Linking FRBR Entities to LOD through Semantic Matching

Naimdjon Takhirov, Fabien Duchateau* and Trond Aalberg

Norwegian University of Science and Technology, NO-7491 Trondheim, Norway,
{takhirov,fabiend,trondaal}@idi.ntnu.no

Abstract. In this paper, we present an approach to automatically link FRBR works identified in metadata to the corresponding entity in Linked Open Data resources. The main contribution is a basis for semantic enrichment and verification of works identified in existing metadata. Through experiments, we demonstrate that FRBR works can be identified in the LOD cloud, which provides a solid ground for further work.

1 Introduction

Metadata related to cultural items such as movies, books and music is a valuable resource that is currently exploited in many applications and services based on mashup and linked data. Semantic Web technologies can be used to expose and interpret the meaning of the data on the Web, publicly available API's enable third parties to develop innovative services for existing data, and new knowledge can be created by linking related and complementary data from different sources.

The use of conceptual domain models is an important part of this environment as they define the universe of discourse and facilitates the proper semantic integration of the information within a domain. The Functional Requirements for Bibliographic Records (FRBR) [10] is a conceptual model that increasingly is being recognized as the common domain model for cultural items, and one of the main challenges deals with the interpretation or conversion of existing data into FRBR-based representations. The proposed semi-automatic approaches [8, 13, 1] have been designed to fulfill this goal but they mainly focus on converting bibliographic records found in library catalogs.

The Linked Open Data (LOD) vision [2] and the increasing demand for semantic aware data has strengthened the interest in FRBR. In this paper, we present a solution for linking the FRBR *works* that can be identified in metadata to its corresponding LOD entity. The main motivation is to bridge the gap between metadata that mainly identify such entities through implicit descriptions and the explicit representation of these entities that we can find in LOD resources such as DBpedia and OpenCyc. The benefit of this solution is the ability to semantically enrich existing metadata with attributes and relationships discovered when linked to LOD, but we will also argue that our approach

* The author carried out this work during the tenure of an ERCIM "Alain Bensoussan" Fellowship Programme.

can serve as a basis for verification purposes, specifically for tools which automatically convert legacy data into FRBR. We demonstrate that our approach is effective by performing a set of experiments with Amazon product data.

2 Preliminaries

2.1 Functional Requirements for Bibliographic Records (FRBR)

The FRBR is a conceptual model of the bibliographic universe published around a decade ago [10]. It models intellectual and artistic endeavor in multiple levels of abstraction: **work** (e.g. *The Two Towers* by J.R.R. Tolkien) , **expression** (e.g., *To Tårn*, a Norwegian translation of that work), **manifestation** (e.g., a paperback format of this expression published by Mariner Books in 2005), and **item** (e.g. a physical book). A **person (or corporate body)** in the FRBR model, is an individual responsible for the creation or realization of a work (e.g., as an author, an illustrator, a translator, etc.). Additionally, the FRBR model provides **a set of relationships** between entities beyond the basic relationships.

2.2 Linking Open Data

The Semantic Web is a technology to support the web of data (contrary to the current web of documents) by relying on semantic and linked data, models (e.g., RDF), query languages (e.g., SPARQL), inference system and applications. More specifically, **Linked Open Data** (LOD) encompasses a vision in which all data is globally accessible and interconnected, thus making it more valuable. All LOD entities such as subjects or properties, are identified by a unique Uniform Resource Identifiers (URI). The LOD cloud refers to interconnected data sources, such as DBpedia, Freebase or OpenCyc, which can be seen as the foundation of the LOD vision. With the emergence of this initiative, an increasing number of data sets is published as linked data. The basic principles of publishing on the LOD cloud is the use of RDF as a data model and RDF links to interlink data from diverse data sources. The primary motivation for publishing data in a LOD cloud is it provides a basis for semantic reuse and integration of data from diverse sources. To reach this goal, data should be represented in a well-defined structure [2].

3 Related Work

Our approach consists in linking a FRBR work to its corresponding LOD entity, and it lies in the intersection of two domains. The former is **entity search**, since we want to discover equivalent entities based on their information. However, one of the entities we intend to match is a semantic entity in the LOD cloud, which deals with **entity ranking**. The rest of this section provides more details about these two research domains.

