build SKPs for as many of the classes in the reference ontology as possible (or for the classes that are of specific importance in some use case). The SKP of c_{main} contains the set of properties from the reference ontology $P_{ont} = \{p_{ont1} \dots p_{ont-n}\}$ and the set of properties from any pre-existing EKP of the main class $P_{ekp} = \{p_{ekp1} \dots p_{ekp-m}\}$, with the requirement that only properties that are actually used in data (or have relations to properties that are actually used in data, see further below, are included). A property from the reference ontology or an EKP, p_i may have a set of "synonymous properties" SP_i induced from data. The decision on synonymity of properties is based on a *synonymity* measure (described in detail in [14]), hence, almost none of the properties are actual synonyms (i.e., with a maximum score) but rather represent properties that are to some extent exchangeable in the particular context of the main class. While we will continue to use the term "synonymous properties" throughout this paper, the reader should bear in mind that these are rarely perfect synonyms, but rather "close matches" (as we shall see later, this is also represented in the resulting model through `skos:closeMatch` rather than equivalence). To decide which properties, or synonym clusters of properties, should be selected to be included in the SKP, their *relevance* is measured based on the frequency of usage of the properties in available Linked Data.

In practice, since SKPs are an extension of EKPs [11], if an EKP already exists it can be used as an abstract frame for the concrete properties and axioms that are added through our SKP generation method. In particular, we use the abstract properties introduced by EKPs (i.e., "links to class X") in order to group properties with range axioms overlapping the general EKP property, to give the SKP a more intuitive structure and improve human understandability of the pattern. The properties are thereby organised in two hierarchical layers, through the `rdfs:subPropertyOf` relation, where, in particular, domain and range restrictions of properties are used to induce sub property relations between the very general properties of a pre-existing EKP and the properties retrieved from data. Note that we are, at this point, not attempting to induce a sub-property structure among the properties found in data, hence, we only group them under the general EKP properties. A more elaborate structuring of the extracted properties is still part of future work.

The most important characteristics of SKPs and their generation are:

– SKPs encode class-specific characterisations of properties that are commonly used with individuals of that class, i.e., synonymous properties, ranges, etc. are all specific to the use of the properties with instances of that class, which provides an interesting and detailed account of property meanings and usage in Linked Data. For example, the same property may be present in several SKPs, but with distinct range axioms, and as part of separate property synonym clusters, depending on that the property is used differently with instances of the respective main class of each SKP.

– Synonymous (i.e., to some extent interchangeable) properties are identified, and information about them are stored to be reused; one possible usage is query expansion, when querying the data underlying the SKP. See [15] for details.
– Ranges are identified for properties that have no range in the reference ontology, hence, showing the actual use of the property in data, which can be used to restrict property selection when building a query or to filter out unwanted data at query-time.
– The method for SKP generation is fully automated, whereby SKPs can be regenerated as soon as data changes, without manual effort, but SKPs are in the meantime used as stored resources, for increased usage efficiency.

The SKP generation process consists of three key components: (1) discovering and grouping synonymous properties of the main class, (2) selecting properties (and groups of properties) to include in the SKP, and (3) collecting additional axioms describing the selected properties, such as `rdfs:subPropertyOf` relations and domain and range restrictions, and creating an ontological representation of the SKP.

Synonymity of Properties To create an SKP we identify the properties used for the SKP main class based on data and measure their synonymity. In [14] we have proposed a novel synonymity measure of properties. The overall process is:

1. Query the dataset for all the instances (IND) of the main class; query the dataset for all triples having any $i \in IND$ in subject position (we denote this triple set TS) and additionally collect the types (through querying for *rdf:type* statements or for a datatype) of the objects of all those triples.
2. For each property used in TS, collect the subset of IND having the property as predicate, IND_{prop}, and collect the corresponding objects of each subject in IND_{prop} – the *subject-object* pairs of this set represents the characteristics of that property, given the main class at hand.
3. Do a pairwise comparison of all subject-object pairs of IND_{prop} for all the properties and calculate a *synonymity* score for each pair of properties.
4. Use the *synonymity* scores (representing evidence of properties being interchangeable) to cluster properties that are likely to represent a sufficiently similar (i.e., sufficiently synonymous) semantic relation.

