Content-Based Recommendations via DBpedia and Freebase: A Case Study in the Music Domain

Phuong T. Nguyen, Paolo Tomeo,
Tommaso Di Noia(✉), and Eugenio Di Sciascio

SisInf Lab, Polytechnic University of Bari, Via Orabona 4, 70125 Bari, Italy
{phuong.nguyen,paolo.tomeo,tommaso.dinoia,eugenio.disciascio}@poliba.it

Abstract. The `Web of Data` has been introduced as a novel scheme for imposing structured data on the Web. This renders data easily understandable by human beings and seamlessly processable by machines at the same time. The recent boom in `Linked Data` facilitates a new stream of data-intensive applications that leverage the knowledge available in semantic datasets such as `DBpedia` and `Freebase`. These latter are well known encyclopedic collections of data that can be used to feed a content-based recommender system. In this paper we investigate how the choice of one of the two datasets may influence the performance of a recommendation engine not only in terms of precision of the results but also in terms of their diversity and novelty. We tested four different recommendation approaches exploiting both `DBpedia` and `Freebase` in the music domain.

Keywords: Linked open data · Quality assessment · Semantic similarity · Content-based recommender systems

1 Introduction

The `Linked Open Data` cloud has been launched in an effort to transform structured data into first class citizens in the Web thus moving it towards the so called `Web of Data`. The data published as `Linked Data` (`LD`) by means of `RDF` covers a wide range of knowledge, including life science, environment, industry, entertainment, to name a few. The new data platform paves the way for several fresh applications but the proliferation of `LD` is overshadowed by the fact that the quality of the newly uploaded data is yet to be thoroughly verified [22] and that the selection of the dataset may heavily influence the performance of an `LD`-based tool. Among all possible data intensive applications, recommender systems are gaining momentum to potentially profiting from the knowledge encoded in `LD` datasets. As background data is of crucial importance to recommender systems, one should consider the suitability of a dataset when designing a recommender system since it may depend on the type of tasks as well as the recommendation algorithm. A reasonable combination of the underlying

© Springer International Publishing Switzerland 2015
M. Arenas et al. (Eds.): ISWC 2015, Part I, LNCS 9366, pp. 605–621, 2015.
DOI: 10.1007/978-3-319-25007-6_35

data and recommendation approach might contribute towards a great difference in performance. This motivates us to perform an investigation on the adequacy of a dataset when adopting a recommendation strategy. In this paper we evaluate the fitness for use of LD sources to feed a pure content-based recommender system [7] and in particular we examine the suitability of two encyclopedic data sources namely DBpedia[1] and Freebase[2] for musical artists recommendation tasks. As the input for the calculation we exploit similarity values computed by four different feature-based semantic similarity metrics. The values are used to find similarities between items and eventually to produce the final recommendation list. Our experimental evaluations are conducted by using the well-known dataset Last.fm for musical artists recommendation[3]. To study the fitness for use of the data sources to recommendation tasks, we conducted an offline evaluation and we analyzed three different dimensions: *Accuracy*, *Sales Diversity*, and *Novelty*. Various indicators are employed to analyze the recommendations pertaining to these characteristics.

The main contributions of the paper can be summarized as follows:

– evaluating the fitness for use of DBpedia and Freebase as input for content-based recommendation tasks in the music domain by means of various quality dimensions and quality indicators;
– providing an evaluation of the performance for four semantic similarity metrics, with regard to recommendation tasks, on the aforementioned encyclopedic datasets.

The remainder of the paper is organized as follows. In Section 2 we summarize the main characteristics of the semantic similarity metrics used in the evaluation while in Section 3 our evaluation methodology is presented. The experimental settings and their outcomes are elaborated in Section 4. Section 5 brings in an overview of related work on recommender systems adopting LD. Finally, Section 6 sketches out future work and concludes the paper.

2 Feature-Based Semantic Similarity Measurement

Information resources in the Web of Data are semantically represented using RDF graphs. To evaluate the similarity between two resources, characteristics like nodes, links, and the mutual relationships are incorporated into calculation. Among others, feature-based semantic similarity metrics quantify similarity between resources in an RDF graph as a measure of commonality and distinction of their hallmarks. The work by Tversky in [1] sheds light on feature-based similarity. It aims at overcoming the major disadvantages of the approaches that compute similarity by measuring distance between points in a space. The work suggests representing objects as a set of common and distinctive features and the similarity of two objects is performed by matching their corresponding collections

[1] http://dbpedia.org
[2] http://www.freebase.com/
[3] http://ir.ii.uam.es/hetrec2011/datasets.html

of features. The features of an object can be represented in one of the following forms: binary values, nominal values, ordinal values, and cardinal values. Measuring similarity using features is based on the premise that the more common features two objects hold, the more similar they are. Bearing on this principle, feature-based semantic similarity metrics first attempt to characterize resources in an `RDF` graph as feature sets and then perform similarity calculation on them. In the following sub-sections we briefly recall the feature-based metrics for computing similarity being exploited in our evaluation. The four metrics have been chosen as representative of the feature-based similarity class since they consider different aspects of the underlying semantic graph for characterizing resources and computing similarity.

GbkSim. The authors in [3] propose a solution to compute similarity by means of a graph-based kernel. By *GbkSim*[4] an abstract walker is sent to explore the `RDF` graph to a specific depth d, en route it collects nodes and edges. The features of a resource α are represented as a vector: $\overrightarrow{a} = (w_{r_1}, w_{r_2}, .., w_{r_n})$. Each element of the vector corresponds to the weight of a resource in the feature set. The weight for resource r_i is calculated as $w_{r_i} = \sum_{m=1}^{d} \gamma_m . c_{\hat{P}^m(\alpha),r_i}$; in which the coefficient γ_m is experimentally selected upon calculation; $c_{\hat{P}^m(\alpha),r_i}$ is the number of edges that connect α to node r_i and it is calculated as: $c_{\hat{P}^m(\alpha),r_i} = |\{(r_i, r_j)|(r_i, r_j) \in \hat{P}^m(\alpha)\}|$; $\hat{P}^m(\alpha)$ is the set of edges collected at depth m. The similarity between two resources α and β is computed as the product of their corresponding feature vectors $\overrightarrow{a} = \{a_i\}_{i=1,..,n}$ and $\overrightarrow{b} = \{b_i\}_{i=1,..,n}$:

$$GbkSim(\alpha, \beta) = \frac{\sum_{i=1}^{n} a_i \times b_i}{\sqrt{\sum_{i=1}^{n}(a_i)^2} \times \sqrt{\sum_{i=1}^{n}(b_i)^2}} \tag{1}$$