The *entity search* problem, also known as *record linkage* or *entity resolution*, is a crucial task for data integration or data cleaning [7]. It mainly aims at identifying entities (objects or data instances) which represent the same real-world entity. Contrary to existing approaches, which are designed to match entities represented in a relational framework [11], we apply entity search to RDF entities. Besides, most of them are based on machine learning techniques, and require training data. Another major difference deals with the quality of the data sources: in our context, we can assume that the data from the LOD cloud does not contain many errors for a given entity.

On the Web, a similar task, called *entity ranking*, involves the discovery of an entity's main page, contrary to traditional search engines which propose documents mentioning a given entity. This task has been extensively studied and two initiatives have an entity ranking track arranged every year: *Initiative for the Evaluation of XML Retrieval* (INEX) [9] and *Text Retrieval Conference* (TREC) [17]. Most approaches which take part in these tracks are either based on information retrieval or semantic web [16, 14]. The main difference between our work and entity ranking is the availability of information. In our context, the type of the searched entity is not always specified, or with a broader topic. Conversely, the work that we want to match to a LOD entity can include useful information such as creator, year, or categories.

4 Matching FRBR Works to LOD

Linking FRBR entities to the LOD cloud is a solution with many benefits. First, it could enable the automatic enrichment of FRBR entities discovered in existing metadata with additional attributes and relationships. For instance, we may discover the relationship between a book and the screenplay that is based on the same novel by looking up the work in the LOD cloud. Secondly, the LOD cloud can be used to verify or guide the FRBR-based interpretation of existing information provided about the product, which can be misleading and ambiguous when interpreting the intellectual aspects due to a lack of semantics. For instance, when using titles and authors to identify works there can be a large number of false positives if there are many translations or adaptations of the same work. The LOD cloud can be used to verify the proper works or to single out the work entities that are of main interest to end users. In the rest of this section, we explain how we discover a relationship between a FRBR work and a LOD entity.

4.1 The Problem

We have a set of works $\mathcal{W}$ and a set of LOD entities $\mathcal{L}$. Note that the LOD entities are linked to other entities by relationships, but we do not need this feature at this stage. All works and entities have a set of attributes. Considering a work $w \in \mathcal{W}$ and a LOD entity $l \in \mathcal{L}$, we note $\mathcal{F}$ the set of attributes shared by w and l. To assess a degree of similarity between w and l, we compute similarity values between their shared attributes. For an attribute $f \in \mathcal{F}$ shared by w and l, a similarity function is defined as follows:

$$sim_f(w, l) \rightarrow [0, 1]$$

The similarity function returns values between 0 and 1, 0 indicating attribute f of w and l is completely dissimilar and 1 if the attribute f is identical for both w and l. The amount of data available on LOD is very large, and even the querying of only one data source can be time consuming. Thus, the goal is first to reduce the search space by obtaining a subset of LOD entities, a process called *Blocking*. Then, we can apply fine-grained *matching* techniques on these entities to compute their degree of similarity with the given FRBR work. This process outputs a ranking for these LOD entities according to their similarity degree. Figure 1 sums up our approach for matching FRBR to LOD. As a running example, we use a work entitled *The fellowship of the ring (LOTR)*. It includes the following (incomplete) list of attributes: *novel* as type, *JRR Tolkien* as creator, *science fiction & fantasy* for categories and no creation date.

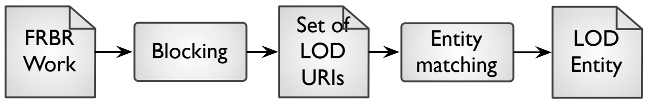


Fig. 1. Workflow of Entity Matching.

4.2 Blocking

In entity matching, the large amount of entities implies to have a method for reducing the search space. For instance, if an entity has a title, a simple blocking method could be the matching of entities that share at least a common word in their titles. In our context, this blocking process is required for two reasons. First, we query remote services with their potential issues (e.g., network overload, query limitations). Secondly, the set of entities is very large: more than 3.4 millions entities for DBpedia[1], 12 millions for Freebase[2] and thousands of entities for OpenCyc[3], which are only a few of the data sources from the LOD cloud. As a consequence, we need to have a heuristic to retrieve only a subset of entities against which we apply matching techniques. The following techniques can be used to search for a LOD entity:

- Knowing or generating the correct URI of the entity, which cannot be applied in our context;
- Querying SPARQL endpoints;
- Querying a Lookup engine.