Selection of Properties The aim of the above process is to discover, for each specific main class, clusters of properties with the same meaning. In practice, a certain number of properties are found to be noise or non-representative of the main class. Thus, we further refine the set of properties for each SKP as follows:

5. Calculate the frequencies of properties used in data, i.e., counting distinct objects in IND_{prop}. For clusters, treat the cluster as if it was a single property hence add the frequency counts of the constituent properties.
6. Use a cutoff threshold T (explored further in [15]) to filter out infrequent properties (or clusters), as they may represent noise in the data. Add those above the threshold to the SKP, including information about their appropriate property type (e.g. `owl:DatatypeProperty` or `owl:ObjectProperty`), with their original namespace intact.
7. For each member of a property cluster that is added to the SKP, add a `skos:closeMatch` relation between the cluster members.

Characterisation of Properties Finally, we add as much information as we can about the selected properties, based on what we can induce from the data, and retrieve from the reference ontology or the pre-existing EKP.

8. For each property, add a range axiom that consists of any range that is given to the property in the reference ontology or the EKP (if present), but if not present instead add any range that is identified in data (i.e., by looking at the frequencies of the object types of the triples above a certain threshold).
9. Add `rdfs:subPropertyOf` axioms for those properties where the ranges match some abstract EKP property (i.e., the "links to class X" abstract properties).
10. Store the SKP as an OWL file.

More in detail the range extraction method starts by inspecting the types of all the triple objects in TS that were retrieved at the beginning of the overall process. This is done on a per-property-basis, i.e., for each property selected for inclusion in the SKP, which does not have a range axiom defined in the reference ontology, the corresponding subject-object pairs are again analysed, and this time inspected together with the types of the objects of those pairs. Assume that the set of distinct objects, for the triples of TS using a property p_i is OBJ_{p_i}. Now, count the frequency of the types of the instances in OBJ_{p_i}, i.e., associating each class (or datatype) $type_j$ that is a type of one of the instances in OBJ_{p_i} with a count value $count_{type_j}$. Then calculate the relative frequency of this type, for the specific property, by dividing $count_{type_j}$ with the total number of distinct objects of that property, i.e., $|OBJ_{p_i}|$. Intuitively, this is a measure of how large fraction of the triple objects in the set of triples characterising this property that "support" this type being in the range of p_i.

For avoiding to include too much noise in the axiomatisation of the SKPs, a threshold is set on this "range support" value, i.e., a class should not be included unless it has sufficient support in the data. Where, "sufficient" may differ depending on if one prioritises precision or recall. We investigate a reasonable trade-off for the relative threshold in Section 5, however, we also set an absolute threshold (for really small triple sets) not to include any type that has less than 10 occurrences in the triple set. Since this process may result in a set of classes being selected as the appropriate range of a property, the range axiom included in the SKP is then expressed as the union of those classes.

4 Results

The resulting patterns have been published[4] in the form of small OWL ontologies. Where pre-existing EKPs exist, they can be extended with new properties, while if no pre-existing EKP existed, the SKP is generated completely from scratch. Overall, an SKP contains the main class that is the focus of the pattern, and the properties that are selected for that SKP, including their domain and range axioms. The name of the SKP is the same as the name of the main class. As an example, we illustrate a small part of the resulting SKP called Language[5] in Figure 1, with the main class `dbpedia:Language`. This is one

[4] SKPs are being made available at `http://www.ontologydesignpatterns.org/skp/`
[5] http://www.ontologydesignpatterns.org/skp/Language.owl

of the smallest SKPs generated in our evaluation set (see Section 5), only including 36 distinct properties, distributed over 35 object properties and 1 datatype property. Each property has kept its original URI, so as to be directly usable for querying data, and is given the main class of the SKP as domain. In this particular SKP we, for instance, find properties such as `dbpedia:spokenIn`, `dbprop:region` and `foaf:name`, i.e., coming from three different namespaces. At a first glance, `foaf:name` may seem to be an error, however, this nicely exemplifies the SKPs ability to reflect actual usage in data. The property was certainly not intended for expressing the name of languages, however, for this particular class the property is actually used in this way and could be useful to include when querying for data about languages. Without seeing the SKP, or experimenting with queries manually, this may be hard to discover.