VsmSim. In [2] an approach to characterize entities and compute similarity is introduced and evaluated. By *VsmSim*, two entities are supposed to be similar if: (i) There exist direct links between them; (ii) They point to the same object with the same property; (iii) They are pointed by the same subject with the same property. The features of a movie α corresponding to property p are the nodes connected to α through p and represented using the Vector Space Model: $\overrightarrow{a_p} = (w_{r_1,p}, w_{r_2,p}, .., w_{r_n,p})$; in which $w_{r_i,p}$ is the weight of movie r_i wrt. property p, it is computed as the `tf-idf` value of the movie: $w_{r_i,p} = f_{r_i,p} * log(\frac{M}{a_{r_i,p}})$; where $f_{r_i,p}$ is the number of occurrence of movie r_i; M is the number of movies in the collection; $a_{r_i,p}$ is the number of movies pointing to a_{r_i} via p. The similarity related to p is obtained by calculating the cosine similarity of the vectors $\overrightarrow{a_p} = \{a_{i,p}\}_{i=1,..,n}$ and $\overrightarrow{b_p} = \{b_{i,p}\}_{i=1,..,n}$:

$$VsmSim_p(\alpha, \beta) = \frac{\sum_{i=1}^{n} a_{i,p} \times b_{i,p}}{\sqrt{\sum_{i=1}^{n}(a_{i,p})^2} \times \sqrt{\sum_{i=1}^{n}(b_{i,p})^2}}$$

[4] For a clear presentation, in the scope of this paper we assign a name for the metrics that have not been named originally.

Given a set P of properties, the final similarity value can be computed as the (weighted) mean of the values computed for each property p

$$VsmSim(\alpha, \beta) = \frac{\sum_{p \in P} \omega_p VsmSim_p(\alpha, \beta)}{|P|} \tag{2}$$

with ω_p being weights computed via a genetic algorithm.

FuzzySim. In an attempt to incorporate the human judgment of similarity, a similarity metric, *FuzzySim* is presented in [4]. Properties are considered as features and intuitively classified into groups in descending order according to their level of importance $(g_1, g_2, .., g_n)$. The similarity value between two resources α and β on group g_i is defined as: $S_i(\alpha, \beta) = \frac{f_i(\alpha,\beta)}{f_i(\alpha)}$; where $f_i(\alpha, \beta)$ is the set of features pertaining to property group g_i that α and β have in common; $f_i(\alpha)$ is the set of features of α wrt. g_i. The membership degree of the similarity value corresponding to g_i is: $\mu(S_i) = (S_i)^{i-r(g_i,c)}$; where $r(g_i, c)$ is the ratio of the number of properties for set g_i wrt. the total number of properties. The weight $\varphi_j(m)$ for the j^{th} element of the property set is given by: $\varphi_j(m) = \sqrt{\frac{\sum_{k=1}^{j} m_k}{\sum_{k=1}^{n} m_k}} - \sqrt{\frac{\sum_{k=1}^{j-1} m_k}{\sum_{k=1}^{n} m_k}}$ in which $m = (\mu(b_1), \mu(b_1), .., \mu(b_n))$ is the ascending sorted membership vector of $(S_1, S_2, .., S_n)$. The similarity between α and β is computed by means of a fuzzy function:

$$FuzzySim(\alpha, \beta) = aggr(S_1, S_2, ..., S_n) = \sum_{j=1}^{n} b_j.\varphi_j(m) \tag{3}$$

Jaccard. For comparison, we use the Jaccard's index to compute similarity between feature sets. The features of a resource are modeled as a set of nodes in its surroundings. For two resources α and β, two abstract walkers are deployed to traverse the graph at a specific depth to acquire features. At each depth, a walker collects nodes, after visiting depth d, the walkers return the set of nodes $N_d(\alpha)$ and $N_d(\beta)$. The metric calculates the similarity between two resources using the Jaccard's index:

$$Jaccard(\alpha, \beta) = \frac{|N_d(\alpha) \bigcap N_d(\beta)|}{|N_d(\alpha) \bigcup N_d(\beta)|} \tag{4}$$

3 Assessment Methodology

Data extracted from LD might be suitable for certain purposes but not for every purpose [22]. The quality of a piece of data is heavily dependent on the usage as well as the tasks performed on it [23]. For measuring the *fitness for use* of a dataset, a set of *quality dimensions* needs to be identified [23]. Scoring functions can be used to calculate an assessment score from the related *quality indicators* as a gauge of how well suitable the data for a particular purpose is. In the scope

Table 1. Formulas used to evaluate the quality of recommendations. rel_k is 1 if the k-th item in the list is relevant for the user u, otherwise it is 0. *test(u)* represents the set of relevant items in the test set for the user u. Since the rating scale in the Last.fm dataset is from 1 to 5, we consider the ratings 4 and 5 as relevant. I is the whole item set; *TopN(u)* is the set of the N items recommended to u; $rec(i)$ represents the number of users who received the recommendation of the item i; *total* is the overall number of recommendations across all users. To compute the Gini coefficient, set I must be indexed in ascending order wrt. the number of recommendations ($rec(i)$).

