



# Empirical Analysis of Ranking Models for an Adaptable Dataset Search

Angelo B. Neves<sup>1</sup>(✉) , Rodrigo G. G. de Oliveira<sup>1</sup> ,  
Luiz André P. Paes Leme<sup>1</sup> , Giseli Rabello Lopes<sup>2</sup> ,  
Bernardo P. Nunes<sup>3,4</sup>, and Marco A. Casanova<sup>3</sup>

<sup>1</sup> Fluminense Federal University, Niterói, RJ, Brazil  
`{nevesangelo,rodrigoguerra,lleme}@id.uff.br`

<sup>2</sup> Federal University of Rio de Janeiro, Rio de Janeiro, RJ, Brazil  
`giseli@dcc.ufrj.br`

<sup>3</sup> PUC-Rio, Rio de Janeiro, RJ, Brazil  
`{bnunes,casanova}@inf.puc-rio.br`

<sup>4</sup> Federal University of the State of Rio de Janeiro, Rio de Janeiro, RJ, Brazil  
`bernardo.nunes@uniriotec.br`

**Abstract.** Currently available datasets still have a large unexplored potential for interlinking. Ranking techniques contribute to this task by scoring datasets according to the likelihood of finding entities related to those of a target dataset. Ranked datasets can be either manually selected for standalone linking discovery tasks or automatically inspected by programs that would go through the ranking looking for entity links. This work presents empirical comparisons between different ranking models and argues that different algorithms could be used depending on whether the ranking is manually or automatically handled and, also, depending on the available metadata of the datasets. Experiments indicate that ranking algorithms that performed best with nDCG do not always have the best Recall at Position k, for high recall levels. The best ranking model for the manual use case (with respect to nDCG) may need 13% more datasets for 90% of recall, i.e., instead of just a slice of 34% of the datasets at the top of the ranking, reached by the best model for the automatic use case (with respect to recall@k), it would need almost 47% of the ranking.

**Keywords:** Linked Data · Entity linking · Recommendation  
Dataset · Ranking · Empirical evaluation

## 1 Introduction

The Web of Data (WoD) has been growing fast and is facing the challenge of increasing the links between entities from distinct datasets. The more interlinked they are, the greater intrinsic value of their underlying knowledge base will be, which allows the development of more innovative applications.

The *entity linking* task with respect to the entities of a target dataset consists of: (1) selecting other so-called *relevant datasets* that would contain related entities; (2) inspecting their content to infer entity relationships, i.e., infer links; and (3) making the relationships explicit by adding new RDF statements to the target dataset. One of the most popular relationships is the equivalence relation (*owl:sameAs*) addressed in [13, 15, 16, 18].

Statistics about the WoD [1] show that more than 70% of the datasets are linked with entities of at most two other datasets, and that the vast majority of them are linked only with popular ones, such as DBpedia, Geonames, W3C and Quitter. This scenario can be explained by at least two main reasons. First, the available datasets vary greatly in their quality. So developers have been choosing to search for links in more reliable and comprehensive datasets, such as DBpedia. This may be a safer strategy, but it narrows the potential of the WoD, as it avoids exploring less known, but more specialized datasets that could aggregate more detailed and important knowledge. The second reason refers to dataset selection, since selecting datasets with related entities is a very error-prone, arduous and time-consuming task. Several search techniques have been proposed in the literature [3, 5–7, 10, 12] to reduce the effort and increase the selection accuracy, however none of them has been widely adopted by the WoD community.

Selecting the most relevant datasets can be cast as a ranking problem, i.e., the task of ranking existing datasets  $d_i \in D$  according to the likelihood of finding entities in  $d_i$  that could be linked with the entities in  $d_t$ . Thus, it is at the user's discretion to decide which datasets to inspect or which slice of the ranking to automatically scan with a program in searching for entity links. More precisely, the problem we address is:

*Given a target dataset  $d_t$ , compute a rank score  $score(d_t, d_i)$  for each dataset  $d_i \in D$ , which induces a ranking  $(d_1, d_2, \dots, d_{|D|})$  of the datasets in  $D$  such that  $score(d_t, d_1) \geq score(d_t, d_2) \geq \dots \geq score(d_t, d_{|D|})$ . The rank score should favor those datasets with the highest probabilities of containing entities that could be linked with entities of  $d_t$ .*

The two use cases are possible in the context of WoD, i.e., either the ranked datasets would be manually selected and sent as input for further entity linking tasks or automated processes would scan the content of each dataset in an upper slice of the rank to find links, and the experiments indicated that different algorithms better suits each case.

Indeed, it is reasonable to propose an adaptable dataset search application that would deal with the two use cases differently, using distinct ranking models. By means of content negotiation, like IRI dereferencing mechanisms, human users can be distinguished from automated processes by the preferred data formats (Accept field) sent in HTTP request headers.