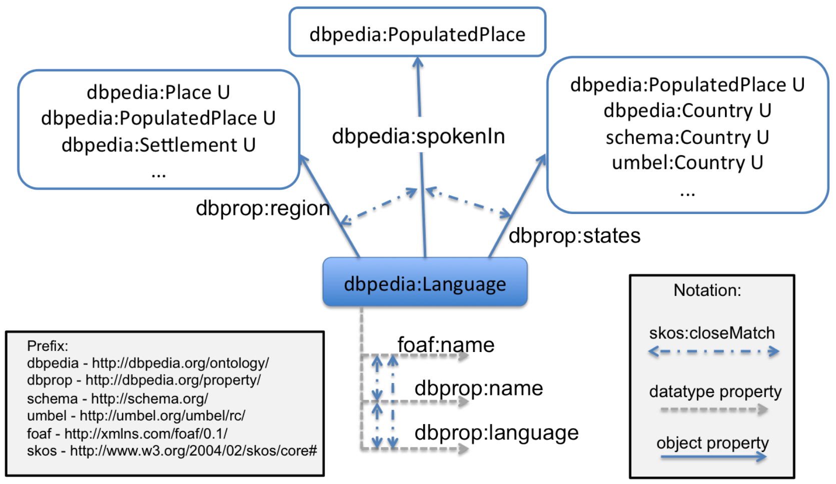

Fig. 1. Illustration of a small part of the Language SKP. Classes are illustrated as boxes, including the union classes representing complex ranges, and properties as arrows. An arrow starting from a class means that is the domain of the property, and the class at the end of the arrow is the range. The `skos:closeMatch`-arrows represent assertions on properties.

The property `foaf:name` is additionally part of a property cluster, which includes additional properties such as `dbprop:name` and `dbprop:language`, which represent properties that may be considered as synonymous to `foaf:name` in the context of the class `dbpedia:Language` and are linked to each other in the SKP though the property `skos:closeMatch`. The property `dbprop:language` is another good example of a highly ambiguous property name, which is not easy to interpret, without actually looking at its detailed use with individuals of this particular class (i.e., individuals of `dbpedia:Language`). Another example of a property cluster is the one containing the object properties `dbpedia:spokenIn`, `dbprop:region`, and `dbprop:states`, which are all used to express the area, or usually the country, where a language is spoken.

The properties `dbprop:region` and `dbprop:states` did not have any prior range axioms defined, since they are not part of the DBpedia ontology, but rather of the part of DBpedia that is generated completely automatically without aligning it to the ontology. As an obvious remedy, one may consider using the range of `dbpedia:spokenIn` also for the other members of the cluster. However, not all properties are involved in clusters that include properties with range axioms in the reference ontology, this is actually true only for a small fraction of the total number of properties. Hence, although not absolutely necessary in this case, we may generate range axioms directly from data for the two properties. The property `dbprop:region` then, for instance, receives the union of the following classes as its range: `dbpedia:Place`, `dbpedia:PopulatedPlace`, `dbpedia:Settlement`, `schema:Place` and `opengis:_Feature`.

5 Experiments

In the related paper [15] the extraction of synonymous properties was evaluated, together with the property selection threshold. In this paper we focus on analysing the range extraction method, but additionally show some general statistics in order to motivate the usefulness of the SKPs we are proposing. For performing the experiments we selected a set of 34 DBpedia classes to focus on, and generated SKPs for these. The classes were not selected randomly, but rather we focused on the DBpedia classes that are involved in answering the benchmark queries in the QALD-1 query set[6], as our evaluation set.