Accuracy	Sales Diversity	Novelty
$P@N(u) = \frac{\sum_{k=1}^{N} rel_k}{N}$ $R@N(u) = \frac{\sum_{k=1}^{N} rel_k}{\lvert test(u)\rvert}$	$coverage = \frac{\lvert\bigcup_{u\in U} TopN(u)\rvert}{\lvert I\rvert}$ $entropy = -\sum_{i\in I}\left(\frac{rec(i)}{total}\right) ln \left(\frac{rec(i)}{total}\right)$ $gini = 2\sum_{i\in I}\left[\left(\frac{\lvert I\rvert+1-i}{\lvert I\rvert+1}\right)\left(\frac{rec(i)}{total}\right)\right]$	$\%Long\text{-}tail = \frac{\sum_{i\in Long\text{-}tail} rec(i)}{total}$

of this paper, we work with a specific use case, LD for the music domain used as input for recommendation tasks. Recommender systems are built to suggest things that are of interest to a user, e.g. books, movies, songs [2]. To be able to provide users with meaningful recommendations, recommender systems may enrich their background data by exploiting external sources. In this sense, the quality of the input data plays a key role in producing adequate recommendations. As seen in Section 5, most of the approaches to recommendation built on top of LD datasets exploits DBpedia. To our knowledge, an analysis on the influence of the underlying dataset for the quality of recommendation results has not been performed yet. Having this observation in mind, we compared recommendation results by using two of the richest encyclopedic LD sources. Data retrieved from both DBpedia and Freebase[5] is then used for computing similarity between resources employing the aforementioned similarity metrics. Afterwards, the computed similarity values are fed into a content-based recommender system to produce the final recommendations. For judging data quality, we take into account the quality dimensions of *Accuracy*, *Sales Diversity*, and *Novelty* in a top-N recommendation task. Recently, accuracy has been recognized to be not sufficient to evaluate a recommender system. *Sales Diversity* represents an important quality dimension for both business and user perspective, since improving the coverage of the items catalog and of the distributions of the items across the users may increase the sales and the user satisfaction [21]. *Novelty* measures the ability of the system to foster discovery in the recommendation workflow [25]. The formulas used to evaluate the quality dimensions are formally described in Table 1 and more discursively below.

(i) Considering only the top N results, for measuring *Accuracy* we use precision P@N (the fraction of the top-N recommended items being relevant to the user u) and recall R@N (the fraction of relevant items from the test set appearing in the N predicted items).

[5] We used the RDF version Freebase released as baseKB available at http://basekb.com/.

(ii) To measure *Sales Diversity*, we consider catalog coverage [19] (the percentage of items in the catalog that have ever been recommended to users), and Entropy and Gini coefficient [20,21] (for the distribution of recommended items). The latter are useful to analyze the concentration degree of items across the recommendations. The scale for Gini coefficient is reversed, thereby forcing small values to represent low distributional equity and large values to represent higher equity.

(iii) One metric is chosen to measure the *Novelty* of the recommendations: the percentage of long-tail items among the recommendations across all users [20], considering the 80 percent of less rated items in the training set as *Long-tail* items.

For our experiments, we re-used the setup adopted in [6]. Specifically, we have implemented a content-based recommender system using a k-nearest neighbors algorithm. It selects the k most similar entities β, called neighbors, to a given item α using a similarity function $sim(\alpha, \beta)$. The score P for a given user-item pair (u, α) is computed using a weighted sum, where the weights are the similarities between the items. The formula takes into account the neighbors of α belonging to the user profile $profile(u)$ and the relative scores $r(u, \beta)$ assigned by the user u.

$$P(u, \alpha) = \frac{\sum_{\beta \in neighbors(\alpha) \cap profile(u)} sim(\alpha, \beta) \cdot r(u, \beta)}{\sum_{\beta \in neighbors(\alpha) \cap profile(u)} sim(\alpha, \beta)}$$

The function $sim(\alpha, \beta)$ was computed using the similarity metrics shown in the previous section and k was fixed at 20. We selected the well-known dataset `Last.fm hetrec-2011`. In order to compare the two `LD` datasets in an ordinary situation, we downsized the number of artists and bands to the 1000 most popular ones and, after that reduction, we removed the cold users, i.e. those having the number of ratings below the average of all users. The reason behind this choice was to reduce as much as possible the well known negative effect on the computation of the recommendation list due to users with a low number of ratings. After that, we used the holdout method to split the dataset into training set and test set. We built the training set by using, for each user, the first 80% of the her ratings and the remaining 20% to build the test set. Therefore, the first 80% of the ratings of each user represents her profile. One of our mapping datasets[6] was utilized to associate each item with its counterpart in `DBpedia` [24]. By using `owl:sameAs` links we were then able to retrieve `Freebase` mappings from the `DBpedia` ones.

4 Experimental Results

Feature sets are a prerequisite in similarity calculation for feature-based similarity metrics. It is, therefore, necessary to build a set of features for each resource.

[6] http://sisinflab.poliba.it/semanticweb/lod/recsys/datasets/

Table 2. The set of properties used for collecting feature sets from `DBpedia`.

Outbound	
rdf:type	dbo:associatedAct
owl:sameAs	dbo:influenced
dbo:instrument	dbo:influencedBy
dbo:writer	dbo:bandMember
dcterms:subject	dbo:formerBandMember
dbo:associatedBand	dbo:currentMember
dbo:associatedMusicalArtist	dbo:pastMember
dbo:background	dbo:occupation
dbo:genre	dbo:birthPlace

Inbound	
dbo:previousWork	dbo:producer
dbo:subsequentWork	dbo:artist
dbo:knownFor	dbo:writer
dbo:award	dbo:associatedBand
dbo:album	dbo:associatedMusicalArtist
dbo:notableWork	dbo:musicalArtist
dbo:lastAppearance	dbo:musicalBand
dbo:basedOn	dbo:musicComposer
dbo:starring	dbo:bandMember
dbo:series	dbo:formerBandMember
dbo:openingFilm	dbo:starring
dbo:related	dbo:composer

In an `LD` setting, building the the set of features goes through the selection of a set of `RDF` properties considered as relevant for the domain. For `DBpedia`, the top 20% most popular properties of the `DBpedia` ontology used in the musical domain apart from `dbo:wikiPageWikiLink` have been chosen, plus `owl:sameAs`, `rdf:type` and the `dcterms:subject` property that connects resources to categories. Table 2 shows the selected list of properties. Similarly, for `Freebase` we selected the set of 20% most popular properties connecting to resources whose type is either `basekb:music.musical_group`[7] or `basekb:music.artist`[8]. This results in 288 outgoing and 220 incoming properties. The set of properties is not listed here due to space limitations. An `RDF` graph consists of a huge number of edges and nodes, spreading out on numerous layers of predicates. It is certainly impractical to address all nodes and edges in it. Therefore, we collected a set of features by expanding the graph using the selected set of properties up to a limited depth. Considering a pair of resources that are involved in the similarity calculation, a neighborhood graph was built by expanding from each resource using the selected set of properties. For each resource, depending on the type of experiments, features can be collected in one or two levels of edges. Furthermore, also depending on the purpose of measurement, an extension can either be done using only outbound edges or using both inbound and outbound edges.