One can come up with three different strategies for dataset ranking: similarity ranking [3, 10]; using known dataset links and their metadata to learn linking rules [3, 5–7, 10, 12]; and identifying relevant hubs [4]. Intuitively, the first strategy suggests that the more similar two dataset descriptions are, the more likely it

will be that their contents will be similar as well. The second strategy, frequently used by recommender systems, is the collaborative filtering. It is assumed that similar groups of people share the same behavior. Of course, the similarity criterion interferes with the acknowledgement of such intuitions. For example, if two datasets are similar in their update metadata, it does not mean that they are similar in their content. The last strategy seeks highly referenced datasets, which then become authorities in certain information domains. If it is possible to identify to which information domains a dataset belongs to, hubs can be recommended as good opportunities of finding entity links. This paper examines the first two strategies, since the last one would not rank all existing datasets, but rather it would remove from the search results the non hub datasets, which implies that rankings generated with this strategy will be non comparable with the rankings of the first two strategies.

The metadata used by ranking strategies vary, but the most used are linksets, topic categories and vocabularies. They can be harvested from catalogs, such as DataHub, VoID descriptions and even from the datasets themselves. Some techniques use known linksets as features of target datasets for ranking. It can be a problem, however, if the target datasets are not yet interlinked with others. Deciding the best set of metadata for ranking is still an open problem. This paper argues that this choice will also influence the ranking model. Indeed, the experiments based on known linksets indicated that Bayesian models perform better; on the other hand, based on topic categories, rule-based classifiers would outperform Bayesian models. The performance gap can reach up to 10% at the accumulated gain (nDCG). An alternate ranking model, based on social networks, would have comparable performance to these two models, with the drawback of requiring the computation of dataset similarities. Moreover, if a dataset is already linked to others, it is better to use linksets instead of topic categories to rank them.

The contributions of this paper are an empirical analysis of five dataset ranking models, using three types of features, and a strategy to use different ranking models for the two use cases. For the first use case, the experiments indicated that the best models are those based on Bayesian and JRip classifiers and that one can use either linksets or topic categories as dataset features. Using at least 5 linksets of a dataset, the best model can improve nDCG by at least 5%, after 40% of top datasets, and even more before 40%. In the case of datasets for which no linkset is known, JRip with topic categories as dataset features would be the best choice. For the second use case, JRip would be the best model with a rank slice of 22%, 27%, and 34% at the recall levels of 70%, 80%, and 90%, respectively. The best ranking model for the first use case (with respect to nDCG) may need 13% more datasets for 90% of recall, i.e., instead of just a slice of 34% of the datasets at the top of the ranking, reached by the best model (with respect to recall@k), it would need almost 47% of the ranking.

The rest of this paper is organized as follows. Section 2 introduces the basic concepts used throughout the paper. Section 4 describes the ranking models. Section 5 addresses the preparation of the test data. Section 6 presents the exper-

iments for assessing the ranking models. Section 3 discusses related work. Finally, Sect. 7 concludes the paper.

## 2 Background Knowledge

In this section we briefly present some background definitions used throughout this paper regarding entity linking, dataset search and ranking evaluation metrics.

*RDF Dataset* – An RDF dataset, or a *dataset* for short, is a set  $d$  of RDF triples of the form  $(s, p, o)$  maintained by a single provider. The *subject*  $s$  of the triple is a global identifier (IRI), which denotes an entity of the real world, the *predicate*  $p$  is an attribute of the entity and the *object*  $o$  is an attribute value of the entity. One says that the subject  $s$  is *an entity of*  $d$ , denoted  $s \in d$ . An object can be either a literal value or an entity IRI. Triples can be accessed on the Web through IRI dereferencing (Linked Data) or via SPARQL queries, and can be stored in triplestores, relational databases, data files, or even HTML pages, thanks to RDF serialization schemes, such as RDFA.

*Linksets* – A *linkset*  $ls$  of a dataset  $d$  is a subset of RDF triples of  $d$  that link entities from two distinct datasets through a particular predicate, i.e., it is a set of triples  $(s, p, o)$  that have the same predicate  $p$ ,  $s \in d$ ,  $o \in d'$ , and  $d \neq d'$ . One says that  $(s, p, o)$  is an *entity link*,  $ls$  is a *linkset of*  $d$ ,  $d'$  is the *target of*  $ls$ , denoted  $\text{target}(ls)$ , and  $d$  is *linked with*  $d'$ . We denote the *set of all linkset targets of a dataset*  $d$  by  $L_d$ , and the *set of all linkset targets of a set of datasets*  $D$  by  $L_D = \bigcup_{d_i \in D} L_{d_i}$ . For the sake of simplicity, from here on, we refer to linkset targets simply as linksets.

