Facet Annotation Using Reference Knowledge Bases

Riccardo Porrini*
Mia-platform
Milan, Italy
riccardo.porrini@mia-platform.eu

Matteo Palmonari
DISCo, University of Milan-Bicocca
Milan, Italy
palmonari@disco.unimib.it

Isabel F. Cruz
ADVIS Lab, Computer Science Dept.,
University of Illinois at Chicago
Chicago, Illinois, United States
isabelcfcruz@gmail.com

ABSTRACT

Faceted interfaces are omnipresent on the web to support data exploration and filtering. A facet is a triple: a domain (e.g., *Book*), a property (e.g., $author$, $language$), and a set of property values (e.g., $\{Austen, Beauvoir, Coelho, Dostoevsky, Eco, Kerouac, Sskind, \ldots\}$, $\{French, English, German, Italian, Portuguese, Russian, \ldots\}$). Given a property (e.g., $language$), selecting one or more of its values ($English$ and $Italian$) returns the domain entities (of type *Book*) that match the given values (the books that are written in English or Italian). To implement faceted interfaces in a way that is scalable to very large datasets, it is necessary to automate facet extraction. Prior work associates a facet domain with a set of homogeneous values, but does not annotate the facet property. In this paper, we annotate the facet property with a predicate from a reference Knowledge Base ($\mathcal{KB}$) so as to maximize the *semantic similarity* between the property and the predicate. We define semantic similarity in terms of three new metrics: *specificity*, *coverage*, and *frequency*. Our experimental evaluation uses the DBpedia and YAGO $\mathcal{KB}$s and shows that for the facet annotation problem, we obtain better results than a state-of-the-art approach for the annotation of web tables as modified to annotate a set of values.

CCS CONCEPTS

• Information systems → Information integration; **• Applied computing** → Electronic commerce;

KEYWORDS

Facet annotation; data semantics; data lifting; table annotation; faceted search; eCommerce

ACM Reference Format:
Riccardo Porrini, Matteo Palmonari, and Isabel F. Cruz. 2018. Facet Annotation Using Reference Knowledge Bases. In *WWW 2018: The 2018 Web Conference, April 23–27, 2018, Lyon, France.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3178876.3186020

1 INTRODUCTION

A considerable amount of information published on the web, for example in eCommerce, is accessible through faceted interfaces, which let users filter query results and explore the information space [7, 8, 12, 24, 39]. Figure 1 shows an example of the Amazon user interface to filter books using the property values for $author$ ($\{MiaSheridan, LaurannDohner, \ldots\}$) and/or for $language$ ($\{English, German, \ldots\}$). In a Comparison Shopping Platform (CSP), which aggregates different marketplaces, faceted search filters the relevant information from millions of products of different categories.

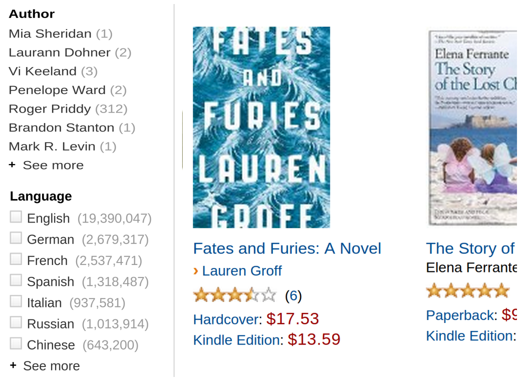


Figure 1: Facets that characterize books.

In information science, a facet is a *characteristic* of a particular *universe* of entities, with each characteristic being associated with a set of possible *terms* [15]. We use the terminology *property*, *domain*, and *values*, respectively. We define a *facet* as a triple $\langle D, p, V \rangle$, where D is the *domain* (set of entities), p the property, and $V = \{v_1, \ldots, v_n\}$ the set of property values. As an example, we represent the *language* facet in Figure 1 as a triple $\langle Book, language, \{English, German, French, \ldots\}\rangle$.

Each property-value pair $\langle p, v_i \rangle$ selects a subset of entities with value v_i for property p, thus supporting a fine-grained and modular classification within a given domain [15, 25]. For example, the pair $\langle language, English \rangle$ can be used to select the books (entities) that are written in English using the faceted interface shown in Figure 1. This definition generalizes to a set of values by taking the union of the sets of entities selected by each of the values.

Well-crafted faceted interfaces enhance user experience [15, 33, 40], which turns into competitive advantage in sectors like eCommerce. Different theories including facet analysis have been proposed in information sciences to drive the process of creating facets [15, 25]. However, creating well-crafted faceted interfaces for different domains requires considerable manual effort. As a result, large eCommerce portals still contain a long tail of product categories for which domain-specific faceted interfaces are not defined. Thus, facet extraction algorithms have been proposed to

*Work by Riccardo Porrini performed in part while employed by 7Pixel S.r.l., enrolled in the PhD program at the Univ. of Milan-Bicocca, and visiting the ADVIS Lab at UIC.

automatically extract facets or to assist domain experts in the facet creation process. These approaches extract and cluster facet values that are deemed to belong to a common property. Some approaches extract clusters of facet values for a domain using data analytics techniques offline [6, 19, 24, 38], while others extract clusters of facet values from search results [8, 13, 42]. Many facet extraction approaches provide hierarchical faceted search over large text collections [6, 19, 38, 42], but they are seldom used in eCommerce.

The above approaches are able to extract clusters of facet values but they do not interpret the facet property. In this paper, we address the *facet annotation problem*, that is, the problem of assigning a property to automatically extracted sets of values with the same semantics. The key insight behind facet annotation is that we can re-use predicates from large structured Knowledge Bases ($\mathcal{KB}$s) such as DBpedia [16], YAGO [34], Freebase [3], or corporate $\mathcal{KB}$s, to interpret the meaning of a facet. We consider $\mathcal{KB}$s with entities classified using types and *relational assertions*, which use binary predicates between entities or between entities and literal values. In these $\mathcal{KB}$s, predicates have both machine-readable semantics, as provided by web standards like RDF, and human-readable semantics, as provided by natural language labels. The input to the facet annotation problem is a triple $\langle D, ?p, V \rangle$ where $?p$ represents an unknown property. Given this input, we want to find a predicate q from a $\mathcal{KB}$ that captures the semantics of the property. For instance, we annotate the properties *author* and *language* of Figure 1 with the predicates *dbo:author* and *dbo:language* used in DBpedia. Facet annotation thus consists of evaluating the similarity between the unknown property and a predicate in a $\mathcal{KB}$. Predicates in the $\mathcal{KB}$ can be ranked by similarity in such a way that the most similar predicate can be chosen using a fully automatic facet annotation method or, in a semi-automatic way, a set of the top-k most similar predicates can be evaluated by domain experts.

Facet annotation problems with input $\langle D, ?p, V \rangle$ are relevant in (at least) two scenarios: facet values are either extracted for a given domain (e.g., a product category without specific facets in an eCommerce application [24]) or from search results [13] that have been filtered by a category (e.g., results for the search "Harry Potter" restricted to the *Book* domain). In both scenarios, $\mathcal{KB}$ predicates annotate unknown facet properties. We focus on the first scenario, which has motivated our work. By combining facet extraction [24] and facet annotation we further automate the facet definition process, and provide an end-to-end solution for the complex task of maintaining and creating facets in a CSP, which can operate in highly dynamic environments where novel products and categories appear at a fast pace [11]. In addition, the reuse of property URIs from a reference $\mathcal{KB}$ enables semantic annotations of faceted classifications in a CSP using, for example, RDFa markup [2], which supports the smart aggregation of product data and semantic search [33, 35]. Finally, vocabulary reuse reduces terminology entropy and is a key factor to ease semantic data access [29].