In order to investigate the effect of the selection of feature sets on the outcome, we carried out experiments using independent settings. First, we considered different levels of depth and then in each setting, the selection of properties for collecting a set of features. Two independent similarity calculations have been performed: similarity computed with one-hop features and similarity computed with two-hop features. The experimental results are clarified in the following sub-sections.

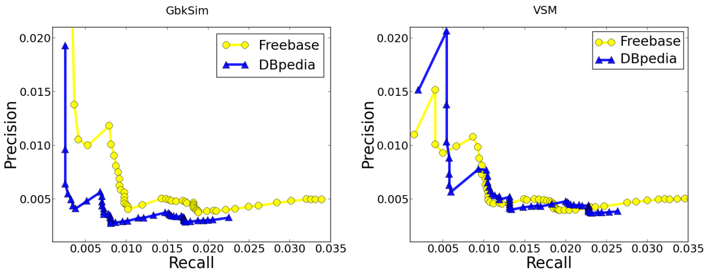
One-hop Features. Experiments were conducted in accordance with two separate configurations:

Configuration 1. *Both inbound and outbound properties are used to build the set of features of a resource.*

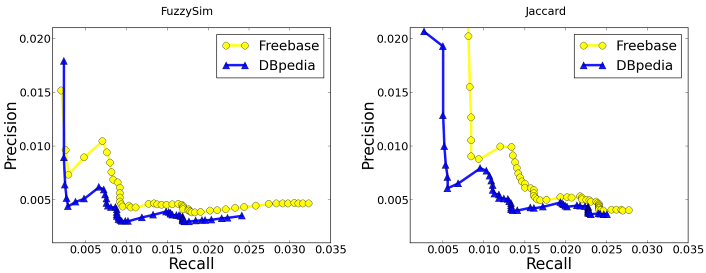
[7] http://rdf.basekb.com/ns/music.musical_group
[8] http://rdf.basekb.com/ns/music.artist

Table 3. Comparison of results for the four algorithms with Top-10, Top-20, Top-30 between `DBpedia` and `Freebase` using both inbound and outbound properties. The name in a cell indicates the dataset that obtains the best result. With largest Top-N the differences between `DBpedia` and `Freebase` are similar to the Top-30 results, therefore they are omitted due to space limitations.

		Precision	Recall	Coverage	Entropy	Gini	%Long-tail
GbkSim	Top-10	Freebase	Freebase	Freebase	DBpedia	DBpedia	DBpedia
	Top-20	Freebase	Freebase	Freebase	DBpedia	DBpedia	DBpedia
	Top-30	Freebase	Freebase	Freebase	DBpedia	DBpedia	DBpedia
VsmSim	Top-10	Freebase	Freebase	Freebase	DBpedia	DBpedia	DBpedia
	Top-20	Freebase	DBpedia	DBpedia	DBpedia	DBpedia	DBpedia
	Top-30	Freebase	DBpedia	DBpedia	DBpedia	DBpedia	DBpedia
FuzzySim	Top-10	Freebase	Freebase	Freebase	DBpedia	DBpedia	DBpedia
	Top-20	Freebase	Freebase	Freebase	DBpedia	DBpedia	DBpedia
	Top-30	Freebase	Freebase	Freebase	DBpedia	DBpedia	DBpedia
Jaccard	Top-10	Freebase	Freebase	Freebase	Freebase	Freebase	DBpedia
	Top-20	Freebase	Freebase	Freebase	Freebase	DBpedia	DBpedia
	Top-30	Freebase	Freebase	Freebase	Freebase	Freebase	DBpedia



(a)



(b)

Fig. 1. Recommedation using similarity values computed on one-hop features: Precision - Recall curves obtained by varying the length of the recommendations list from 1 to 50, with 20 neighbors. Inbound and outbound links are used in combination.

Accuracy. Figure 1 shows the precision and recall values for all metrics. Generally, recommendations computed using data extracted from `Freebase` have a better precision-recall balance and higher recall values. This holds for all similarity metrics except for *VsmSim*. Using the latter, generally there is an overlap among the values, but still `Freebase` helps achieve the highest recall values. Table 3 displays the quality indicators for all the metrics on both datasets

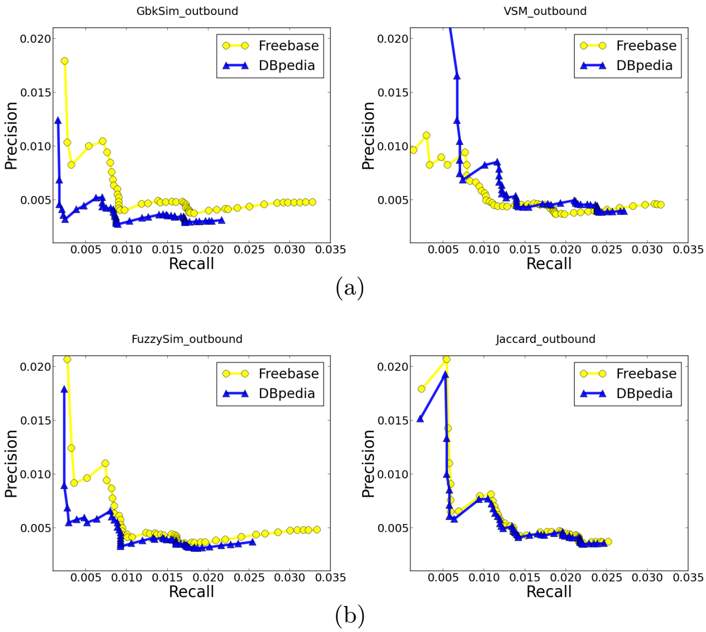


Fig. 2. Recommendation using similarity values computed on one-hop features: Precision - Recall curves obtained by varying the length of the recommendations list from 1 to 50, with 20 neighbors. Only outbound links are used.

considering `Top-10`, `Top-20` and `Top-30`. Those results demonstrate that `Freebase` dataset brings the highest accuracy for all the similarity metrics, except for *VsmSim* as mentioned before. However, the differences between the two datasets often have a marginal significance, whereas the charts in Figure 1 show a more complete and general view in term of accuracy.