Let  $ls$  be a linkset and  $dfreq(ls)$  be the number of datasets in  $D$  that have  $ls$  as linkset. We define  $tf-idf(ls)$  as follows.

$$tf-idf(ls) = \frac{|ls|}{\max(\{|ls_i| / ls_i \in L_d\})} \cdot \log \left( \frac{|D|}{dfreq(ls)} \right) \quad (1)$$

*Topic categories* – The *set of topic categories of a dataset*  $d$ , denoted  $C_d$ , is the set of topic IRIs from a particular knowledge base, e.g. DBpedia, that describe the information content of the dataset.

It can be inferred from literal values or extracted from VoID descriptions. In the case of inference, literal values are scanned with named entity recognition tools, such as DBpedia Spotlight, as proposed by Caraballo et al. [4], the recognized entities are matched with entities of a knowledge base and the topic categories associated with the entities are harvested. DBpedia, for example, associates a list of topic categories to entities through the predicate *dcterms:subject* and each category can be subsumed by others through the predicate *skos:broader*. We say that a category  $c$  is in  $C_d$  iff there exists a property path [8]  $\{e \text{ dcterms:subject/skos:broader}^* c\}$  from a named entity  $e$  to  $c$  in DBpedia. The *set of topic categories of a set of datasets*  $d_i \in D$  is  $C_D = \bigcup_{d_i \in D} C_{d_i}$ .

Topic categories can also be extracted from VoID descriptions. According to the VoID vocabulary, datasets can be partitioned by subject such that one can describe subsets of triples whose subjects are associated with a given category. In the code snippet of an example VoID file of Fig. 1, the dataset  $d$  has 100 triples containing entities associated with the topic category  $dbc:Information\_retrieval$ . The number of triples in each subset can be taken as an estimate of the occurrence frequency of the topic category for the sake of computing  $tf-idf(c)$  as follows.

---

```

@prefix dcterms: <http://purl.org/dc/terms/> .
@prefix void:   <http://rdfs.org/ns/void#> .
@prefix dbc:    <http://dbpedia.org/resource/Category:> .

<d> a void:Dataset ;
    void:subset [ a                   void:Dataset ;
                  dcterms:subject dbc:Information_retrieval ;
                  void:triples    100; ] .

```

---

**Fig. 1.** Code snippet of an example VoID file.

Let  $ocurr(D, c)$  be the number of entity occurrences in  $D$  associated with a topic category  $c$ . We define  $C'_D$  and  $C'_d$  as follows.

$$C'_D = \{c | c \in C_D \wedge o1 \leq occur(D, c) \leq o2\} \quad (2)$$

$$C'_d = (C_d \cap C'_D) \quad (3)$$

such that  $\Delta = max(\{ocurr(D, c_i)/c_i \in C_D\}) - min(\{ocurr(D, c_i)/c_i \in C_D\})$ ,  $o1 = min(\{ocurr(D, c_i)/c_i \in C_D\}) + 0.1\Delta$  and  $o2 = max(\{ocurr(D, c_i)/c_i \in C_D\}) - 0.1\Delta$ . Cutting limits were empirically chosen. The reason for narrowing category sets is that the very frequent or rare categories do not discriminate datasets appropriately, like indexing terms in traditional Information Retrieval.

Let  $ocurr(d, c)$  be the number of entity occurrences in a dataset  $d \in D$  associated with  $c$ ,  $dfreq(c)$  be the number of datasets  $d' \in D$  that have category  $c$ ,  $c \in C'_d$ ,  $c_i \in C'_d$ . We define  $tf-idf(c)$  of a dataset  $d$  as follows.

$$tf-idf(c) = \frac{ocurr(d, c)}{max(\{ocurr(d, c_i)/c_i \in C_d\})} \cdot log \left( \frac{|D|}{dfreq(c)} \right) \quad (4)$$

*Ranking evaluation* – One of most commonly used metric for ranking evaluation is the normalized Discounted Cumulative Gain (nDCG). It is a user-centric measure which expresses the degree of novelty unveiled by rankings as users go through their elements. It is computed by ranking datasets  $d_i \in D$  for a set of target datasets  $d_{t_j} \in T$ , for each of which it is known the relevance degree of  $d_i$ . Let  $rel(i)$  be the relevance degree of the  $i$ th dataset of the ranking for  $d_t$  and  $relI(i)$  be the relevance degree of an ideal ranking, which would arrange datasets decreasingly by relevance degree. nDCG is defined as follows [2].

$$DCG[i] = \frac{rel(i)}{\log(i)} + DCG[i - 1] \quad (5)$$

$$IDCG[i] = \frac{relI(i)}{\log(i)} + IDCG[i - 1] \quad (6)$$

$$nDCG[i] = \frac{DCG[i]}{IDCG[i]} \quad (7)$$

such that  $\overline{DCG}[i]$  and  $\overline{IDCG}[i]$  are averages over all  $d_t \in T$  and  $DCG[1] = rel[1]$  and  $IDCG[1] = relI[1]$ . Ranking computing models are compared by the area under the respective interpolated  $nDCG[i]$  curves. The best model has the largest area.