Solving the facet annotation problem is challenging because facet values are ambiguous, their interpretation is highly domain dependent, and it is difficult to determine which predicates capture more specifically the semantics of a facet. For example, in Figure 1, English, German, French, Spanish may refer to the language of a book or to the nationality of a basketball player. To describe book authors, the predicate *author* should be preferred to more general predicates such as *creator*. We address these challenges by providing a representation of the $\mathcal{KB}$ that eases the matching of facet values with relational assertions from the $\mathcal{KB}$, and the matching of the facet domain with the entity types of the $\mathcal{KB}$. We design a ranking function that is able to effectively quantify semantic similarity.

The facet annotation problem is related to relation annotation in web tables [4, 17, 22, 26, 27, 30, 36, 37, 43, 44]. Table annotation aims to interpret semi-structured tables embedded in web pages by annotating cells with named entities, columns with entity types, and column pairs with predicates representing the relation that holds between column values (*relation annotation*). A facet describes a relation between an entity type (e.g., *Book*) and facet values, which is *imbalanced* when compared with the *balanced* relation between a pair of table columns. Although table annotation approaches can be adapted to work with the imbalanced input of the facet annotation problem, our approach significantly outperforms such adaptations by better capturing the similarity between facets and $\mathcal{KB}$ predicates, as we will show in our experiments.

In summary, in this paper we make the following contributions:

- we define the *facet annotation problem* and its motivation along with the semantic similarity between a facet and a predicate from a $\mathcal{KB}$, in Section 2;
- we discuss the similarities and differences between facet annotation and table annotation and, in particular, relation annotation, in Section 3;
- we propose a *filter and rank* facet annotation approach which, given a facet, selects predicates from a $\mathcal{KB}$ and ranks them according to their similarity, in Section 4;
- we provide an extensive evaluation using $\mathcal{KB}$s with different characteristics and compare our approach with state-of-the-art relation annotation approaches adapted to facet annotation, in Section 5, before concluding the paper in Section 6.

2 PROBLEM SETTING

In this section we introduce a motivation scenario for our work, and define the three main concepts of *facets*, *Knowledge Bases*, and *facet annotation*, and highlight the intuition behind our approach.

2.1 Motivating Scenario

Our work addresses the needs of domain experts who are in charge of maintaining classifications, mappings, and facets in a CSP. Automatic methods that define new facets are helpful for the long tail of product categories. We have developed an automatic facet extraction algorithm [24] for a leading CSP in Italy, which could be used for platforms like Amazon or AliBaba, which have millions of users worldwide. Given a product category, our algorithm finds clusters of homogeneous values, each one corresponding to one facet. The clusters are disjoint, thus making facets orthogonal as recommended in information science [15]. The clusters are computed by analyzing mutually disjoint terms occurring in thousands of taxonomies from the local marketplaces whose data is integrated in the CSP. Domain experts can refine the suggested facet values, assign a label to the facet property and decide which facets will be finally included in the CSP. The tool has been used in the production environment of the CSP to create new facets for 16 product categories in about one year (out of 512 product categories).

Our facet anotation algorithm outputs a ranked list of suitable predicates for the property of each extracted facet using a reference $\mathcal{KB}$. Domain experts can use this list to select a suitable predicate q, which will provide a label and a URI for the facet property.

During the facet extraction process, the relation between the individual entities (product offers) and the facet values, which come from local taxonomies, is lost. Thus, when the facet annotation algorithm is run, the entity-value pairs, similarly to those used in instance-based relation annotation, are not available. Once facets are defined, pattern matching is used to reconstruct this relation and thus support faceted search in the CSP interface. In addition, entities are usually represented by internal identifiers that cannot be matched with a reference $\mathcal{KB}$, with offer titles (e.g., Apple iPhone 5C 8 GB Unlocked, Blue) containing noisy descriptions of product features rather than their clean (standard) names (e.g., Apple iPhone 5C 8 GB). This is the same problem that occurs in facets extracted with query-driven approaches.

Depending on the domain of interest, the reference $\mathcal{KB}$ can be an open, cross-domain $\mathcal{KB}$ like DBpedia, or a proprietary $\mathcal{KB}$ used to describe the products in a CSP. We created a proprietary $\mathcal{KB}$ of the products in the CSP we have worked with. However, to support a comparative evaluation and the reproducibility of our results, we use open $\mathcal{KB}$s in this paper.

2.2 Facet Annotation

A Knowledge Base, $\mathcal{KB}$, is a collection of descriptions of *entities*, which are classified using *types*. Adhering to the RDF data model [18], entities, predicates and types are uniquely identified by their URIs. Entities are described by *assertions* (s, q, o), where s, q and o are respectively the subject, the predicate, and the object of the assertion. The subject of an assertion is an entity, while the object can be either an entity, a *literal value* (e.g., 2010), or a type. We distinguish between *relational assertions*, where the subject is an entity and the object is either an entity or a literal value, and *typing assertions*, where the subject is an entity and the object is a type. Typing assertions are represented explicitly for entities or can be derived from typed literals (e.g., "1978-07-20"^^xsd:date) [18].

Many $\mathcal{KB}$s, including DBpedia, also use a reference ontology, which specifies the intensional semantics of types and predicates using a formal language like OWL. In these ontologies, ontology classes and datatypes are entity types. However, several $\mathcal{KB}$s also use complementary classification systems. For example, DBpedia uses Wikipedia categories, which we also consider to be entity types. Types are organized in a *subtype graph*, which represents a binary and transitive relation $\preceq$ that defines a partial order. Therefore if t' is a *subtype* of t or equivalently *more specific* than t, then $t' \preceq t$. The subtype t' of t can be the child of t or any descendant of t. Each entity $e \in \mathcal{KB}$ is an instance of one or more types. We denote by $T(e)$ the set of types of e, including all their supertypes. A $\mathcal{KB}$ may use more than one type graph (e.g., DBpedia). Both types and entities are associated with a set of *lexicalizations*. A lexicalization is a sequence of natural language tokens that are associated with a type, or an entity, or a literal value. We denote by $L(t)$ and $L(e)$ the lexicalization of a type t and of an entity e, respectively. For instance, for an entity J_R_R_Tolkien we may have L(J_R_R_Tolkien) = {"John Ronald Reuel Tolkien", "J. R. R. Tolkien"}. The lexicalization of a literal value is the set that contains the value itself. The extensional semantics of a predicate q is defined by its extension, that is, every pair (s, o) such that $(s, q, o) \in \mathcal{KB}$. Given a predicate q, its *domain* D^q and *range* R^q are defined respectively as the sets of all the subjects and objects, which occur in the extension of q.

In a facet f we use D^f, p^f, and V^f to denote respectively its domain, property, and values. The lexicalization of a facet domain is the string that denotes it (e.g., "Book"). When the property is unknown, we write $f = \langle D^f, ?p^f, V^f \rangle$ (denoting, for example, the output of a facet extraction algorithm). Given the facet f and a reference $\mathcal{KB}$, the problem of facet annotation consists of finding a predicate q from the $\mathcal{KB}$, such that the semantics of q is *similar* to the semantics of $?p^f$. Our approach to the annotation of a facet property consists of building a list of $\mathcal{KB}$ predicates ranked by similarity. It is up to an automatic program or to a user to select one of them, likely the top ranked one.