Sales Diversity. As shown in Table 3, using `Freebase` data always produces better coverage. In terms of distribution (Entropy and Gini), generally using data from `DBpedia` obtains better values compared to `Freebase`. However, those results are not easily comparable because the `DBpedia` coverage values are too low. By recommending very few items, it is much more likely to obtain a good distribution; whereas, by recommending more items, many of these may be suggested few times (even just once). This is confirmed by the fact that the entropy values are closer than the Gini values between `DBpedia` and `Freebase`, considering that Gini index is more sensible to the inequality and Entropy to the distribution among the recommendations.

Novelty. In terms of percentage of long-tail items, `DBpedia` contributes to a better novelty compared to `Freebase` in almost every configuration. This means that using `DBpedia` tends to suggest a smaller subset of items, but these do not necessarily belong to the most popular ones. In contrast, `Freebase` can help cover more items but generally with a slightly larger popularity bias.

Configuration 2. *Only outbound properties are used to build the set of features of a resource.*
Figure 2 shows the accuracy obtained by the recommendations computed using similarity results in this setting. A noteworthy observation is that, for all similarity metrics, the accuracy of the recommendations calculated by using data from `DBpedia` is analogous to the accuracy obtained by using data from `Freebase`. We also observed the same trend for all metrics by other quality dimensions (Sales Diversity and Novelty). Thus, the corresponding quality indicators are not depicted due to space limitations. Compared with Configuration 1, we come to the conclusion that the utilization of both inbound and outbound properties for computing semantic similarity contributes towards an improvement in the recommendation results.

Table 4. Comparison of results for the four algorithms with Top-10, Top-20, Top-30 between `DBpedia` and `Freebase` with exploration up to two hops using both inbound and outbound properties. The name in a cell indicates the dataset that obtains the best result.

		Precision	Recall	Coverage	Entropy	Gini	%Long-tail
GbkSim	Top-10	Freebase	Freebase	Freebase	Freebase	DBpedia	DBpedia
	Top-20	Freebase	Freebase	Freebase	DBpedia	DBpedia	DBpedia
	Top-30	Freebase	Freebase	Freebase	DBpedia	DBpedia	DBpedia
VsmSim	Top-10	DBpedia	DBpedia	Freebase	Freebase	Freebase	DBpedia
	Top-20	DBpedia	DBpedia	Freebase	DBpedia	DBpedia	DBpedia
	Top-30	DBpedia	DBpedia	Freebase	Freebase	DBpedia	DBpedia
FuzzySim	Top-10	Freebase	Freebase	Freebase	Freebase	DBpedia	DBpedia
	Top-20	Freebase	Freebase	Freebase	DBpedia	DBpedia	DBpedia
	Top-30	Freebase	Freebase	Freebase	Freebase	DBpedia	DBpedia
Jaccard	Top-10	Freebase	Freebase	Freebase	DBpedia	DBpedia	Freebase
	Top-20	Freebase	Freebase	Freebase	DBpedia	DBpedia	DBpedia
	Top-30	Freebase	Freebase	Freebase	DBpedia	DBpedia	DBpedia

Two-hop Features. We studied the influence of exploration depth for collecting features over the recommendation outcomes. Hence, the same experimental procedures were replicated with depth $d = 2$ and the results obtained are as follows:

Configuration 1. *Both inbound and outbound properties are used*

The accuracy values for all metrics using 2 hops are depicted in Figure 3. Similar to the experiments performed using one-hop features, we witnessed the same pattern of the quality indicators for this experimental setting. Using the `Freebase` dataset to produce recommendations yields a better precision-recall balance as well as higher recall values. For both *VsmSim* and *Jaccard*, similarity values on the `DBpedia` dataset help produce the best recommendations in terms of accuracy; meanwhile similarity values computed by *Jaccard* on the `Freebase` dataset contribute to a better precision-recall balance. Considering Top-10, Top-20 and Top-30, the corresponding quality indicators for all the metrics are shown in Table 4. Once again, apart from *VsmSim*, recommendation with the `Freebase` dataset using other similarity metrics still brings the highest accuracy.

Configuration 2. *Only outbound properties are used*

For this experimental setting, by all metrics we also obtained comparable results using similarity values calculated from Configuration 2 for one-hop features. Figure 4 depicts the precision-recall balance for all similarity metrics. The

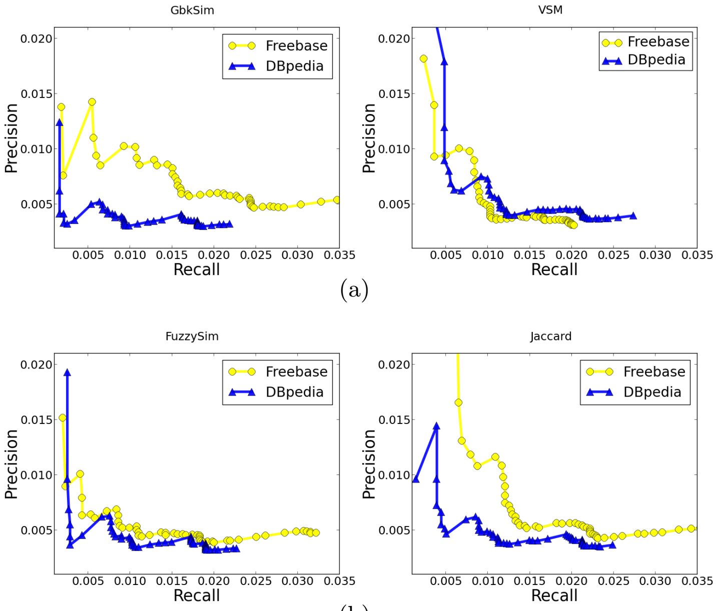


Fig. 3. Recommedation using similarity values computed on two-hop features: Precision - Recall curves obtained by varying the length of the recommendations list from 1 to 50, with 20 neighbors. Inbound and outbound links are used in combination.

results obtained using `DBpedia` show no substantial difference compared to the results with considering also inbound properties. While the results for `Freebase` show an overall strong decrease both in terms of precision-recall balance and recall values, demonstrating that the inbound properties in `Freebase` dataset play an important role, as already seen for one-hop configuration. This decrease is particularly evident using *GbkSim* and *Jaccard*.