A second metric is Recall at Position k (recall@k), intuitively defined as the usual recall measure at each rank position. Let  $tp(i)$  be the number of relevant datasets to  $d_t$  in the first  $i$  rank positions and  $R$  be the total number of relevant datasets to  $d_t$ . Formally, recall@k is defined as follows.

$$recall[i] = \frac{tp(i)}{R} \quad (8)$$

$$recall@k[i] = \overline{recall}[i] \quad (9)$$

such that  $\overline{recall}[i]$  is the average over all  $d_t \in T$ . Ranking computing models are compared at each recall level by the size of ranking slice, the smaller the  $i$  at the same recall level, the better the ranking will be.

In order to compare rankings with different sets  $D$ , we take  $i' = i/|D|$  and compute  $nDCG[i']$  and  $recall@k[i']$ .

### 3 Related Work

Liu et al. [10] get inspiration from methods of social network analysis by computing several network measures, such as PageRank and Preferential Attachment, and use them as features for the Random Forest algorithm to classify datasets as relevant or not with respect to a given dataset. The links between datasets are defined based on known linksets of each dataset that represent equivalence links (owl:sameAS).

Martins et al. [12] adopts a content-based filtering approach based on the tokens extracted from the labels of the entities. They define that if two datasets have similar sets of tokens then it is likely that they will have related entities.

Ellefi et al. [6] propose a technique based on known linksets and topic profiles to rank relevant datasets for a given target dataset. Topic profiles are generated with the Latent Dirichlet Allocation algorithm and serve as descriptors of the datasets. Two datasets are compared with a similarity measure proportional to the amount of common linksets normalized by the total number of linksets between them. Descriptors and similarities are combined such that to penalize

datasets that resemble each other through very popular topics. Intuitively two datasets sharing very popular features would likely be less related than if it was through unpopular topics.

Emaldi et al. [7] propose a method based on comparing RDF subgraphs of two datasets. Those pairs of datasets with a greater amount of similar subgraphs were supposed to have a higher correlation of content and therefore a greater chance of containing more correlated entities.

Ellefi et al. [5] use an intentional approach that compares profiles of different datasets. The most similar profiles indicate that two datasets may contain similar entities. The profiles are obtained by representing datasets as text documents composed of words extracted from textual descriptions of the classes of their schemes (the objects of the predicates *rdf:type*) that are captured from Linked Open Vocabularies. Very common or rare classes are filtered out because they are little or very discriminatory. To reduce the set of comparisons between profiles, only profiles that have at least two classes in common are compared. The comparisons between classes are made with similarity functions applied to the class labels.

## 4 Ranking Models Used in the Experiments

This section briefly defines five ranking models and the variations used in the experiments. In what follows, let  $F_D$  be the set of distinct features of a dataset corpus  $D$  to be ranked and  $F_d$  be the set of distinct features of a single dataset,  $d$ .

### 4.1 Ranking by Cosine Similarity

The first ranking model scores datasets  $d_i \in D$  according to their similarities with a target dataset  $d_t$ . Intuitively, the more similar  $d_i$  and  $d_t$  are, the greater the likelihood that they will contain related entities. The similarity is estimated by the cosine of the angle  $\theta_{\vec{d}_t \vec{d}_i}$  between the vector representations of  $d_t$  and  $d_i$  denoted  $\vec{d}_t$  and  $\vec{d}_i$ . Therefore, the  $score(d_t, d_i)$  function is defined as follows

$$score(d_t, d_i) = \cos(\theta_{\vec{d}_t \vec{d}_i}) \quad (10)$$

The vector coordinates correspond to the distinct features  $f_i \in F_D$  and their values can be either  $tf-idf(f_i)$  or 0, if  $f_i$  does not belong to  $F_d$ . Recall that  $tf-idf(\cdot)$  over linksets and topic categories were defined in Sect. 2. We tested three feature sets:  $F_D = L_D$ ,  $F_D = C'_D$  and  $F_D = L_D \cup C'_D$ . The number of features of  $d_i$  had no limit, since it depends only on the available metadata, while the number of features of  $d_t$  was limited to 5; i.e., five linksets (5L), five categories (5C) or five linksets and five categories (5L5C), as summarized in Table 1. Other similarity scores could have been used, but this was left for future work.