2.3 Specificity Driven Semantic Similarity

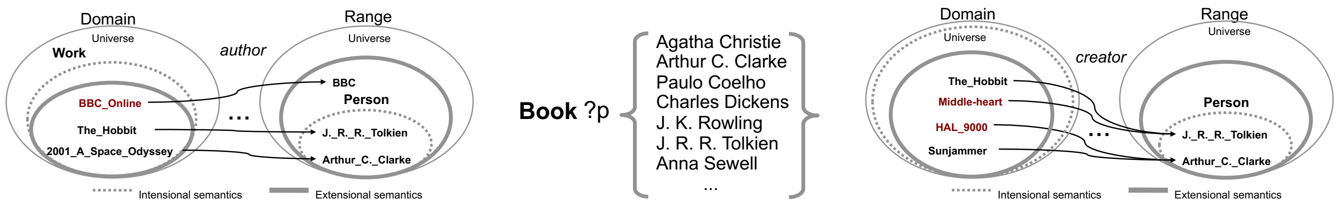
The similarity of a predicate from a $\mathcal{KB}$ to a facet property relies on three metrics: *specificity*, *coverage*, and *frequency*. The most important metric is *specificity*. We use the example of Figure 2b to explain its importance.[1] The facet values include, for instance, Arthur C. Clarke, and J. R. R. Tolkien, while the facet domain is *Book*. The two predicates *author* (Figure 2a) and *creator* (Figure 2c) relate some books (2001_A_Space Odyssey, The_Hobbit, and Sunjammer) to J._R._R._Tolkien and/or Arthur_C._Clarke. For each predicate there are two relational assertions that provide evidence for its similarity to the facet property $?p$. However, looking at the extension of *creator* in the $\mathcal{KB}$, we realize that the domain of *creator* includes not only books but also fictional characters (HAL_9000) and places (Middle-heart). Instead, the domain of *author* contains prevalently books, with BBC_Online being of type *Work*, a parent of *Book*. Thus, the predicate *author* is a better candidate than *creator* to annotate the facet. To determine the best possible predicate, we need to compare the facet domain with the predicate domain and the facet values with the predicate range and choose the closest.

Coverage is based on the estimated overlap between the facet values and a predicate range. A range R^q that contains a large subset of the facet value set V^f provides evidence for the similarity between q and $?p^f$.

Frequency is orthogonal to coverage. Given a facet f the more frequently facet values are connected by a predicate q to instances from the domain D^f, the higher the similarity between q and the facet property $?p^f$ is. The rationale behind frequency is to favor predicates that are more frequently used within the $\mathcal{KB}$ with respect to the facet domain, for instance favoring predicates like *language* over *origLanguage*.[2]

Given a facet, we will quantify the *specificity*, *coverage*, and *frequency* of predicates, by looking at their extensional semantics. Another possibility would be to consider the intensional semantics of predicates specified in the ontology of the $\mathcal{KB}$ and solve the facet annotation problem as an ontology matching problem [5, 10, 31]. However, facet properties have no attached semantics and thus there is nothing that can be directly compared to the semantics of

[1]The example is from DBpedia (version 3.9) with some simplifications.
[2]Both of them are used in DBpedia to denote the original language of a book.



(a) A *specific* predicate w.r.t. books. **(b) A facet describing authors of books.** **(c) A *generic* predicate w.r.t. books.**

Figure 2: An example that illustrates the difference between specific and generic predicates.

the predicate. Also, facet domains refer to specific types whereas facet values refer to entities (e.g., the entity Arthur_C._Clarke) or literals (e.g., 2010). Finally, the semantics of predicates is often underspecified in large $\mathcal{KB}$s: for several predicates, the ontology does not constrain the subjects and/or objects related by such predicates to belong to specific types [1, 32]. For these reasons, the intensional approach is not directly applicable to the facet annotation problem.

3 RELATED WORK

Facet annotation has similarities with web table annotation. Tables often include a *subject column* containing the subject entities described by the table [36, 37, 43]. The remaining columns contain entities or literals that are in a relation with the subject entities or with entities appearing in other columns. Entities, types, or relations are annotated using structured $\mathcal{KB}$s, such as YAGO [17, 30], DBpedia [26, 44], the combination of the two [22], Freebase [43], Probase [37], or custom $\mathcal{KB}$s mined from web pages [36].

Facet annotation is closer to relation annotation that concentrates on non-subject columns, hence, in what follows, we focus on those approaches that annotate non-subject columns [17, 22, 36, 37, 43], not on those that do not [26, 30, 44]. However, as mentioned in Section 1, facet annotation is *imbalanced* and relation annotation is *balanced*. Relation annotation relies heavily on the analysis of the reciprocal distributions of values in different columns on the same row [36, 43]. Due to the imbalanced specification of facets, this kind of information is not available when annotating the facets. Still, because we leverage the extensional semantics of predicates in a $\mathcal{KB}$ similarly to what is done for relation annotation [17, 22, 36, 37, 43], we discuss the adaptability of relation annotation approaches to facet annotation. As discussed in a recent article [27], relation annotation is also similar to instance-based ontology matching when applied to properties [31]. Our considerations about relation annotation also apply to instance-based property matching.

The $\mathcal{KB}$ of Venetis et al. [36] is a custom database of mined relations. They use a maximum likelihood method for relation annotation by computing the frequencies within the $\mathcal{KB}$ of all pairs of values in the same row of the table, without considering type information. Hence, they do not quantify the specificity of a predicate with respect to an entity domain, which is crucial for a facet annotation algorithm. We compare experimentally our approach with their maximum likelihood method as adapted to facet annotation and show that ours is more effective. Wang et al. [37] use the Probase probabilistic $\mathcal{KB}$ [41], built similarly to the $\mathcal{KB}$ of Venetis et al., to annotate web tables. Their approach depends on the probabilstic model used in Probase, while we rely solely on types and relational assertions, which are present in any $\mathcal{KB}$.

TableMiner [43] annotates tables with entities and types using an iterative refinement process, and, after this process, with predicates, using Freebase as $\mathcal{KB}$. It uses contextual information extracted from the web page that contains the table (e.g., table caption, surrounding text, Microdata annotations), and the interdependence among columns. However, we lack contextual information in our facet annotation setting. The interdependence among table columns was leveraged also in earlier approaches, which perform collective inference to jointly compute the optimal annotations for different columns [17, 22]. However, facets lack the structure of tables.

Ritze et al. [28] first disambiguate values by matching them to entities in the $\mathcal{KB}$. Then they match pairs of those disambiguated values with candidate predicates. In turn, the matched predicates are used to refine value disambiguation in an iterative process. Value pairs are not available in the imbalanced definition of the input facet, which means that their method has to be adapted to facet annotation. Ranking of candidate predicates is based on weighted majority voting, where weights are based on matching scores computed between the values in the table and the entities in the $\mathcal{KB}$ that are linked to them. The adaptation of this principle to our imbalanced representation becomes similar to the evaluation of (weighted) frequency, which is only one of the three measures we use.

Pham et al. [23] learn the semantic labels of table columns. Semantic labels are represented by pairs $\langle class, predicate\rangle$, where the *class* indicates the type of the subject entities that are in a relation with the values in the column. They evaluate the similarity between the column and $\mathcal{KB}$ predicates by comparing the respective lists of values (textual and numerical), and the column header with the predicate names. The computed similarity scores are used as features in a Logistic Regression classifier. While we also compute the similarity between a facet and a predicate by comparing lists of values, we cannot use anything similar to the column header, which is shown to be important for the classifier. Finally, while their approach is supervised, our approach to facet annotation is unsupervised.