5.1 Pattern Characteristics

SKPs aim at reducing the complexity of understanding and querying data, by reducing the diversity of properties to only include the core properties of the main SKP class. However, to be useful in practice, such a reduced representation should still allow for accessing as large part of the underlying data as possible. This is a trade-off that the SKPs must be able to sufficiently support if they are to be used in practice. To illustrate that the SKPs do fulfill both these requirements sufficiently well, Table 1 presents some statistics of the set of 34 SKPs in our evaluation set.

	Min	**Average**	**Max**
Number of included properties	31	107	436
Percentage of included properties	12%	27%	38%
Percentage of data triples covered	88%	94%	97%

Table 1. Characteristics of the generated SKPs

The patterns range in size (in terms of the number of properties of the main class) between 31 and 436 properties. While 436 properties may be perceived as a large number,

[6] QALD-1 contains a "gold standard" of natural language questions associated with appropriate SPARQL queries and query results, see: `http://greententacle.techfak.uni-bielefeld.de/~cunger/qald1/evaluation/dbpedia-test.xml`

this should be considered in light of the second row of the table, i.e. the fraction of the total number of properties used for that main class in the data that the included properties represent. For instance, the largest pattern, with 436 properties included, is the `AdministrativeRegion` pattern characterising the `AdministrativeRegion` class in the DBpedia ontology, which in total uses 1235 distinct properties with its 28229 instances in the DBpedia dataset. Hence, those 436 properties constitute only 35% of the total number of distinct properties, but still allows us to access 89% of the data triples, about `AdministrativeRegion` instances. In the last row of the table we summarise similar results for the complete SKP set, i.e., on average the SKPs allow us to still access 94% of the data about their instances, while reducing the number of properties to on average 27% of the original number. One should also keep in mind that these are SKPs generated with a particular property inclusion threshold (see [15] for a detailed evaluation and discussion of the threshold), whereby tailored sets of SKPs could also be generated with a specific use case in mind, prioritising either triple coverage or reduced size of the SKP as needed.

We have not yet evaluated how the accuracy of the data, and responses to queries, are affected by filtering out some portion of the properties used in data. This is mainly due to the difficulty of evaluating the quality of data in DBpedia in general, i.e., what is a correct triple and what is not? Ideally, we would like to be able to measure also how correct the data is, and evaluate if the data that is no longer accessible (if using only the SKP property set) is correct and useful data, or perhaps mostly consist of noise. However, we believe that crowdsourcing efforts such as the DBpedia Data Quality Evaluation launched, may be able to provide evaluation datasets that makes this feasible.

5.2 Range Extraction

For evaluating the range extraction method, which had to be done manually, a set of SKPs were selected (among the 34 we initially generated, corresponding to the QALD query classes). Unfortunately due to lack of evaluators, we were not able to evaluate the complete set of 34 SKPs, but had to focus on 8 SKPs that were randomly selected but where we made sure to cover both "small" and "large" SKPs (in terms of number of properties and range axioms). Using different cutoff thresholds for the inclusion of range classes, all the resulting proposals for range axioms were manually assessed by three evaluators (each range axiom was evaluated by at least 2 evaluators). The evaluators were asked to assess if the range class could be considered correct or not, in the context of the particular SKP main class, and for the property at hand. Initially, the evaluators simply assessed if the range class was correct or not (an "unsure" alternative was also available), but in addition, if deemed correct the evaluators were also asked to assess the level of abstraction of the range class. The latter, to evaluate if the method used was able to arrive at range classes that are neither too specific nor too general.

For instance, consider the SKP `Actor`, where the main class is `dbpedia:Actor`. This SKP includes a property `dbprop:spouse`, which relates an actor to his or her spouse. One class that is extracted as being part of the property range is the `dbpedia:Actor` class. However, despite this being a common type of the objects, it is not an appropriate range class – it is more of a coincidence that most actors are actually married to other actors, rather than a general axiom. A more appropriate class to include would be a superclass of `dbpedia:Actor`, i.e., `dbpedia:Person`. On the other hand, more general is not always better. Consider the superclass of `dbpedia:Person`, which is `dbpedia:Agent` (a class that also includes subclasses such as `dbpedia:Organisation`). This would not be an appropriate class either, since there are agents, e.g., companies, that cannot be the spouse of an actor. Through this example, we note that there is often a level of abstraction that is the most appropriate for expressing the range axioms, although more specific or more general classes cannot be considered as "wrong".