**Table 1.** List of similarity-based computing ranking models.

Ranking model label	$F_D$	$score(d_t, d_i)$
cos-5L	$L_D$	$\cos(\theta_{d_t \vec{d}_i})$
cos-5C	$C'_D$	
cos-5L5C	$L_D \cup C'_D$	

## 4.2 Ranking by Preferential Attachment

The second ranking model comes from the domain of social network analysis and it was previously proposed by Lopes et al. [11]. Taking friendship as dataset links, one may transpose this approach to the context of dataset ranking, as follows [11].

$$score(d_t, d_i) = pa(d_t, d_i) = \frac{|P_{d_i}|}{|D|} \cdot \sum_{d_j \in S_{d_t} \cap P_{d_i}} \frac{1}{|P_{d_j}|} \quad (11)$$

where  $pa(\cdot, \cdot)$  is the preferential attachment metric.

Equation 11 defines that the likelihood of  $d_i$  being relevant to  $d_t$  is directly proportional to the popularity of  $d_i$  and inversely proportional to the popularity of those datasets that have  $d_i$  as one of their linksets. In this work, we defined  $S_{d_t}$ , the *similarity set* of  $d_t$ , as the set of all datasets in  $D$  that have at least 10% of the features of  $d_t$  in common. This similarity filtering was empirically defined.  $P_{d_i}$ , the *popularity set* of  $d_i \in D$ , is the set of all datasets in  $D$  that have links to  $d_i$ , and similarly  $P_{d_j}$  is the popularity set of  $d_j \in D$ . A preprocessing step computes  $P_{d_i}$  from  $L_D$ , which must be given. We also tested different feature sets and limited the number of features for  $d_t$  to 5 or 12, as summarized in Table 2. Similarly to the first ranking model, 5L means that  $d_t$  has five linksets, 12C means that  $d_t$  has twelve categories and 5L12C means that  $d_t$  has five linksets and twelve categories.

**Table 2.** List of social-network-based computing ranking models.

Ranking model label	$F_D$	$score(d_i, d_t)$
sn-5L	$L_D$	$pa(d_i, d_t)$
sn-12C	$C'_D$	
sn-5L12C	$L_D \cup C'_D$	

## 4.3 Ranking by Bayesian Probabilities

The third ranking model is inspired by Bayesian classifiers and was previously proposed by Leme et al. [9]. It computes the probability that  $d_i$  is relevant to  $d_t$

given that  $d_t$  has features  $f_i \in F_d$ . The naive assumption on the joint probability of having multiple features induces the following score function.

$$score(d_t, d_i) = P(d_i|d_t) = \left( \sum_{j=1..n} log(P(f_j|d_i)) \right) + log(P(d_i)) \quad (12)$$

$P(f_j|d_i)$  is the probability that a dataset has feature  $f_j$  if it is linked to  $d_i$ , and  $P(d_i)$  is the probability of  $d_i$  being a linkset. A preprocessing step computes probabilities from  $L_D \cup C'_D$ , which must be given. We also tested different feature sets as summarized in Table 3 with the same notation conventions used in previous models.

**Table 3.** List of Bayesian computing ranking models.

Ranking model label	$F_D$	$score(d_i, d_t)$
bayesian-5L	$L_D$	$prob(d_i, d_t)$
bayesian-12C	$C'_D$	
bayesian-5L12C	$L_D \cup C'_D$	

#### 4.4 Ranking with Rule Classifiers

The last two ranking models use the machine learning algorithms C4.5 and RIPPERk through their respective Java implementations J48 and JRip in the Weka Toolkit [19]. They are rule-based classification algorithms that learn conjunctive rules from vector representations of  $d_i \in D$ . The algorithms differ in the pruning heuristics of the decision tree, which may impact computing and classification performances. Each learned rule  $R_j^C$  for a class  $C$  has an associated probability  $P_{R_j^C}$  which estimates the confidence of classifying an instance as being of the class  $C$  with  $R_j^C$ . We trained a set of binary classifiers for the classes  $d_i$  and  $\neg d_i$ , such that  $d_i \in D$ . Classifying a target dataset  $d_t$  as an instance of a class  $d_i$  means that  $d_t$  may have entity links to  $d_i$ , i.e.,  $d_i$  may be a linkset of  $d_t$ , i.e.,  $d_i$  is the target of a linkset of  $d_t$ . We defined  $score(d_t, d_i)$  function as follows

$$score(d_t, d_i) = \begin{cases} P_{R_j^{d_i}} & \text{if } d_t \in d_i \\ 1 - P_{R_j^{\neg d_i}} & \text{if } d_t \in \neg d_i \end{cases} \quad (13)$$

such that  $j$  is the rule index for which  $R_j^{d_i}$  OR  $R_j^{\neg d_i}$  applies to  $d_t$  and that has the biggest  $P_{R_j^C}$ . Classifiers were trained with sets of positive and negative examples of each class. Positive examples of the class  $d_i$  are datasets that have  $d_i$  as one of their linksets and negative examples are the opposite. The feature sets are summarized in Table 4 with the same notation conventions for model labels.