Both SPARQL and Lookup queries can return a set of LOD entities that match the search query. Thus, we reduce the search space by using these services, since they return an acceptable number of results (in our case usually between 0

[1] `http://dbpedia.org/`, January 2011
[2] `http://www.freebase.com/`, January 2011
[3] `http://www.opencyc.org/`, January 2011

and 200). In order to increase the probability of obtaining the correct entity in the search results, we need to build different queries based on the information contained in the work's attributes.

We have identified three interesting attributes of a work that can be used to generate a set of queries: title, creator and type. However, these attributes cannot be used directly in the query. They need to be transformed to remove extra information, to split creator's name, or to broaden a type. The idea is to create a set of query tokens for each of these attributes. More formally, we want to obtain three sets *titles*, *creators* and *types* containing query tokens such as:

$$\begin{aligned} \textit{titles} &\rightarrow \{title, normalized\_title\} \\ \textit{creators} &\rightarrow \{creator_1, ..., creator_k\} \\ \textit{types} &\rightarrow \{type, ext\_type_1, ..., ext\_type_m\} \end{aligned}$$

The *titles* set contains the full title of the work, and a normalized title in which extra information (e.g., inside parenthesis) and useless grammatical words are removed. In other words, this normalized title only includes the most important words after a normalization process [6]. For the creators set, each creator's name is used as a query token. Finally, the *types* set contains the type of the work and its extensions. These extensions are hypernyms and synonyms from a predefined list (from Wordnet[4]), e.g., the type *novel* is extended with *print* and *book*.

Once we have produced the three sets with their query tokens, we can combine the query tokens to generate a query. Combining these tokens is required either to obtain more results or to disambiguate. For instance, the novel entitled *airport* only returns a list of airports if the type is not included in the query. So the idea is to perform all combinations of 1, 2 or 3 tokens, each token belonging to a different set, and use these combinations as queries. All results returned by each query are merged based on the unique entity URI. Note that if all individual tokens do not return any results, there is no need to send queries which include this combination. At the end of this blocking process, we obtain a set of LOD entities (represented by their URI) against which we apply refined matching techniques.

We have generated different queries for our example work dealing with *The fellowship of the ring (LOTR)*. Table 1 shows some of these queries and provides the number of results returned by a Lookup service. Here, we highlight the need for sending multiple queries. Even with a well-known artistic work such as *The fellowship of the rings*, the lookup did not return any results with the full title, hence the need to simplify this title. Similarly, a query including the normalized title and the type of the work did not provide any results, contrary to the normalized title combined with an extended type.

4.3 Entity Matching

After the blocking step, we obtain a normalized set of LOD entities, and we need to match them against our work. To fulfill this goal, we first identify which

[4] `http://wordnet.princeton.edu`, January 2011

Type of Query	**Query**	**# Returned Entities**
title	The fellowship of the ring (LOTR)	0
norm_ title	fellowship ring	5
title + creator	The fellowship of the ring (LOTR) JRR Tolkien	0
norm_ title + creator	fellowship ring JRR Tolkien	0
norm_ title + type	fellowship ring novel	0
norm_ title + ext_ type	fellowship ring book	1
norm_ title + ext_ type	fellowship ring print	0
creator + type	JRR Tolkien novel	0
creator + ext_ type	JRR Tolkien book	1

Table 1. A Subset of Generated Queries for our Work Example

shared attributes can be matched, and then we describe the similarity functions applied to these attributes. A global similarity value between a work and a LOD entity is finally computed, and filters may be used to discard some of the matched entities.

Identifying Attributes. First, we have identified the most important attributes that we can use to compare a FRBR work and a LOD entity. Although these attributes depend on the data sources we have on both sides (work and entity), five attributes are at least very common:

1. Title. In our running example, the work title has the value "The fellowship of the ring (LOTR)";
2. Type of work/entity. For instance, the work type of *The fellowship of the ring (LOTR)* is "novel" while the type of the corresponding entity is "book";
3. Creator. All artistic works have one or more creators. "J.R.R. Tolkien" is the creator of our example work;
4. Categories. They represent the genres or domains to which the artistic work belongs. *The lord of the Rings* categories may include "heroic fantasy", "Middle Earth universe" or "science fiction & fantasy";
5. Date of creation. *The fellowship of the ring (LOTR)* has been originally created in "1954".