To combine the results of the three evaluators we have classified something as correct if at least one evaluator considered it correct, and the others either agreed that it was correct or were not sure. We have classified something as incorrect if, on the contrary, one evaluator considered it incorrect, and the others either agreed or were not sure. If the evaluators disagreed, e.g., one considering it correct and one incorrect, or they agree on the "unsure" alternative, the combined result is classified into the "unsure" category.

In Figure 2 we can see the results of the correctness evaluation of range axioms. On average, for each SKP, the method is able to find an appropriate range (one or more classes) for about 8 properties that were to be included in the SKP but that previously had no range axioms. In the figure we can see that for a cutoff threshold of 0.1 (meaning that a range class is included if it is the assigned type of more than 10% of the objects in triples using this specific property, and that are covered by this SKP) already around 80% of the proposed range classes are deemed as correct by the evaluators. This fraction increases as the cutoff threshold is raised, and at a threshold of 0.5 it is about 87%. As can be seen, the fractions of incorrect (and unsure) range classes stays well below 10% for a threshold of 0.3 and higher, and even before that the maximum fraction of incorrect suggested ranges is only about 12%.

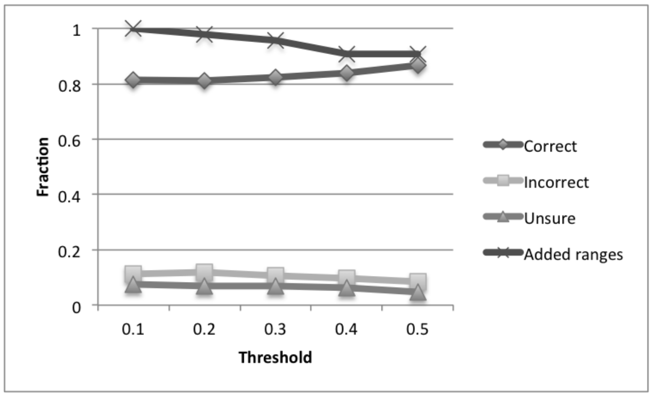

Fig. 2. Correctness of new range axioms, and fraction of properties that still receive a range axiom as threshold increases.

However, this increase in precision comes at a price of fewer suggested range axioms. In the figure we have therefore included also the "loss" of range axioms, in terms of the fraction of the properties where (correct) range axioms were proposed at threshold 0.1, but which when the threshold is increased no longer will have any range axiom in the SKP (called “Added ranges” in the diagram). When increasing the threshold above 0.3, this drop starts to become significant, e.g. going from 96% at the 0.3 threshold down to 91% at 0.4.

An additional drawback when raising the threshold, which is not directly visible in the figure, is the level of abstraction of the included range classes. In general, the agreement between evaluators is quite poor when it comes to evaluating the level of abstraction, and it varies quite a lot between the 8 SKPs that were assessed, hence, we do not provide any numerical results of this part of the evaluation. Instead, based on the cases when the evaluators do agree, and the trends in their individual assessments, we try to summarise some tendencies. The trend is that as the threshold increases, the first (correct, but not necessarily appropriate with respect to abstraction level) range axioms that are removed seem to be the ones that are considered too specific (c.f. `dbpedia:Actor` in the example above) by at least some evaluator. However, continuing to further increase the threshold, i.e., from 0.4 and onwards, seems to remove a significant amount of (agreed on) appropriate range classes as well as the overly general ones, hence, increasing the threshold too much seems to come with too much negative side-effects in terms of increasing the fraction of overly general range classes compared to the appropriate ones.