**Table 4.** List of rule-based computing ranking models.

Ranking model label	$F_D$	$score(d_i, d_t)$
j48-5L	$L_D$	
jrip-5L		
j48-12C	$C'_D$	$pRule(d_i, d_t)$
jrip-12C		
j48-5L12C	$L_D \cup C'_D$	
jrip-5L12C		

## 5 Data Preparation and Methodology

The data for the experiments [14] is a collection of VoID descriptions of the datasets in the LOD Cloud.

DataHub is a catalog of open data used by the Linked Data community to disseminate metadata about the datasets available in the LOD Cloud. This catalog is built on top of the Comprehensive Knowledge Archive Network (CKAN) platform that has a RESTful API through which one can browse the content of the catalog. Datasets that do not belong to the LOD Cloud have been disregarded in this paper. Among others, the available metadata on the catalog are linksets, SPARQL endpoints and dumps. The CKAN adopts DCAT as the standard metadata scheme, but some conventions allowed to record particularities of RDF datasets. The following example of an HTTP request returns a JSON document doc with metadata of the Association for Computing Machinery (ACM) dataset, where  $m = \text{doc}[\text{'result'}][\text{'results'}][0]$  is a dictionary with the metadata itself.

[https://datahub.ckan.io/api/3/action/package\\_search?fq=name:rkb-explorer-acm](https://datahub.ckan.io/api/3/action/package_search?fq=name:rkb-explorer-acm)

Linksets can be identified in  $m$  with two structures of different formats, but with similar contents, which are  $ls1 = m[\text{'relationships\_as\_subject'}]$  and  $ls2 = m[\text{'extras'}]$ . In  $ls1$ , the target dataset of a linkset is identified by its local ID  $ls1[i][\text{'id'}]$ , where  $i$  is an index of the linksets' vector, and the number of triples is  $ls1[i][\text{'comment'}]$ . In  $ls2$ , the target dataset is  $ls2[\text{'key'}]$  and the number of triples is  $ls2[\text{'value'}]$ .

The metadata of each dataset was enriched with topic categories as follows. Let  $e$  be a named entity recognized in literal values of the dataset. A topic category  $c$  should be associated with the dataset if and only if there exists a path  $\{e \text{ } dcterms:subject/skos:broader^* c.\}$  between  $e$  and  $c$  in DBpedia. Named entities recognition was performed with DBpedia Spotlight, which is also available through a RESTful API. Topic categories were annotated as subsets of the datasets according to the pattern in Fig. 1.

Datasets without available dumps were not annotated with topic categories. Both linksets and datasets were annotated with their respective number of triples.

There is a total of 1,113 datasets with at least one linkset, from which 348 datasets have more than 8 linksets and 153 have more than 8 linksets and some topic category. This filtering was necessary to select appropriate datasets for the ranking models. The usable sets of datasets were randomly partitioned into three groups for a 3-fold cross validation.

The set of 1,113 datasets was divided into three equal parts  $P_1$ ,  $P_2$  and  $P_3$  and, in a 3-fold cross validation process, the datasets in two parts  $P_i$  and  $P_j$  were ranked for each dataset of the third part  $P_k$ , which was taken as the set  $T$  of target datasets. Recall from Sect. 2 that nDCG and recall@ can be computed for a set of target datasets  $P_k$  as the mean of these measures for the datasets in  $P_k$ . The consolidated cross-validation result is the mean of nDCG and recall@ for  $k \in 1, 2, 3$ . This process was repeated for each of the proposed ranking models.

For each dataset in  $P_k$ , it was created a representation based on its available characteristics which was used as input for the ranking algorithms. Remember from Sect. 4 that these representations can be based on linksets, categories, and a combination of the two.

Notice that DataHub stores a list of known linksets for each dataset in the LOD Cloud. The targets of these linksets (objectsTarget - VoID) are, by definition, datasets with which there are links and, therefore, are the datasets that one would wish to find in higher ranking positions, i.e., they are the set of relevant datasets, denoted  $R$ . Only when a representation of a target dataset ( $d_t$ ) includes a linkset, the objectsTarget of that linkset must be removed from  $R$ .

## 6 Experiments

We refer the reader to Neves et al. [17] for the full set of ranking evaluations. This section presents results for the best ranking models.