4 FACET ANNOTATION

We may consider a relational assertion $(s, q, o) \in \mathcal{KB}$ as positive evidence of the similarity between a facet property f and q if: (1) s is an instance of the same type described by the facet domain D^f, and (2) o is equal to a facet value $v \in V^f$. However, it is also the case that the facet domain and the facet values must be matched with types and predicate objects from the $\mathcal{KB}$. Furthermore, we cannot assume that the $\mathcal{KB}$ completely covers the domain of knowledge conceptualized by f. In this situation, it may be that the $\mathcal{KB}$ only partially represents the domain of knowledge conceptualized by

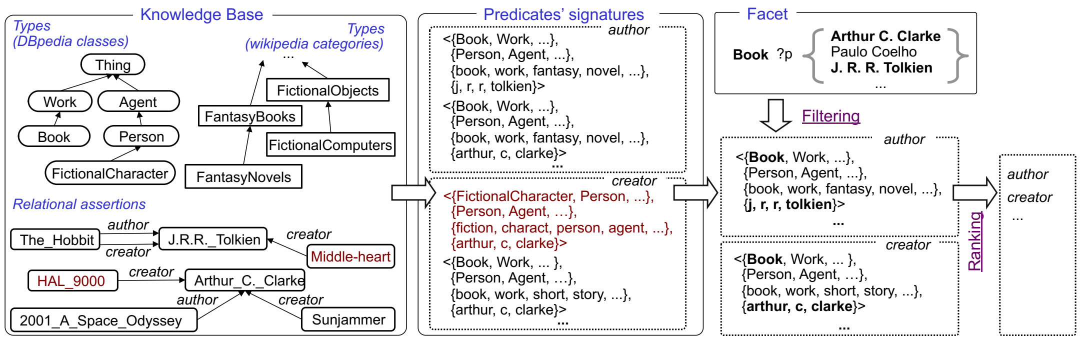

Knowledge Base
Types (DBpedia classes)
Types (wikipedia categories)
Thing
Work
Agent
Book
Person
FictionalCharacter
FictionalObjects
FantasyBooks
FictionalComputers
FantasyNovels
Relational assertions
The_Hobbit
author
creator
J.R.R._Tolkien
Middle-heart
HAL_9000
Arthur_C._Clarke
2001_A_Space_Odyssey
Sunjammer
Predicates' signatures
author
<{Book, Work, ...}, {Person, Agent, ...}, {book, work, fantasy, novel, ...}, {j, r, r, tolkien}>
<{Book, Work, ...}, {Person, Agent, ...}, {book, work, fantasy, novel, ...}, {arthur, c, clarke}>
creator
<{FictionalCharacter, Person, ...}, {Person, Agent, ...}, {fiction, charact, person, agent, ...}, {arthur, c, clarke}>
<{Book, Work, ... }, {Person, Agent, ...}, {book, work, short, story, ...}, {arthur, c, clarke}>
Facet
Book ?p
Arthur C. Clarke
Paulo Coelho
J. R. R. Tolkien
Filtering
Ranking
author
creator

Figure 3: The Facet Annotation process.

a facet (i.e., some book authors may not correspond to any entity in the $\mathcal{KB}$). Thus, we cannot consider it as a source of negative evidence whenever the facet domain cannot be matched to any $\mathcal{KB}$ type, or some facet values cannot be matched to any object. For this reason, our approach relies on positive evidence only.

Our facet annotation approach is depicted in Figure 3 and it consists of two distinct phases: predicate *filtering* and predicate *ranking*. Given a facet f, we first match the facet domain and the facet values to (predicate) *signatures*, which provide representations of the extensions of the predicates derived from the relational assertions in the $\mathcal{KB}$. The signature of a pair (s, o) lists the subject types, the object types, the lexicalization of the subject type and the lexicalization of the object, thus supporting comparison with the imbalanced specification of a facet. The result of the filtering phase is a set of matching signatures for each $\mathcal{KB}$ predicate q, which we refer to as the *extensions of q induced by f*. In the ranking phase, we analyze the non empty extensions of predicates induced by f in order to quantify the degree of similarity between a predicate q and a facet f, adhering to the *specificity*, *coverage*, and *frequency* principles described in Section 2.

4.1 Predicate Filtering

A signature $\delta_{(s,o)}$ is an *abstract* and *lexicalized* representation of a pair (s, o) that belongs to the extension of a predicate q. It is *abstract* as it includes all the types of the subject s as well as the types of the object o. It is *lexicalized* as it includes the lexicalization of o as well as the lexicalization of *all* the types of s. Lexicalizations of entities and types are extracted from URIs and literal values of the *rdfs:label* property. More formally, a signature $\delta_{(s,o)}$ is defined as

$$\delta_{(s,o)} = \langle T(s),\ T(o),\ L^T(s),\ L(o) \rangle \tag{1}$$

where $T(\mathrm{s})$, $T(\mathrm{o})$ are the set of types of the subject and the object, respectively, and $L(\mathrm{o})$ is the lexicalization of the object. Special attention must be given to the set of subject types lexicalizations $L^T(s) = \bigcup_{t \in T(s)} L(t)$, which is defined as the union of the lexicalizations of *all* the types of the subject. A signature explicitly records the co-occurrence of the subject and object types. Signatures are indexed and stored in an inverted index, so as to support the filtering phase, where predicate extensions are selected by matching the facet with the signatures.

We represent the *extension of a predicate* q, denoted by Δ_q, as the set of extracted signatures $\delta_{(s,o)}$ for that predicate

$$\Delta_q = \{\delta_{(s,o)} \mid (s, q, o) \in \mathcal{KB}\} \tag{2}$$

To collect the evidence needed to evaluate the similarity between a given facet property $?p^f$ and predicates in the $\mathcal{KB}$ and to filter out irrelevant predicates, we match f with the signatures. Our approach is based on the combination of two different matching functions: for the facet domain, *d-match*, and for the facet values (range), *r-match*. The *d-match* function selects the signatures whose subject types match the facet domain. In order to accomplish this task we consider the lexicalizations of the subject types $L^T(s)$ and compare them with the lexicalization of the facet domain $L(D^f)$. Orthogonally, the *r-match* function selects those signatures whose object matches *at least one* facet value. We then combine *d-match* and *r-match* so as to select the extension Δ_q^f induced by f, which is defined as the signatures whose subject types match the facet domain *and* the object matches at least one facet value

$$\Delta_q^f = \{\delta_{(s,o)} \in \Delta_q \mid d\text{-}match(f, \delta_{(s,o)}) \\ \wedge\ r\text{-}match(f, \delta_{(s,o)})\} \tag{3}$$

When $\Delta_q^f = \emptyset$, we can conclude that f and q are not similar, because there is no evidence in $\mathcal{KB}$ of their similarity, and thus we can exclude q from the subsequent predicate ranking phase.

In the filtering phase, we do not try to disambiguate facet values using the $\mathcal{KB}$ entities and some entity linking algorithm for a number of reasons. First, we seamlessly process facet values that contain entities and literal values, such as dates or numbers. Second, entity linking algorithms are usually optimized for text [21], while algorithms tailored for tables use table rows as context to support disambiguation [9]. We have plain sets of values, which would provide little context to support disambiguation. Our approach smooths the impact of possibly incorrect matches found in the filtering phase by collectively considering the evidence coming from all matches collected for every value.

4.2 Predicate Ranking

In the *predicate ranking* phase, we encode all the principles described in Section 2 (i.e., *specificity*, *coverage*, *frequency*) into a set of scores that is aggregated into an overall semantic similarity score. Finally, we rank the predicates accordingly.

Specificity. We capture the *specificity* of a predicate q with respect to facet f by comparing the extension Δ_q with the extension Δ_q^f induced by f. If Δ_q^f is close to Δ_q, we may consider the semantics of q to be somehow preserved when annotating f with q. In principle, to capture this closeness, one may apply a majority voting approach, that is, the more Δ_q^f and Δ_q overlap (i.e., more votes for q), the more we can consider q similar to f. However, this majority voting approach will likely fail to capture the specificity of q with respect to f. The cardinality of V^f is usually several orders of magnitude smaller than the cardinality of R^q, thus making the direct comparison of the two extensions not discriminative enough. Moreover, Δ_q^f is the result of a matching algorithm and it may include signatures resulting from false positive matches. To account for these issues, we consider types instead of signatures. We compare the sets of subject and object types extracted from the signatures in Δ_q with the ones extracted from signatures in Δ_q^f. We follow the intuition that the more these sets are similar, the more Δ_q^f is close to Δ_q and thus the semantics of q is preserved.