The first three attributes are in most cases present in both work and entity. On the contrary, the last two attributes may lack in one or both data sources. Although the year of creation may be misleading, it is useful in specific cases. Dealing with the work about the movie *the lord of the rings : the return of the king*, there exist a first movie produced by Bass and Rankin in 1980 and a second one by Peter Jackson in 2003. If the creator's names are lacking or subject to mistakes, the dates could help us to disambiguate the two candidate movies. Finally, the idea is to compute the similarity for these five shared attributes of a work and an entity.

Computing Individual Similarity Values. We compute a similarity value between the same attributes of the work and the entity. However, the nature of these attributes are different: the *title* and *creator* are plain text while the *categories* are a set of words. The *type* is a word from a finite set of values while the year can have different formats. As a consequence, we need different similarity measures for matching these attributes. Schema matching and ontology alignment research fields have provided many techniques to discover similar elements in various data sources that we can apply in our context [6].

Attributes title and creator. To measure the similarity between character strings, we have selected three terminological similarity measures: Jaro Winkler, Monge Elkan and Scaled Levenshtein. Combining these similarity measures enables us to avoid the drawbacks related to one of the measure (e.g., the Levenshtein returns high similarity for small-sized strings which are very dissimilar) [4]. Given the titles t (respectively creators c) of a work w and a LOD entity l, we compute the following similarity sim_{title} (resp. sim_{creat}) as the average between the three similarity measures:

$$sim_{title}(w, l) = \frac{jaro(t_w, t_l) + monge(t_w, t_l) + leven(t_w, t_l)}{3}$$

Attribute categories. As these categories are represented by a set of strings, we define a very basic similarity function sim_{cat} between two sets. It computes the number of identical categories between the set of categories of a work w and a LOD entity l.

$$sim_{cat}(w, l) = \frac{|cat_w \cap cat_l|}{\max(|cat_w|, |cat_l|)}$$

Attribute type. The type (extracted from its manifestations for the work) is limited to predefined values such as *book*, *movie*, *novel*. As the number of values is not large, we have built a small taxonomy extracted from the Wordnet hierarchy. To compute the similarity between two types, we can therefore apply the Resnik similarity [12]. It evaluates the similarity of these types based on the concepts that subsume them in our taxonomy. Figure 2 depicts a part of our taxonomy. For instance, the similarity value between the types *book* and *novel* in our taxonomy is equal to 0.29.

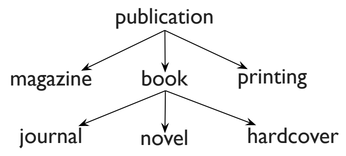


Fig. 2. A Fragment of our Taxonomy for Matching the Attributes *type*

Attribute date. The idea is to extract only the year, which is a meaningful temporal granularity for artistic works. Thus, we compare the date value with several predefined patterns to extract the year, both for the work and for the entity. If the extracted years from the work and the entity are identical, the similarity function returns 1. Else, it returns a 0 value. Back to our running example: Table 2 shows a LOD entity with each of its attribute's value. The last column indicates the similarity value for the attribute with regards to the corresponding attribute of the work (which is detailed in Section 4.1). We notice that the title and the creator are terminologically similar (similarity values around 0.8). As the work does not contain a date, the similarity value for creation date equals 0.