Based on these results, we conclude that, both from the perspective of including as many correct range axioms as possible without introducing too many errors, and from the (somewhat inconclusive) indications on appropriate level of generality, a selection threshold around 0.3 seems to be a reasonable pick. This threshold has been used for generating the SKPs in the current catalogue.

6 Discussion

Originally, the notion of Ontology Design Patterns (ODPs) referred solely to a top-down view on modelling best practices, and constituted manually designed patterns representing those best practices. More recently, however, the more general notion of KPs has been proposed, and such patterns have also been created in a bottom-up fashion, i.e., representing the way information on the Web or Linked Data is actually represented, rather than how it “should” be. It is highly relevant in this context to discuss the relation between best practices and patterns. Although we do agree that actual modelling patterns, found in data, do not necessarily conform to best practices, we also acknowledge that determining what is a “best practice” is very difficult. By investigating real-world data we observe actual practices, and by storing these as SKPs users are able to understand the current practice. For many use cases (e.g., querying or linking to data) it is more important to understand and adhere to *current practices*, rather than best practices that may not at all be used in the data at hand. Since our SKPs are dynamic, i.e., can be re-generated as soon as data changes, we envision that assuming data and model quality increases over time, the gap between best practices and actual practices is reduced.

Another general aspect of the SKPs that is worth mentioning is their generalisability over different datasets. Our experiments have so far been limited to DBpedia data, however, the method we are using is in no way restricted to this particular data. Although

DBpedia may be a particularly tricky dataset (due to its semi-automatic construction, and large coverage), we have observed that similar problems with duplicated properties and lack of ranges and other axioms do exist also in other datasets. However, the most interesting problem arises when starting to extract cross-dataset SKPs, which will be our next step. To find “synonymous” properties across vocabularies and datasets, and to be able to compare patterns between overlapping datasets is where we envision that the substantial benefits arise. The methods presented here are sufficiently general to be applied to this extended scenario with only minor modifications to the current implementation.

7 Conclusions and Future Work

KPs are more and more being extracted bottom-up, e.g., from Linked Data, rather than only being hand-crafted in a top-down fashion, e.g., as ODPs. This new kind of KPs is important since they can assist in making sense of datasets, and allow users and systems to formulate appropriate queries over data, while managing the diversity of properties used in datasets. Diversity of data representation, and lack of agreement on schemas and ontologies, is currently a major obstacle towards taking full advantage of the Semantic Web and Linked Data. Therefore, approaches like ours, for characterising and structuring data (e.g., by identifying synonymous properties and property ranges), are of essence.

This paper has provided an overview of our method for generating SKPs from Linked Data (details on the synonymy detection and property selection in [14,15]) focusing particularly on the final part; characterising the properties, e.g., through range axioms. Generally, SKPs can characterise classes from any reference ontology, by presenting their most frequent properties and property characteristics, based on analysing the underlying data. SKPs are stored as OWL ontologies but can be continuously updated in a completely automated fashion to reflect changes in the underlying data. We have exemplified the method by applying it to classes of the DBpedia ontology, and in particular we have thereby evaluated our method for extracting range axioms. Results show that by setting appropriate thresholds, SKPs can be generated that cover (i.e., allow us to query, using the properties of the SKP) over 94% of the triples about individuals of that class, while only needing to care about 27% of the total number of distinct properties that are used in the data. The range extraction method results in range axioms that are on average correct in 82% of the cases (merely 10% are clear errors), at the selected threshold level. These results clearly show that it is possible to make sense of data, and manage the diversity of Linked Data, by analysing the data and identifying the underlying patterns.

The catalogue of SKPs for the DBpedia classes is being published at the moment. While this will be an important resource, it is simply one example of a reference ontology that can be used. As future work we intend to publish the method described in the paper as a software component to be reused by others, over their dataset of choice. We also intend to extend the generated set of DBpedia-based SKPs, by taking into account other datasets that align to DBpedia, creating cross-dataset SKPs that can be used to formulate queries (and distribute queries) over several dataset. Another interesting line of future work is to use the SKPs in order to analyse data quality, similar to what is described for “key properties” in [1], by studying the triples that do not adhere to the pattern.