Given a generic extension Δ, we form the subject type set $D[\Delta]$ by selecting the types of all the subjects in the signatures of Δ. More formally, the subject type set $D[\Delta]$ of an extension Δ is defined as

$$D[\Delta] = \{t \mid \exists\delta_{(s,o)} \in \Delta,\ t \in T(s) \wedge \nexists t' \in T(s), t' \preceq t\} \quad (4)$$

$D[\Delta]$ does not contain *every* type of every subject, but only the types of every subject that are *minimal.* A type t is minimal for an entity e if there are no other types in $T(e)$ that are subtypes of t.

By considering the minimal types only, $D[\Delta]$ captures the specificity of the domains of Δ, as $D[\Delta]$ contains the most *specific* types that are used to categorize the subjects of the signatures from Δ. We then capture the *domain-specificity* of a predicate q with respect to f, $d\text{-}spec(f,q)$, by comparing the subject type sets extracted from the two extensions Δ_q and Δ_q^f. We consider q specific with respect to f if their extensions contain the same set of specific subject types and compare $D[\Delta_q]$ with $D[\Delta_q^f]$ using the well-known weighted Jaccard set similarity

$$d\text{-}spec(f,q) = \frac{w\text{-}card(D[\Delta_q] \cap D[\Delta_q^f])}{w\text{-}card(D[\Delta_q] \cup D[\Delta_q^f])} \quad (5)$$

The *w-card* (for *weighted cardinality*) function in Equation 5 computes the weighted sum of the depths of the types $t \in T$ in the subtype graph, each depth being weighted by the inverse of the maximum depth of the descendants of t

$$w\text{-}card(T) = \sum_{t \in T} \frac{depth(t)}{\max\limits_{t' \in DES[t]} depth(t')} \quad (6)$$

where $DES[t] = \{t' \mid t' \preceq t\}$. The rationale of using this scaled depth is: (1) to ensure values in the $[0,1]$ interval, and (2) to further bias the function *d-spec* towards the most specific subject types.

We compute also the *range-specificity* of q with respect to f, $r\text{-}spec(f,q)$, by adapting the *d-spec* function to consider the object types instead of the subject types. Similarly to Equation 4, we define the set $R[\Delta]$ of object types extracted from a generic extension Δ as

$$R[\Delta] = \{t \mid \exists\delta_{(s,o)} \in \Delta,\ t \in T(o) \wedge \nexists t' \in T(o), t' \preceq t\} \quad (7)$$

We thus adapt Equation 5 accordingly

$$r\text{-}spec(f,q) = \frac{w\text{-}card(R[\Delta_q] \cap R[\Delta_q^f])}{w\text{-}card(R[\Delta_q] \cup R[\Delta_q^f])} \quad (8)$$

Coverage. We capture the *coverage* of a predicate q over a facet property f, $cov(f,q)$ by computing the fraction of facet values for which there exists at least one matched signature $\delta_{(s,o)} \in \Delta_q^f$

$$cov(f,q) = \frac{|\{v \in V^f \mid \exists\delta_{(s,o)} \in \Delta_q^f\}|}{|V^f|} \quad (9)$$

Intuitively, a higher coverage is estimated for those predicates whose objects cover larger subsets of the facet value set V^f.

Weighted frequency. Specificity and coverage do not take into account the cardinality of the extension Δ_q^f, where a large extension provides evidence for q and f being similar. To capture this principle, we introduce the *weighted frequency*, $freq(f,q)$

$$freq(f,q) = \frac{1}{|V^f|} \sum_{\delta_{(s,o)} \in \Delta_q^f} \frac{|L^T(s) \cap L(D^f)|}{|L^T(s) \cup L(D^f)|} \quad (10)$$

Intuitively, we count signatures in Δ_q^f, weighting them considering the similarity between the lexicalization of the facet domain $L(D^f)$ and the lexicalization of all the subject types $L^T(s)$. Our intuition is that the more similar $L(D^f)$ and $L^T(s)$ are, the higher quality of the evidence provided by the signature. In Equation 10 the quality of each signature Δ_q^f is computed as the Jaccard similarity between the two lexicalizations. The factor $1/|V^f|$ is introduced to scale down the resulting weighted frequency by the number of the facet values (i.e., the cardinality of V^f).

The idea of evaluating specificity and frequency on top of the matches found for facet values and domain using domains and ranges of $\mathcal{KB}$ predicates is somewhat similar to the idea behind the Information Retrieval SoftTFIDF measure. However, when used to compare two documents of m and n words, respectively, SoftTFIDF computes string similarity between $m \cdot n$ pairs of words to find the most similar word pairs. SoftTFIDF is thus effective in comparing short text fragments like entity names [20], but is not suitable for the comparison of longer text fragments such as those that would encode facet values and, in particular, predicate ranges (which may contain several thousand values). In addition, our *specificity* and *frequency* measures go beyond value comparison, by considering minimal types.

Final Rank Computation. We aggregate the scores that capture the specificity, coverage, and frequency into a single, global similarity score, expressed by the *domain and range comparison* function for a predicate q and a facet f, $drc(f,q)$. The scores have values in the $[0,1]$ interval, with the exception of the weighted *freq* score. The *freq* score may in principle have a huge impact on the overall score because it provides positive evidence captured in an unbounded way. As a result, it can possibly skew the overall score. Thus, to make all the scores more comparable we smooth them using a logarithmic function. The lower bound of each score is one and then we multiply all the smoothed scores together, resulting in a similarity score that is a non-decreasing monotonic function with no upper bound and a lower bound equal to 1

$$\begin{aligned} drc(f,q) = &\Big(1 + \ln(d\text{-}spec(f,q) + 1)\Big) \cdot \\ &\Big(1 + \ln(r\text{-}spec(f,q) + 1)\Big) \cdot \\ &\Big(1 + \ln(cov(f,q) + 1)\Big) \cdot \\ &\Big(1 + \ln(freq(f,q) + 1)\Big) \end{aligned} \quad (11)$$

The +1 inside the logarithm ensures each score is greater or equal to zero, while the +1 outside the logarithm enforces 1 as the lower bound. Intuitively, we let each score contribute by a positive factor, as we only consider positive evidence.

5 EVALUATION

In our experiments, we annotate facets with predicates from DBpedia and YAGO $\mathcal{KB}$s. We compare our ranking approach with the *majority voting* model and the *maximum likelihood* model [36] , as adapted to the facet annotation problem. We rely on two different gold standards for the annotation: one that is manually created for DBpedia and another one that is the state-of-the-art for YAGO and was used in the evaluation of table annotation [17].

We implemented our approach along with the baseline algorithms using the Java programming language and the Lucene library.[3] The code repository, the gold standards with which we compared our approach, and the experimental results provided in this section are publicly available.[4]

Table 1: Gold Standards' statistics.

	Facets		*Predicates*	
	#	**Domains**	**Accurate**	**Correct**
dbpedia-numbers	8	7	~4	~19
dbpedia-entities	31	13	~7	~7
dbpedia	39	13	~3	~9
yago-explicit	83	17	1	-
yago-ambiguous	83	10	1	-

5.1 Gold Standards

Knowledge Bases. DBpedia (version 3.9) contains 753958 types, 53195 predicates and more than 96M relational assertions. In our experiments, we include two type graphs: one extracted from the DBpedia ontology, and the other extracted from DBpedia categories. DBpedia categories are extracted from Wikipedia categories and are known to be noisy and low quality [14]. The second $\mathcal{KB}$ we consider is YAGO (version: 2008-w40-2),[5] which includes 184512 types, 89 predicates and more than 5M relational assertions.

The datasets used for the evaluation consist of two disjoint sets of facets manually annotated with predicates from DBpedia and YAGO, respectively. In the case of DBpedia, we found that there exists more than one predicate that is similar to the facet, while in the case of YAGO there is only one similar predicate for each facet. Statistics about these two gold standards are summarized in Table 1.