Attribute	LOD Property Value	FRBR Work Attribute Value	Similarity Value
Title	The Fellowship of the Ring	The fellowship of the ring (LOTR)	0.77
Type	Book	Novel	0.29
Creator	J._R._R._Tolkien	JRR Tolkien	0.81
Categories	Fantasy	science fiction & fantasy	0.00
Date	1954-07-24	-	0.00

Table 2. Attributes and Similarity Values of the Entity *The_Fellowship_of_the_Ring*

Computing a Global Similarity Value. From these attribute similarity values, we are able to derive a global similarity value. We have chosen a weighted average function to aggregate the values of all individual similarities. The global similarity value is computed with the following formula, where w is the work and l is the LOD entity, i.e., $w \in \mathcal{W}$ and $l \in \mathcal{L}$:

$$\text{sim}(w,l) = \frac{\alpha sim_{title}(w,l) + \beta sim_{type}(w,l) + \gamma sim_{creat}(w,l) + \delta sim_{cat}(w,l) + \zeta sim_{year}(w,l)}{\alpha + \beta + \gamma + \delta + \zeta}$$

In our running example, the DBpedia entity *The_Fellowship_of_the_Ring* and the work have a global similarity value equal to 0.37. As a comparison, the DBpedia entity related to the movie *The_Fellowship_of_the_Ring* obtains a similarity value of 0.22.

Filtering the Candidate Matches. Similarly to many matching approaches, we can filter the candidate matches by selecting those with a similarity value above a given threshold. A correct tuning of this threshold is crucial since it directly impacts the quality. Note that a constraint filter could also be applied in our context: if the work deals with a *movie*, then all LOD entities with a *book* type should be discarded. We demonstrate in Section 5 the impact of a threshold filter. As a result, all remaining entities discovered for a work can be ranked given their similarity values. Similarly to most matching approaches, the user still needs to decide if one of the proposed entities corresponds to the work. However, we show in our experiment results that our approach often ranks the correct entity at the first position.

4.4 Discussion

First, the LOD cloud is incomplete, i.e., it does not contain all entities that correspond to the FRBR works. Yet, our blocking process may return several

LOD entities, hence the need to compute their degree of similarity with the work. On the contrary, there may be no LOD entity returned by the blocking process. This does not mean that the LOD entity corresponding to the work does not exist. The benefits of our approach are threefold. First, it enables the verification of FRBRized data. But it can also be used to add new entities in the LOD cloud when a work has no corresponding entity. Specialized knowledge bases already use this mechanism to automatically create entities for a generic knowledge base, often with incomplete information. The last benefit is the semantic enrichment. Once a LOD entity is validated as correct for a given work, we can enrich this work by adding attributes extracted from the LOD entity. In addition, we can infer some simple relationships. For instance, we can link our work *The fellowship of the ring (LOTR)* with the other works of the trilogy thanks to the DBpedia property *dbpprop:books-of*. The attribute set we have chosen can be easily mapped to the attribute sets of different knowledge bases. If the number of attributes are too large for a manual mapping, tools such as Falcons [3] enable us to detect attributes that represent the same concept.

5 Experiments

In our experiments, a list of the 80 best selling fiction authors from Wikipedia[5] was used to query for product descriptions on Amazon bookstore (using the Amazon Product Advertising API[6]). These product descriptions have been FRBRized using the FRBRPedia approach [5, 15], thus resulting in the generation of 684 distinct FRBR works. The challenge is to discover a correct entity on the LOD cloud for each of these works. In this experiment, we have chosen DBpedia as our main source of corresponding entities. Note that our approach is not limited to this knowledge base and that we could have used another source such as Freebase or OpenCyc. However, DBpedia is regarded as the center of this LOD cloud as it has the largest number of connections to other data sources.

5.1 Experimental Protocol

To reduce the search space, we could use SPARQL or Lookup queries. However, we have noticed that SPARQL queries are time-consuming with multiple constraints involving free-text strings. Thus, we use the *Lookup* API provided by DBpedia[7] to obtain a subset of DBpedia URIs representing entities that could correspond to the work using various queries as explained in Section 4.2. We used this reduced set of URIs as candidate matches for a given work. Matching techniques presented in Section 4.3 have been applied between the attributes of a work and those of the candidate matches. During this initial set of experiments, the global similarity value is computed with all weights equal to 1, which means that we do not promote any attribute. Similarly, we did not apply any filter to

[5] `http://j.mp/fiction_authors`, January 2011
[6] `http://j.mp/amznProductAPI`, v.2010-10-01
[7] `http://lookup.dbpedia.org`, December 2010

this global similarity value (i.e., the threshold value is tuned to 0). The blocking and matching processes for the 684 works were performed in 10 to 12 minutes (without caching). Finally, we ranked the candidate matches for each work. For half of the 684 works, we were not able to discover any DBpedia entity. The remaining 343 works have at least one DBpedia entity. We presented the top-3 candidate matches for manual validation. This validation step was performed by 8 different people from our research group, which means that they have to check all proposed LOD entities and decide whether it corresponds to the given work (based on available information, such as creators, titles, summaries, or types). If none of the proposed entities is correct, participants validated the work by manually searching DBpedia. This manual validation forms a ground truth for the collection, based on which compute quality results of our approach.