It can be seen that, the outcomes of the recommendations on two-hop features confirm the experimental results for recommendation using one-hop features.

Comparison between using One-hop and Two-hop Features. We carried out a comparative analysis between using one-hop and two-hop features. As a matter of fact, the exploration of the graph comes at a price and sometime it might not be necessary. Using `DBpedia` with inbound and outbound properties, there are no relevant differences expanding the features up to two hops. Considering Figures 1 and 3, with respect to `Freebase` with inbound and outbound properties, *GbkSim* metric with two-hop features obtains better results in terms of precision with respect to one-hop configuration. In terms of recall, using the *Jaccard* metric with two-hop features obtains better results with respect to one-hop configuration. Conversely, the recall values using *VsmSim* decrease with two-hop instead one-hop features. There are no substantial differences in

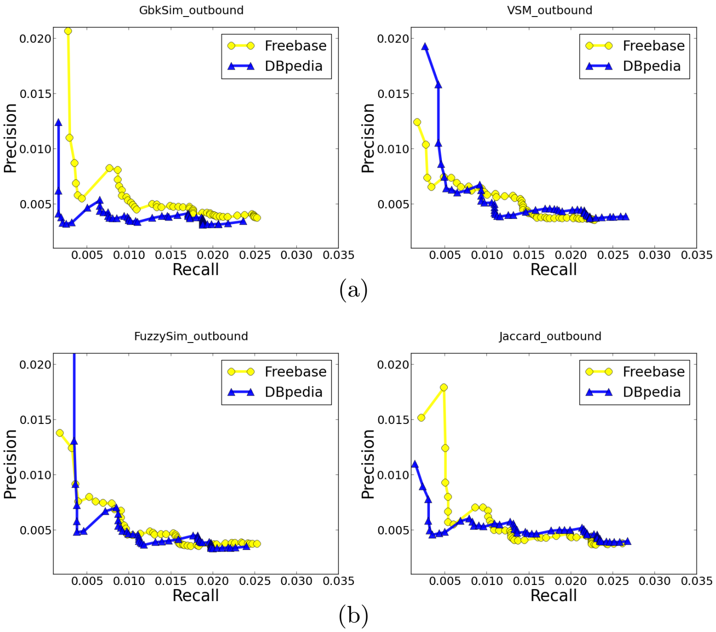


Fig. 4. Recommendation using similarity values computed on two-hop features: Precision - Recall curves obtained by varying the length of the recommendations list from 1 to 50, with 20 neighbors. Only outbound links are used.

the case of *FuzzySim*. Table 5 shows the gains and losses obtained expanding the features up to two hops with Top-10, Top-20 and Top-30, confirming what has been said so far. Considering the Sales Diversity measure, using `DBpedia` we obtain better results with two-hop features using all the similarity metrics. Using `Freebase` gains better results with two-hop features using *Jaccard* and *VsmSim*. However, `Freebase` always overcomes `DBpedia`. It is worth noticing that the recommendation distribution (Entropy and Gini measures) achieves substantial improvements with two-hop features for each configuration. Instead, when only outbound properties are used, the performances by utilizing `DBpedia` are slightly lower expanding the features up to two hops, especially in terms of precision with *VsmSim* and *FuzzySim*. With respect to `Freebase`, the recall decreases especially with *GbkSim* and *FuzzySim*. The adoption of `Freebase` instead of `DBpedia` shows its benefits when used in conjunction with *GbkSim*, when two-hop features are considered. The other similarity metrics – even though they are relatively simple – do not exhibit that considerable improvements to justify the increased computational effort needed to further explore the semantic graph of one more hop.

Table 5. Gains and losses obtained using two-hop features respect to one-hop ones using both inbound and outbound properties. The symbol + indicates a gain, − a loss while ∼ a negligible variation.

			Precision	**Recall**	**Coverage**	**Entropy**	**Gini**	**%Long-tail**
GbkSim	Top-10	Freebase	+	+	−	+	+	−
		DBpedia	−	−	+	−	−	−
	Top-20	Freebase	+	+	−	+	+	+
		DBpedia	+	+	+	+	+	∼
	Top-30	Freebase	+	+	−	+	+	∼
		DBpedia	+	+	+	∼	+	−
VsmSim	Top-10	Freebase	−	−	+	+	+	−
		DBpedia	−	−	+	+	+	−
	Top-20	Freebase	−	−	+	+	+	−
		DBpedia	−	−	+	+	+	−
	Top-30	Freebase	−	−	+	+	+	−
		DBpedia	−	−	+	+	−	−
FuzzySim	Top-10	Freebase	−	−	−	+	+	−
		DBpedia	+	+	+	−	∼	∼
	Top-20	Freebase	+	+	∼	+	+	−
		DBpedia	+	+	+	∼	+	+
	Top-30	Freebase	+	+	−	+	+	−
		DBpedia	+	+	+	+	+	∼
Jaccard	Top-10	Freebase	−	−	+	+	∼	+
		DBpedia	−	−	+	+	+	−
	Top-20	Freebase	−	−	+	−	−	−
		DBpedia	−	−	+	+	+	−
	Top-30	Freebase	∼	∼	+	−	−	−
		DBpedia	−	−	+	+	+	∼

4.1 Discussion

In this section we discuss the general trends emerging from Table 3, 4 and 5.

By looking at Table 3 and Table 4, an interesting question arises: *why `Freebase` seems to facilitate better accuracy and catalog coverage while `DBpedia` helps obtain superior novelty and aggregate diversity*[9]?