Recall from Sect. 4 that we consider two use cases for dataset rankings. In the first use case, datasets are manually selected and users intuitively focus on the initial ranking positions. Comparing ranking models with nDCG would unveil models with the highest gain rate of relevance. In order to compute nDCG, it is necessary, however, to define the degree of relevance of each entry of the ranking. Let

- $D$  be a dataset corpus to be ranked
- $L_{d_t}$  the linksets of a target dataset as extracted from DataHub
- $F_{d_t}$  the feature set of  $d_t$
- $R = (D \cap L_{d_t}) - F_{d_t}$ , be the datasets relevant to  $d_t$  in  $D$
- $\text{triples}(r_i)$  be the number of triples of the linkset  $ls$  of  $d_t$  that has  $\text{target}(ls) = r_i$
- $T_1 = \min(\text{triples}(r_i)/r_i \in R)$
- $T_2 = \max(\{\text{triples}(r_i)/r_i \in R\})$
- $\Delta = (T_2 - T_1)/3$

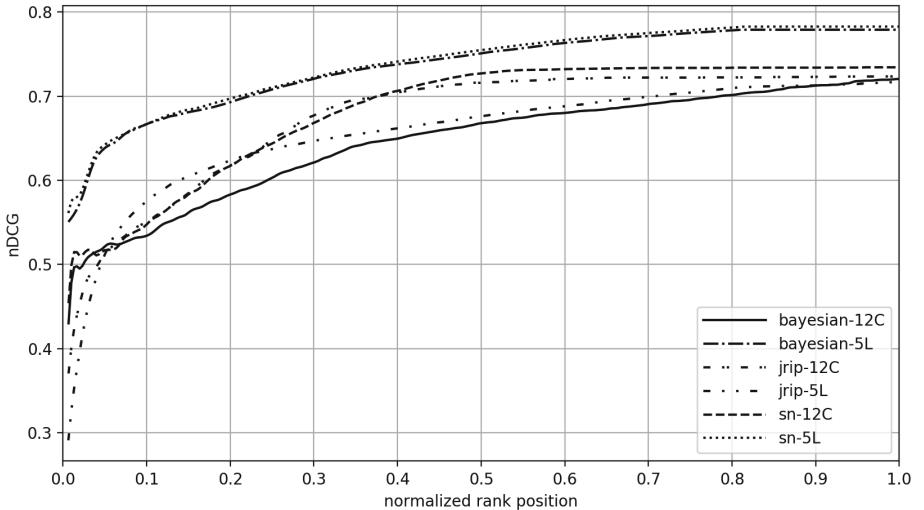
Notice that  $R$  is the set of datasets which are taken as unknown linksets and that must be better positioned in the ranking. The degree of relevance of  $d_i \in D$  to  $d_t$  is defined as follows.

$$\begin{aligned}
rel(d_i) = 0, d_i \notin R \\
rel(d_i) = 1, d_i \in R \wedge T_1 \leq triples(r_i) < T_1 + \Delta \\
rel(d_i) = 2, d_i \in R \wedge T_1 + \Delta \leq triples(r_i) < T_1 + 2\Delta \\
rel(d_i) = 3, d_i \in R \wedge T_1 + 2\Delta \leq triples(r_i) \leq T_2
\end{aligned}$$

In the second use case, programs would scan a slice of the ranking in search of entity links. In such cases, the best models would be those that would provide the best recall@k for the same ranking size.

The use of different feature sets causes  $D$  to have different sizes depending on the ranking model, that is, not all datasets have all possible feature sets. To compare rankings with different sizes we compute  $nDCG(i')$  and  $recall@k(i')$ , where  $i' = i/|D|$ , we call  $i'$  as the normalized rank position.

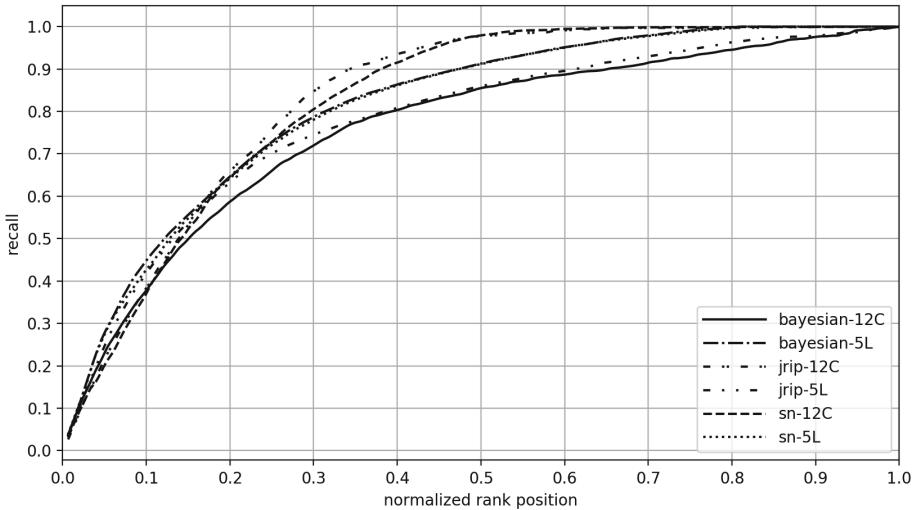
Figure 2 shows that the best models for the first use case, based on nDCG. Traditional use of rankings are those based on Bayesian classifiers, Social Network and JRip classifiers. One can see that knowing at least 5 linksets of a dataset can improve at least 5%, after 40% of top datasets (normalized rank position = 0.4), and even more before 40%. In the case of datasets for which no linkset is known, the best it can be done is to use topic categories with JRip or Social Network ranking models. Ranking models with a mixed set of features (Linksets and Topic Categories) did not achieve comparable performances [17]. This is an important outcome of the experiments. Moreover, as Bayesian and JRip approaches have ranking performances very similar to that of the Social Network (SN), one can avoid computational cost of the similarity calculations needed for SN.