5.2 Quality of Results

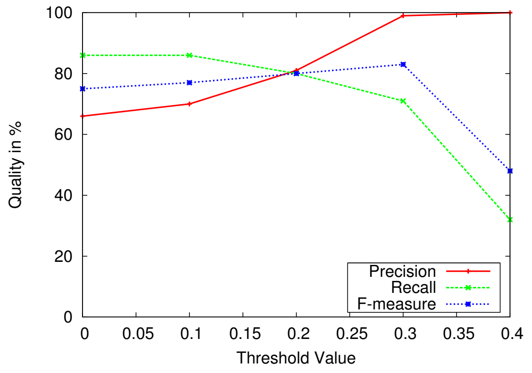
To assess the quality, we study the impact of the three parameters, namely the top-K matches, the threshold filter and the tuning of the weights in the global similarity value. Let us begin with the **top-K**. The number of correct discovered matches (true positives) at top-1, top-2 and top-3 are shown in Table 3. Most of the correct matches (189) are ranked at the top. At top-3, we only discover 12 more entities. Thus, our approach is able to present to the user the correct DBPedia entity at the top of the ranking.

	Top-1	**Top-2**	**Top-3**
Number of True-Positives	**189**	197	201

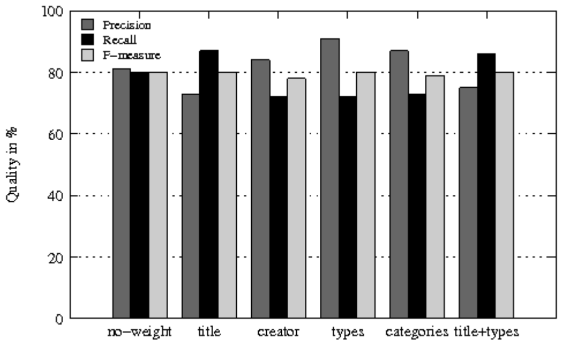
Table 3. Number of True Positives by Top-k

The following experiment deals with **the impact of the threshold filter** (see Section 4.3). We compute the quality in terms of precision, recall and f-measure, as discussed in [6]. Precision represents the percentage of correct matches among those discovered at top-k while recall stands for the percentage of correct matches at discovered by our approach w.r.t. the total number of correct matches. F-measure is a tradeoff between precision and recall. Figure 3(a) depicts the quality obtained by our approach at top-1 when the threshold value for filtering matches varies. Without any threshold (value equal to 0), the f-measure reaches 76%. The recall value is around 85%, which means that we do not miss too many correct matches. However, we still discover many incorrect matches (precision at 66%). When we increase this threshold value, then the precision value increases while the recall score decreases. A balanced f-measure value (80%) is achieved for a 0.2 threshold. With higher threshold values, we are able to reach 100% precision, but at the expense of recall (71%). A peak is reached when the threshold is in the range of 0.3 and 0.4.

In the last experiment we study **the impact of weights in the global similarity function** (see Section 4.3). We have previously shown that a threshold value equal to 0.2 provides balanced results between precision and recall, so we have used this value in this experiment. Figure 3(b) depicts the top-1 quality when we apply a higher weight to one or more individual similarity measures. For instance, the precision, recall and f-measure values respectively equal 73%, 87% and 80% with a weight on the *title* similarity measure. We notice two interesting points. The former deals with a weight on the title which enables the promotion of recall (87%). Indeed, when a work matches a LOD entity, their titles are often similar. But this high title similarity is limited by the other individual similarity functions. Thus, tuning the weight of the title allows us to discover more correct matches, but at the expense of precision. The latter point is the weight applied to types which promotes precision (91%). Indeed, a hard constraint on the types avoids the discovery of matches involving a work and a LOD entity with different types (such as movie and book).