As for accuracy, we assume that in `Freebase`, at least for our target domain, items considered as similar by users are actually connected by relevant properties with each other. This reflects the strong crowd-sourced nature of `Freebase` and also means that, in this case, `Freebase` is richer than `DBpedia` in terms of encoded knowledge. Both data sources are derived from `Wikipedia`, however `Freebase` can be flexibly edited by user communities who utilize numerous sources for encoding metadata. Thus, each `Freebase` topic consists of an expansion of the original `Wikipedia` topic, which is not the case in `DBpedia`. Especially for domains being managed by Google, `Freebase` has a higher topic coverage than `DBpedia` [26]. Moreover, the social nature of `Freebase` also implies that items resulting popular among the users are also "popular" in the underlying graph. This means that they are richer in terms of related data and are more connected to other entities. This also explains both the higher value of precision and recall and the lower values of novelty when using `Freebase`. Indeed, on the one side we know that computing recommendations based on items popularity results in good predictions for the end users [5]; on the other side, as with `Freebase` we concentrate more on popular items we have lower results when evaluating novelty (long-tail) compared to `DBpedia`. Regarding the differences between Coverage and aggregate diversity (*Entropy* and *Gini* index) a possible explanation is due to the very low values of catalog coverage when using

---

[9] A further and more detailed investigation is needed for *VsmSim*.

DBpedia. Since there are less recommended items from the catalog, they have a higher probability to be better distributed across the users.

The results summarized in Table 5 show other interesting trends when exploring the underlying graph to compute recommendation. We see that values for novelty tend to decrease when we move from a one-hop to a two-hop exploration while this is not the case for catalog coverage and aggregate diversity. Possible explanations for these behaviors are: (i) popular items get more connected when exploring the graph thus obtaining better similarity results. This justifies the novelty decrease; (ii) the increasing in the number of connections also reflects in the selection of more items (better coverage) even if the new items are selected mostly among the popular ones; (iii) finally, as we have better similarity values due to better overlaps among items descriptions, we gain in aggregate diversity as a better similarity values means a better chance to be recommended.

5 Related Work

To the best of our knowledge, none of the existing work has conducted a comprehensive evaluation on the fitness for use of datasets in combination with different recommendation strategies. Some studies partly address the issue in different settings. In this section we review the most notable work on this topic.

Leveraging LD sources like DBpedia for recommendation tasks appears to be highly beneficial as demonstrated by numerous applications. One of the first approaches that exploits Linked Data for building recommender systems is [9]. The authors of [8] present a knowledge-based framework leveraging DBpedia for computing cross-domain recommendations. A graph-based recommendation approach utilizing model- and memory-based link prediction methods is presented in [10]. LD datasets are exploited in [11] for personalized exploratory search using a spreading activation method for finding semantic relatedness between items belonging to different domains. For recommending movies, a content-based system exploiting data extracted from DBpedia has been proposed in [2] based on the adaptation of Vector Space Model to semantic networks. In [24] a hybrid algorithm - named *Sprank* - is proposed to compute *top-N* item recommendations from implicit feedback. Path-based features are extracted from DBpedia to detect subtle relationships among items in semantic graphs. Afterwards, recommendations are produced by incorporating ontological knowledge with collaborative user preferences. The proposed algorithm gains good accuracy, especially in conditions of higher data sparseness. A work that can be considered as a base for our paper is [6]. Two semantic similarity metrics, *SimRank* and *Personalized PageRank* are used to compute similarity between resources in RDF graphs. There, exploiting semantic similarity in producing input for a content-based recommender system has proven to bring benefits. A full SPARQL-based recommendation engine named RecSPARQL is presented in [12]. The proposed tool extends the syntax and semantics of SPARQL to enable a generic and flexible way for collaborative filtering and content-based recommendations over arbitrary RDF graphs. The authors of [13] propose an approach for topic suggestions based on some proximity measures defined on the top of the DBpedia graph.

In [14] the authors present an event recommendation system based on LD and user diversity. A semantic-aware extension of the SVD++ model, named SemanticSVD++, is presented in [15]. It incorporates semantic categories of items into the model. The model is able also to consider the evolution over time of user's preferences. In [16] the authors improve their previous work for dealing with cold-start items by introducing a vertex kernel for getting knowledge about the unrated semantic categories starting from those categories which are known. Another interesting direction about the usage of LD for content-based RSs is explored in [17] where the authors present Contextual eVSM, a content-based context-aware recommendation framework that adopts a semantic representation based on distributional models and entity linking techniques. In particular entity linking is used to detect entities in free text and map them to LD.

Finally, in [18] the authors propose the usage of recommendation techniques for providing personalized access to LD. The proposed method is a user-user collaborative filtering recommender wherein the similarity between the users takes into account the commonalities and informativeness of the resources instead of treating resources as plain identifiers.

6 Conclusion

In this paper we analyze the fitness for use of two LD encyclopedic datasets, namely DBpedia and Freebase, to cope with recommendation tasks in the music domain. Similarity values computed on data retrieved from DBpedia and Freebase were used to feed a content-based recommender system to produce recommendation lists. To further study the influence of the selection of features on the recommendations, we performed experiments using (i) four different feature-based similarity values, (ii) two levels of depth in the graph exploration and (iii) different property sets for gathering features from RDF graphs. We executed a series of experiments on the Last.fm dataset thus comparing the recommendation results measuring their performances in terms of accuracy, catalog coverage, distribution and novelty. For most of the experimental settings, we saw that exploiting Freebase obtains better accuracy and catalog coverage. Whereas, the dataset from DBpedia generally fosters the novelty of recommendations. Regarding the distribution, at first glance using the DBpedia dataset appears to perform better, but a careful analysis shows that the results are somehow comparable. For all settings, the selection of both inbound and outbound links for computing similarity makes a difference to the overall performance. Indeed, it is worth noticing that considering links as undirected has a positive impact in the performance of the recommendation engine. We also saw that Freebase obtains improvements using *GbkSim* expanding the features up to two hops. Although Freebase will be retired at the end of June 2015 as a standalone project, all its data will flow into the Wikidata project thus becoming its stable nucleus. Hence, we are confident that the results presented in this paper will be useful also in the light of a comparison with the upcoming edition of Wikidata. In conclusion, we confirm that encyclopedic LD datasets are an interesting source of data to build content-based recommender systems, but the choice of the right dataset might affect the performance of the system with regards to some evaluation dimensions such as accuracy, novelty and diversity of results.