DBpedia Gold Standard. Following the methodology proposed for the creation of gold standards for table annotation tasks [22, 36, 43], we manually collected 39 facets from the web from different domains: 32 facets were collected from Wikipedia (non infobox) table columns, and 7 were collected from IMDB, eCommerce, and sports statistics websites. For each facet, we selected from DBpedia all the predicates whose objects match *at least one* facet value, without considering facet domain matching. We presented the gold standard to a total of 10 human annotators, who rated the similarity of each predicate using a Likert scale from 0 to 2 (0 = *incorrect*, 1 = *correct*, 2 = *accurate*). We presented only a portion of the gold standard to each annotator in order to reduce the cognitive workload, ensuring that each predicate was rated by at least 3 annotators. We aggregated all the judgements, manually resolving inconsistencies. We report an average correlation of 0.42 between annotators using Spearman's rank correlation coefficient, because we have a reasonably large sample size. We then partitioned the gold standard into two disjoint sets of facets: **dbpedia-numbers** (i.e., facets whose values are digits) and **dbpedia-entities** (i.e., facets whose values are strings).

YAGO Gold Standard. The gold standard for the YAGO dataset has been used to evaluate several table annotation algorithms [17] and consists of tables where cells have been annotated with entities, columns with types and column pairs with predicates. We derived 90 facets from the same number of table columns that are involved in at least one relation annotation, considering the lexicalization of the annotated type for the corresponding subject columns to be the facet domain, when present. Among these 90 facets, 7 had no type annotation for the corresponding relation's subject column and thus were discarded. We ended up with 83 annotated facets. There are no numeric facets. We then created two versions of the YAGO gold standard. Within the **yago-explicit** gold standard we used the local name of types (extracted from the type URI) as facet domains (e.g., "wikicategory-American-science-fiction-novels"). By considering the local name of types, we minimize ambiguity in the facet domains. In the **yago-ambiguous** gold standard, we used more general (and thus, ambiguous), lexicalizations for facet domains, such as "novels".

5.2 Algorithms

Most of the state-of-the-art table annotation approaches have been compared with the *majority voting* based model [17, 26, 36] (MAJ). Given f, the majority based approach selects all the predicates that match f and ranks them by frequency. The *maximum likelihood* based model [36] (MAX) is based on the application of the maximum likelihood hypothesis. Given a facet f, the most similar predicate is the one that maximizes the conditional probability of occurrence of the predicate given the facet values V^f, that is,

$$ml(q, f) = K_s \prod_{v \in V^f} \frac{\Pr[\, q \mid v \,]}{\Pr[\, q \,]}, \tag{12}$$

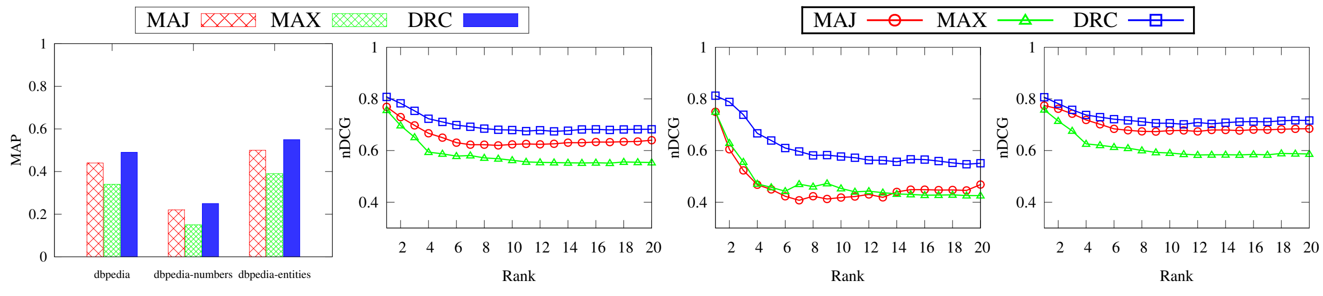
where K_s is a normalization parameter that is chosen so that $\sum_q ml(q, f) = 1$. $\Pr[\, q \mid v \,]$ and $\Pr[\, q \,]$ are computed over the $\mathcal{KB}$ using the same parameters but predicate frequency instead of the original scoring function, which was specifically defined for a database of mined relations. We refer to the original paper [36] and our code repository for further details.

The approaches based on majority vote and maximum likelihood used in table annotation leverage the quasi-relational structure of a table. When annotating a pair of columns, they match values in the first column to predicate subjects and values in the second column to predicate objects. For a fair comparison, we compare our predicate ranking function, the *domain and range comparison* function, henceforth denoted by DRC, with the baselines applied to the extensions Δ_q^f induced by the facets, computed by matching both the facet domain and the facet values to the relational assertions.

[3] http://lucene.apache.org/

[4] https://github.com/rporrini/facet-annotation

[5] We use an old version of YAGO because the state-of-the-art annotated gold standard uses this version [17].



(a) Whole DBpedia dataset. (b) dbpedia. (c) dbpedia-numbers. (d) dbpedia-entities.

Figure 4: Evaluation of DRC over the DBpedia dataset.

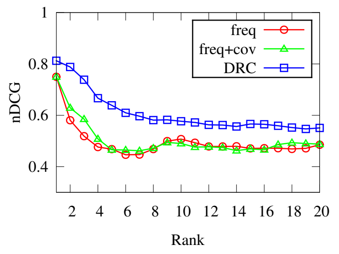


Figure 5: The effect of the *specificity* score on the dbpedia-numbers dataset.

5.3 Evaluation Measures

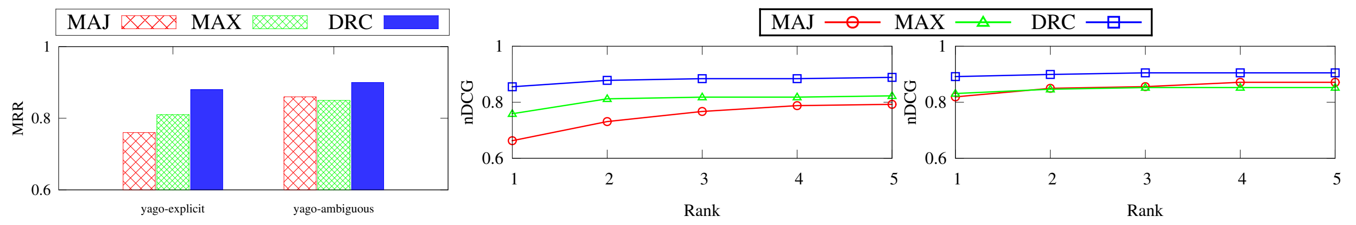
In our experiments we compare the effectiveness of DRC, MAJ, and MAX using three different measures: Normalized Discounted Cumulative Gain (nDCG), Mean Average Precision (MAP), and Mean Reciprocal Rank (MRR). The nDCG measure compares the ranking of predicates established by one algorithm with the ideal ranking derived by ratings specified by human annotators. Intuitively, a higher nDCG value corresponds to a better agreement between the results proposed by the facet annotation approach and an ideal ordering obtained from human annotators. nDCG provides a fine-grained measure of the effectiveness of DRC and the baselines, at different ranks, for both the DBpedia and YAGO datasets.

The MAP measure is computed by averaging the precision calculated at the rank of each *correct* or *accurate* predicate in P (as judged by human annotators) for each facet and finally averaged over all the facets. MAP takes into account the fact that there exists more than one similar predicate for each facet, and provides a single valued, coarse-grained measure of effectiveness. We use it for comparison with the DBpedia datasets. The MRR measure is an adaptation of the MAP measure to facets for which there exists only one predicate. MRR is defined as the multiplicative inverse of the rank of the *accurate* predicate (if present, 0 otherwise), averaged over all the facets. Intuitively, MRR captures to which extent a high rank is given by the algorithm to the (only) similar predicate. We use it for comparison with the YAGO gold standard.