(a) Quality Results (precision, recall, f-measure) w.r.t. a Threshold Filter



(b) Quality Results w.r.t. the Weights of Individual Similarities

5.3 Discussion

Our first observation is concerned with the quality of the input data and of their conversion into FRBR. This process obviously has an impact when linking to DBpedia. For instance, the search results from Amazon need to be cleaned. Indeed, they can contain dirty data such as “The Lord of the Rings: The Return of the King (Widescreen Edition)” and unrelated products (given the query). We also faced several issues with the DBpedia knowledge base. A lack of information in the DBpedia entity leads to no match, a case which may occur for DBpedia entities which are automatically created from other knowledge bases but with incomplete attributes. Similarly, an entity page can redirect to a related entity page (e.g. author, concept, event). As for the experiment results, our global similarity measure is reliable since most correct matches are discovered at top-1 and we miss only a few entities. Furthermore, the approach is flexible with the weights/threshold, which both enable users to promote either precision or recall.

6 Conclusion

In this paper, we have presented a generic framework to link a FRBR work to its corresponding LOD entity, using a query builder as blocking process and refined similarity measures as matching process. As a result of experiments with Amazon, we have successfully discovered the correct DBpedia entity for most products. Thus, our approach is a basis both for verification purposes and for semantic enrichment. As for future work, the framework can be integrated with other LOD data sources (e.g. *Freebase*, *LastFM*). Indeed, linking a work to a specialized database (e.g. *MusicBrainz* for musical work) may provide a higher probability for discovering the correct match than a general knowledge base.

References

1. T. Aalberg, F. B. Haugen, and O. Husby. A tool for converting from marc to frbr. In *Proc. of ECDL*, pages 453–456. Springer, 2006.
2. C. Bizer, T. Heath, and T. Berners-Lee. Linked Data - The Story So Far. *International Journal on Semantic Web and Information Systems*, 2009.
3. G. Cheng, W. Ge, and Y. Qu. Falcons: searching and browsing entities on the semantic web. In *Proc. of WWW*, pages 1101–1102, 2008.
4. W. Cohen, P. Ravikumar, and S. Fienberg. A comparison of string distance metrics for name-matching tasks. In *Proc. of IIWeb*, 2003.
5. F. Duchateau, N. Takhirov, and T. Aalberg. FRBRPedia: a Tool for FRBRizing Web products and Linking FRBR Entities to DBpedia. In *Proc. of JCDL*, 2011.
6. J. Euzenat and P. Shvaiko. *Ontology matching*. Springer, 2007.
7. I. P. Fellegi and A. B. Sunter. A theory for record linkage. *Journal of the American Statistical Association*, 64:1183–1210, 1969.
8. N. Freire, J. L. Borbinha, and P. Calado. Identification of FRBR Works Within Bibliographic Databases: An Experiment with UNIMARC and Duplicate Detection Techniques. In *Proc. of ICADL*. Springer, 2007.
9. S. Geva, J. Kamps, and A. Trotman. Focused retrieval and evaluation, inex. In *INEX*, volume 6203. Springer, 2010.
10. IFLA Study Group on the FRBR. Functional requirements for bibliographic records, final report. *UBCIM publications ; new series*, 19(1), 1998.
11. H. Köpcke and E. Rahm. Frameworks for entity matching: A comparison. *Data Knowl. Eng.*, 69:197–210, February 2010.
12. P. Resnik. Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language. *Journal of Artificial Intelligence Research*, 11:95–130, 1999.
13. J. Riley. Enhancing interoperability of frbr-based metadata. In *Dublin Core and Metadata Applications*, 2010.
14. H. Rode, P. Serdyukov, and D. Hiemstra. Combining document- and paragraph-based entity ranking. In *SIGIR*, pages 851–852, 2008.
15. N. Takhirov, F. Duchateau, and T. Aalberg. Supporting frbrization of web product descriptions. In *Proc. of TPDL*. Springer, 2011.
16. A.-M. Vercoustre, J. A. Thom, and J. Pehcevski. Entity ranking in wikipedia. In *SAC*, pages 1101–1106, 2008.
17. E. Voorhees and D. Harman. Trec experiment and evaluation in information retrieval. MIT Press, USA, 2005.