Acknowledgments. The authors acknowledge partial support of PON01_03113 ERMES, PON02_00563_3470993 Vincente and PON02_00563_3446857 KHIRA.

References

1. Tversky, A.: Features of similarity. Psychological Review **84**(4), 327–352 (1977)
2. Di Noia, T., Mirizzi, R., Ostuni, V.C., Romito, D., Zanker, M.: Linked open data to support content-based recommender systems. In: Proc. of I-SEMANTICS 2012, pp. 1–8 (2012)
3. Ostuni, V.C., Di Noia, T., Mirizzi, R., Di Sciascio, E.: A linked data recommender system using a neighborhood-based graph Kernel. In: Hepp, M., Hoffner, Y. (eds.) EC-Web 2014. LNBIP, vol. 188, pp. 89–100. Springer, Heidelberg (2014)
4. Zadeh, P.D.H., Reformat, M.Z.: Fuzzy semantic similarity in linked data using the OWA operator. In: Annual Meeting of the North American Fuzzy Inf. Proc. Society (2012)
5. Cremonesi, P., Koren, Y., Turrin, R.: Performance of recommender algorithms on top-n recommendation tasks. In: Proc. of RecSys10. pp. 39–46 (2010)
6. Nguyen, P.T., Tomeo, P., Di Noia, T., Di Sciascio, E.: An evaluation of SimRank and personalized PageRank to build a recommender system for the web of data. In: 7th International Workshop on Web Intelligence & Communities. ACM (2015)
7. Lops, P., Des Gemmi, M., Semeraro, G.: Recommender Systems Handbook, pp. 73–105. Springer, Heidelberg (2011)
8. Fernández-Tobías, I., Cantador, I., Kaminskas, M., Ricci., F.: A generic semantic-based framework for cross-domain recommendation. In: Proc. of HetRec 2011, pp. 25–32 (2011)
9. Heitmann, B., Hayes., C.: Using linked data to build open, collaborative recommender systems. In: AAAI Spring Symposium: Linked Data Meets Artificial Intelligence (2010)
10. Lommatzsch, A., Plumbaum, T., Albayrak., S.: A linked dataverse knows better: boosting recommendation quality using semantic knowledge. In: Proc. of SEMAPRO 2011, pp. 97–103 (2011)
11. Marie, N., Corby, O., Gandon, F., Ribière., M.: Composite interests' exploration thanks to on-the-fly linked data spreading activation. In: Proc. of HT 2013 (2013)
12. Ayala, V.A.A., Przyjaciel-Zablocki, M., Hornung, T., Schätzle, A., Lausen., G.: Extending sparql for recommendations. In: Proc. of SWIM 2014 (2014)
13. Stankovic, M., Breitfuss, W., Laublet., P.: Linked-data based suggestion of relevant topics. In: Proc. of I-Semantics 2011, pp. 49–55 (2011)
14. Khrouf, H., and Troncy., R.: Hybrid event recommendation using linked data and user diversity. In: Proc. of RecSys 2013, pp. 185–192 (2013)
15. Rowe, M.: SemanticSVD++: incorporating semantic taste evolution for predicting ratings. In: Proc. of 2014 IEEE/WIC/ACM WI 2014 (2014)
16. Rowe, M.: Transferring semantic categories with vertex Kernels: recommendations with SemanticSVD++. In: Mika, P., Tudorache, T., Bernstein, A., Welty, C., Knoblock, C., Vrandečić, D., Groth, P., Noy, N., Janowicz, K., Goble, C. (eds.) ISWC 2014, Part I. LNCS, vol. 8796, pp. 341–356. Springer, Heidelberg (2014)

17. Musto, C., Semeraro, G., Lops, P., de Gemmis, M.: Combining distributional semantics and entity linking for context-aware content-based recommendation. In: Dimitrova, V., Kuflik, T., Chin, D., Ricci, F., Dolog, P., Houben, G.-J. (eds.) UMAP 2014. LNCS, vol. 8538, pp. 381–392. Springer, Heidelberg (2014)
18. Dojchinovski, M., Vitvar, T.: Personalised Access to Linked Data. In: Janowicz, K., Schlobach, S., Lambrix, P., Hyvönen, E. (eds.) EKAW 2014. LNCS, vol. 8876, pp. 121–136. Springer, Heidelberg (2014)
19. Ge, M., Delgado-Battenfeld, C., Jannach, D.: Beyond accuracy: evaluating recommender systems by coverage and serendipity. In: Proc. of RecSys 2010, pp. 257–260, ACM (2010)
20. Adomavicius, G., Kwon, Y.: Improving Aggregate Recommendation Diversity Using Ranking-Based Techniques. IEEE TKDE **24**(5), 896–911 (2012)
21. Vargas, S., Castells, P.: Improving sales diversity by recommending users to items. In: Proc. of RecSys 2014, pp. 145–152 (2014)
22. Knuth, M., Kontokostas, D., Sack, H.: Linked data quality: identifying and tackling the key challenges. In: Proc. of LDQ@SEMANTiCS (2014)
23. Mendes, P.N., Mühleisen, H., Bizer, C.: Sieve: linked data quality assessment and fusion. In: Proc. of EDBT-ICDT 2012, pp. 116–123. ACM (2012)
24. Ostuni, V.C., Di Noia, T., Di Sciascio, E., Mirizzi, R.: Top-N recommendations from implicit feedback leveraging linked open data. In: Proc. of RecSys 2013 (2013)
25. Celma, O., Herrera, P.: A new approach to evaluating novel recommendations. In: Proc. RecSys 2008. ACM (2008)
26. Lehmann, J., et al.: DBpedia - A large-scale, multilingual knowledge base extracted from Wikipedia. Journal of Semantic Web **6**, 167–195 (2015)