5.4 Experiments with DBpedia

We compare DRC with the baselines with respect to MAP computed over the top 20 most similar predicates. We do not make a distinction between predicates judged to be *correct* or *accurate* by human annotators, because we want to ensure that DRC is able to discriminate between similar and not similar predicates in a Boolean fashion. The experimental results are depicted in Figure 4a. DRC consistently achieves better effectiveness in terms of MAP over all the DBPedia datasets. Figure 4b gives an overview of the nDCG obtained at different ranks. DRC is more effective than MAJ and MAX in capturing the similarity at all ranks from 1 to 20, achieving a maximum nDCG of 0.81 (at rank 1), compared to 0.77 of MAJ and 0.76 of MAX. The interesting observation is that the more naive MAJ baseline performs better than the MAX baseline, at all ranks. These results are in agreement with those provided by the authors of MAX [36]. In their work, they composed their approach with the majority based approach (MAJ) to achieve better results. However, both MAJ and MAX are less effective than DRC in capturing the semantic similarity between facets and DBpedia predicates. Our interpretation for this behavior is that the baselines are less robust to ambiguity introduced by the imbalanced specification of facets, which potentially results into a high number of false positive matches between the facet and predicate instances. The baselines, which are not explicitly designed to cope with this distinctive characteristic of facets, are biased towards predicates whose extension induced by the input facet has a higher cardinality, even if it contains false positive matches. In contrast, DRC is less sensitive to false positive matches as frequency is weighted by the similarity between the facet domain and the subject types. For example, the top ranked predicate computed by DRC for the facet $\langle Airports, ?p, \{Milan, Rome, \ldots, Turin\}\rangle$ is *cityServed*, which is ranked only in fourth place by both MAJ and MAX.

The observation about the robustness of DRC on the **dbpedia-numbers** dataset with respect to facet ambiguity is confirmed by nDCG (see Figure 4c). The semantics of numerical values is intrinsically ambiguous and is highly context dependent, as exemplified by the facet $\langle BasketballPlayers, ?p, \{1948, 1949, \ldots, 2012\}\rangle$. This facet has been extracted from a web page that lists NBA basketball players by their draft year. The semantics of this facet is ambiguous because, besides the draft year (i.e., the year where a player starts playing professionally), it may represent, for example, the birth or retirement year. However, the draft year is one of the most specific predicates for basketball players, while birth or retirement year characterize people or athletes in general. In this setting, the specificity scores implemented by DRC are able to boost predicates such as *draftYear*, which is used in few categories of sportspeople, over more generic ones such as *birthDate*. Instead, MAX and MAJ, which do not capture the specificity of predicates, assign a higher rank to predicates such as *birthDate* or *deathDate*.



(a) Whole YAGO dataset. (b) yago-explicit. (c) yago-ambiguous.

Figure 6: Evaluation of DRC over the YAGO dataset.

The effect of the *specificity* score on the performance of DRC on the **dbpedia-numbers** dataset is shown in Figure 5. We compare three different versions of DRC in terms of nDCG: the first one uses only the frequency score, *freq*, the second one uses both the frequency and the coverage score, *freq+cov*, while the third one is the complete DRC ranking function. Experimental results highlight the importance of capturing the specificity of predicates. The specificity score is crucial to providing high quality annotations for numeric facets. We also study the effect of the specificity score over the **dbpedia-entities** dataset and found that it provides an observable, though slight, improvement of the effectiveness. This was expected, as strings are less ambiguous than numerical values. We report a higher nDCG at all ranks for DRC compared with *freq* and *freq+cov* scores, with an average improvement of around 4.5% and 5%, respectively.

5.5 Experiments with YAGO

We compare DRC with the baselines with respect to MRR computed over the top 5 similar predicates of the YAGO dataset. Insights from the previous experiments are further confirmed: the most similar predicate for facets is generally ranked higher by DRC when compared with the baselines, as shown in Figure 6a. We also compare DRC with the baselines in terms nDGC (Figures 6b and 6c). Our approach is able to provide a better characterization of the similarity between a facet and predicates also in the presence of a single, well defined type hierarchy and a much smaller number of possibly similar predicates (89 vs 53195 of the DBpedia dataset), making it suitable also for the annotation of facets with precision oriented $\mathcal{KB}$s, such as domain specific $\mathcal{KB}$s. DRC achieves better results at each rank. We observe that the most noticeable improvement over the baselines is obtained at rank 1. This is crucial in order to support a fully automatic facet annotation process. An example of such capability is provided by the facet ⟨*Actors*, ?*p*, {*Blondie Hits the Jackpot, Charlie Chaplin Cavalcade*, . . . , *Trouble Chasers*}⟩. For this facet, the top ranked predicate computed by DRC is *actedIn*, which is ranked at the second position (after *created*) by MAJ and ranked at the third position (after *livesIn* and *created*) by MAX.

We have also experimentally observed that the specificity score have little impact on the effectiveness of DRC on the YAGO dataset (we omit the plots for lack of space). This can be explained by two observations: (1) The YAGO dataset includes *no* numeric facets, and thus we expected the specificity score to have little impact on quality in this particular dataset, and (2) The type graph provided by YAGO is deeper and more specialized than the DBpedia type graphs. As a result, the subject and object type sets extracted from the extension of predicates are more sparse than the ones that are extracted from extensions induced by facets, thus making the specificity score less discriminative.

6 CONCLUSIONS

Facet data is omnipresent on the web. In this paper we address the problem of annotating facets with a predicate from a Knowledge Base. Our approach is capable of handling the intrinsic ambiguity of facets by annotating them with the most specific predicates with respect to the facet domain. Experiments conducted by annotating facets with two large Knowledge Bases (DBpedia and YAGO) confirm the effectiveness of our approach. Using our facet annotation approach, we can assist domain experts in charge of managing facets in large and dynamic data spaces in the task of labeling and lifting facets into a semantic web framework, that is, of transforming lexical data into semantic-rich data.

We will investigate further this work in the context of geospatial healthcare applications taking advantage of earlier results [4]. We also foresee at least two more directions for future work. The first one consists of further refining our definition of the specificity of a relation with respect to a facet domain, when dealing with Knowledge Bases with a deep subtype hierarchy. Another interesting extension is the adaptation of our approach to solving relation annotation problems in particular constrained settings. For example, when the subjects of a relation cannot be matched with a reference Knowledge Base, the relation annotation problem becomes similar to the facet annotation problem addressed in this paper. Thus, we plan to incorporate our approach in ASIA, a table annotation tool that we are developing in the context of the EU funded projects EW-Shopp[6] and EuBusinessGraph.[7]

ACKNOWLEDGMENTS

The research at UIC was partially funded by NSF awards CNS-1646395, III-1618126, and CCF-1331800, and by a Bill & Melinda Gates Foundation Grand Challenges Explorations grant. The research at UNIMIB was partially funded by the EU H2020 projects EW-Shopp - Grant n. 732590, and EuBusinessGraph - Grant n. 732003. We thank the reviewers for their detailed and insightful comments, which contributed to the final version of the paper.

[6]www.ew-shopp.eu
[7]www.eubusinessgraph.eu

REFERENCES

[1] Z. Abedjan, J. Lorey, and F. Naumann. Reconciling Ontologies and the Web of Data. In *CIKM*, pages 1532–1536, 2012.

[2] B. Adida, M. Birbeck, S. McCarron, and I. Herman. RDFa Core 1.1 - Third Edition. W3C Recommendation, March 2015.

[3] K. D. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: A collaboratively created graph database for structuring human knowledge. In *SIGMOD*, pages 1247–1250, 2008.