Acknowledgements

Part of this research has been sponsored by the EPSRC funded project LODIE: Linked Open Data for Information Extraction, EP/J019488/1.

References

1. Atencia, M., David, J., Scharffe, F.: Keys and pseudo-keys detection for web datasets cleansing and interlinking. In: Proc. of the 18th International Conference, EKAW 2012, Galway City, Ireland, October 8-12, 2012. LNCS, vol. 7603, pp. 144–153. Springer (2012)
2. Auer, S., Demter, J., Martin, M., Lehmann, J.: Lodstats - an extensible framework for high-performance dataset analytics. In: Proc. of the 18th International Conference, EKAW 2012, Galway City, Ireland, October 8-12, 2012. LNCS, vol. 7603, pp. 353–362. Springer (2012)
3. Basse, A., Gandon, F., Mirbel, I., Lo, M.: DFS-based frequent graph pattern extraction to characterize the content of RDF Triple Stores. In: Proc. of the WebSci10: Extending the Frontiers of Society On-Line, April 26-27th, 2010, Raleigh, NC: US [Online proc.] (2010)
4. Blomqvist, E.: Ontocase-automatic ontology enrichment based on ontology design patterns. In: Proc. of the 8th International Semantic Web Conference (ISWC 2009). LNCS, vol. 5823, pp. 65–80. Springer (2009)
5. Blomqvist, E., Sandkuhl, K.: Patterns in ontology engineering: Classification of ontology patterns. In: ICEIS 2005, Proc. of the Seventh International Conference on Enterprise Information Systems, Miami, USA, May 25-28, 2005. pp. 413–416 (2005)
6. Gangemi, A.: Ontology Design Patterns for Semantic Web Content. In: The Semantic Web ISWC 2005. LNCS, vol. 3729. Springer (2005)
7. Gangemi, A., Presutti, V.: Handbook on Ontologies, chap. Ontology Design Patterns. Springer, 2nd edn. (2009)
8. Gangemi, A., Presutti, V.: Towards a pattern science for the Semantic Web. Semantic Web 1(1-2), 61–68 (2010)
9. Musetti, A., Nuzzolese, A., Draicchio, F., Presutti, V., Blomqvist, E., Gangemi, A., Ciancarini, P.: Aemoo: Exploratory Search based on Knowledge Patterns over the Semantic Web (2011), [Finalist of the Semantic Web Challenge 2011]
10. Nuzzolese, A.G.: Knowledge Pattern Extraction and Their Usage in Exploratory Search. In: Proc. of the 11th International Semantic Web Conference (ISWC 2012). LNCS, vol. 7650, pp. 449–452. Springer (2012)
11. Nuzzolese, A.G., Gangemi, A., Presutti, V., Ciancarini, P.: Encyclopedic knowledge patterns from wikipedia links. In: Proc. of the 10th International Semantic Web Conference (ISWC 2011). pp. 520–536. LNCS, Springer (2011)
12. Presutti, V., Aroyo, L., Adamou, A., Schopman, B.A.C., Gangemi, A., Schreiber, G.: Extracting Core Knowledge from Linked Data. In: Proc. of the Second International Workshop on Consuming Linked Data (COLD2011), Bonn, Germany, October 23, 2011. vol. 782. CEUR-WS.org (2011)
13. Presutti, V., Blomqvist, E., Daga, E., Gangemi, A.: Pattern-based ontology design. In: Ontology Engineering in a Networked World, pp. 35–64. Springer (2012)
14. Zhang, Z., Gentile, A.L., Augenstein, I., Blomqvist, E., Ciravegna, F.: Mining equivalent relations from linked data. In: Proc. of the annual meeting of the Association for Computational Linguistics (ACL) 2013 (2013)
15. Zhang, Z., Gentile, A.L., Blomqvist, E., Augenstein, I., Ciravegna, F.: Statistical knowledge patterns: Identifying synonymous relations in large linked datasets. In: (To appear) Proceedings of ISWC2013. LNCS, Springer (2013)