[4] I. F. Cruz, V. R. Ganesh, and S. I. Mirrezaei. Semantic Extraction of Geographic Data from Web Tables for Big Data Integration. In *ACM SIGSPATIAL Workshop on Geographic Information Retrieval (GIR)*, pages 19–26, 2013.

[5] I. F. Cruz, F. Palandri Antonelli, and C. Stroe. AgreementMaker: Efficient Matching for Large Real-World Schemas and Ontologies. *PVLDB*, 2(2):1586–1589, 2009.

[6] W. Dakka and P. G. Ipeirotis. Automatic Extraction of Useful Facet Hierarchies from Text Databases. In *ICDE*, pages 466–475, 2008.

[7] D. Dash, J. Rao, N. Megiddo, A. Ailamaki, and G. M. Lohman. Dynamic Faceted Search for Discovery-driven Analysis. In *CIKM*, pages 3–12, 2008.

[8] Z. Dou, S. Hu, Y. Luo, R. Song, and J.-R. Wen. Finding Dimensions for Queries. In *CIKM*, pages 1311–1320, 2011.

[9] V. Efthymiou, O. Hassanzadeh, M. Rodriguez-Muro, and V. Christophides. *Matching Web Tables with Knowledge Base Entities: From Entity Lookups to Entity Embeddings*, pages 260–277. Springer, 2017.

[10] D. Faria, C. Pesquita, E. Santos, M. Palmonari, I. F. Cruz, and F. M. Couto. The AgreementMakerLight Ontology Matching System. In *ODBASE*, volume 8185 of *LNCS*, pages 527–541. Springer, 2013.

[11] M. Hepp. E-Business Vocabularies As a Moving Target: Quantifying the Conceptual Dynamics in Domains. In *EKAW*, pages 388–403. Springer, 2008.

[12] W. Kong and J. Allan. Extracting Query Facets from Search Results. In *SIGIR*, pages 93–102, 2013.

[13] W. Kong and J. Allan. Precision-Oriented Query Facet Extraction. In *CIKM*, pages 1433–1442. ACM, 2016.

[14] D. Kontokostas. Making Sense out of the Wikipedia Categories (GSoC). http://blog.dbpedia.org/?p=74, 2013. Accessed: 2016-02-12.

[15] K. La Barre. Facet analysis. *Annual Review of Information Science and Technology*, 44(1):243–284, 2010.

[16] 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*, 2014.

[17] G. Limaye, S. Sarawagi, and S. Chakrabarti. Annotating and Searching Web Tables Using Entities, Types and Relationships. *PVLDB*, 3(1):1338–1347, 2010.

[18] F. Manola and E. Miller. RDF Primer. W3C Recommendation, February 2004.

[19] O. Medelyan, S. Manion, J. Broekstra, A. Divoli, A.-L. Huang, and I. H. Witten. Constructing a Focused Taxonomy from a Document Collection. In *ESWC*, pages 367–381, 2013.

[20] E. Moreau, F. Yvon, and O. Cappé. Robust Similarity Measures for Named Entities Matching. In *COLING*, pages 593–600, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[21] A. Moro, A. Raganato, and R. Navigli. Entity Linking Meets Word Sense Disambiguation: A Unified Approach. *Transactions of the Association for Computational Linguistics*, 2:231–244, 2014.

[22] V. Mulwad, T. Finin, and A. Joshi. Semantic Message Passing for Generating Linked Data from Tables. In *ISWC (1)*, pages 363–378, 2013.

[23] M. Pham, S. Alse, C. A. Knoblock, and P. A. Szekely. Semantic Labeling: A Domain-Independent Approach. In *ISWC*, volume 9981 of *LNCS*, pages 446–462, 2016.

[24] R. Porrini, M. Palmonari, and C. Batini. Extracting Facets from Lost Fine-Grained Categorizations in Dataspaces. In *CAiSE*, pages 580–594, 2014.

[25] U. Priss. Facet-like Structures in Computer Science. *Axiomathes*, 18(2):243–255, Jun 2008.

[26] G. Quercini and C. Reynaud. Entity Discovery and Annotation in Tables. In *EDBT*, pages 693–704, 2013.

[27] D. Ritze and C. Bizer. Matching Web Tables To DBpedia - A Feature Utility Study. In *EDBT*, pages 210–221. OpenProceedings.org, 2017.

[28] D. Ritze, O. Lehmberg, and C. Bizer. Matching HTML Tables to DBpedia. In *International Conference on Web Intelligence, Mining and Semantics (WIMS*, pages 10:1–10:6. ACM, 2015.

[29] J. Schaible, T. Gottron, and A. Scherp. Survey on Common Strategies of Vocabulary Reuse in Linked Open Data Modeling. In *ESWC*, pages 457–472, 2014.

[30] W. Shen, J. Wang, P. Luo, and M. Wang. LIEGE: Link Entities in Web Lists with Knowledge Base. In *KDD*, pages 1424–1432, 2012.

[31] P. Shvaiko and J. Euzenat. Ontology Matching: State of the Art and Future Challenges. *IEEE Trans. Knowl. Data Eng.*, 25(1):158–176, 2013.

[32] B. Spahiu, R. Porrini, M. Palmonari, A. Rula, and A. Maurino. ABSTAT: Ontology-Driven Linked Data Summaries with Pattern Minimalization. In *ESWC 2016 Satellite Events, Revised Selected Papers*, volume 9989 of *LNCS*, pages 381–395, 2016.

[33] A. Stolz and M. Hepp. Adaptive Faceted Search for Product Comparison on the Web of Data. In *ICWE*, volume 9114 of *LNCS*, pages 420–429. Springer, 2015.

[34] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: A Core of Semantic Knowledge. In *WWW*, pages 697–706, 2007.

[35] D. Vandic, J.-W. Van Dam, and F. Frasincar. Faceted Product Search Powered by the Semantic Web. *Decis. Support Syst.*, 53(3):425–437, June 2012.

[36] P. Venetis, A. Y. Halevy, J. Madhavan, M. Pasca, W. Shen, F. Wu, G. Miao, and C. Wu. Recovering Semantics of Tables on the Web. *PVLDB*, 4(9):528–538, 2011.

[37] J. Wang, H. Wang, Z. Wang, and K. Q. Zhu. Understanding Tables on the Web. In *ER*, pages 141–155, 2012.

[38] B. Wei, J. Liu, J. Ma, Q. Zheng, W. Zhang, and B. Feng. DFT-Extractor: A System to Extract Domain-specific Faceted Taxonomies from Wikipedia. In *WWW (Companion Volume)*, pages 277–280, 2013.

[39] B. Wei, J. Liu, Q. Zheng, W. Zhang, X. Fu, and B. Feng. A Survey of Faceted Search. *J. Web Eng.*, 12(1&2):41–64, 2013.

[40] B. Wei, J. Liu, Q. Zheng, W. Zhang, X. Fu, and B. Feng. A Survey of Faceted Search. *J. Web Eng.*, pages 41–64, 2013.

[41] W. Wu, H. Li, H. Wang, and K. Q. Zhu. Probase: A Probabilistic Taxonomy for Text Understanding. In *SIGMOD*, pages 481–492, 2012.

[42] N. Yan, C. Li, S. B. Roy, R. Ramegowda, and G. Das. Facetedpedia: Enabling Query-dependent Faceted Search for Wikipedia. In *CIKM*, pages 1927–1928, 2010.

[43] Z. Zhang. Effective and efficient semantic table interpretation using tableminer$^+$. *Semantic Web*, 8(6):921–957, 2017.

[44] S. Zwicklbauer, C. Einsiedler, M. Granitzer, and C. Seifert. Towards Disambiguating Web Tables. In *ISWC Posters & Demonstrations*, pages 205–208, 2013.