**Fig. 2.** nDCG of the best ranking computing models.

Figure 3 shows the best models for the second use case, based on recall@k. Note that, after 20% of the top datasets, the rankings start diverging in performance. As the average size of the ranking is 143, it means that the divergence

starts at the 28th position, on average. For a recall of 70%, bayesian-12C would need 28% of the top datasets, while jrip-12C would need 22%. For a recall level of 80%, the bayesian-12C would need 40% of top datasets, while the jrip-12C would require just 27%. The difference would be even greater at 90% of recall: the bayesian-12C would need 65% of the top ranking, while jrip-12C would need just 34%. The best ranking model for the first use case (with respect to nDCG) may need 13% more datasets for 90% of recall, i.e., instead of just a slice of 34% of the datasets at the top of the ranking, reached by the best model (with respect to recall@k), it would need almost 47% of the ranking.



**Fig. 3.** Recall@k of the best ranking computing models.

We can then conclude that if one wants to exhaustively examine rankings looking for entity links, one would better use jrip-12C as the ranking model. Besides better performance, JRip with topic categories has the advantage that it does not depend on the assumption that all datasets would have known linksets, but only on the existence of topic categories, which can be frequently provided for a dataset. Moreover, the results pose empirical limits for sizing the slice of the ranking depending on the desired recall level, for example, if one wants to find 80% of the linksets of a target dataset, Fig. 3 shows that a program can be coded to scan only the top 27% of the ranking, for a recall of 70% it would scan just 22%, and so on.

## 7 Conclusions and Future Work

The growth of the Web of Data strongly depends on entity interlinking, as the traditional Web depends on hyperlinks. Current strategies, which focus only on

well known datasets, although safe, overlook important opportunities for entity interlinking. The dataset ranking techniques discussed in this paper strongly facilitate this task, since they can reduce the computational effort of searching links and unveiling important datasets.

This paper presented an empirical comparison of several ranking models in order to identify the conditions in which they are best applied. The first conclusion is that, for human interactions with a dataset search tool, the best ranking models (with respect to nDCG) are based on Bayesian classifiers and JRip. Bayesian is preferable when one knows linksets, since it can have the nDCG at least 5% greater, otherwise JRip is the best choice. Secondly, the similarity computation of social network approach can be avoided, since Bayesian and JRip have similar performances. Thirdly, we can conclude that models with the JRip classifier and topic categories are always desirable, when one wants to automatically scan rankings. Besides better performance (with respect to recall@k), 13% less datasets for 90% of recall, JRip with topic categories has the advantage that it does not depend on the assumption that all datasets would have known linksets, but only on the existence of topic categories, which can be frequently provided a dataset. Finally, The experiments also indicated the ranking size to be traversed for each desired level of recall, which may be taken as input of the search. For a recall level of 70% scan 22% of the ranking, for a recall level of 80% scan 27%, for a recall level of 90% scan 34%, and so on.

One limitation of the experiments was the amount of data available. The lack of availability of dataset samples (dumps) did not allow the use of all data obtained from DataHub. Expanding this availability and comparing other proposed methods may bring new conclusions to the design of dataset ranking methods with the purpose of entity interlinking.

**Acknowledgments.** This work has been funded by FAPERJ/BR under grants E-26/010.000794/2016, E-26/201.000337/2014 and CNPq under grant 303332/2013-1.

## References

1. Abele, A., McCrae, J.P., Buitelaar, P., Jentzsch, A., Cyganiak, R.: Linking open data cloud diagram 2017. Technical report, Insight Centre for Data Analytics at NUI Galway (2017). <http://lod-cloud.net>
2. Baeza-Yates, R.R., Ribeiro-Neto, B.: Modern Information Retrieval: The Concepts and Technology Behind Search, 2nd edn. ACM Press, New York (2011)
3. Caraballo, A.A.M., Arruda, N.M., Nunes, B.P., Lopes, G.R., Casanova, M.A.: TRTML - a tripleset recommendation tool based on supervised learning algorithms. In: Presutti, V., Blomqvist, E., Troncy, R., Sack, H., Papadakis, I., Tordai, A. (eds.) ESWC 2014. LNCS, vol. 8798, pp. 413–417. Springer, Cham (2014). [https://doi.org/10.1007/978-3-319-11955-7\\_58](https://doi.org/10.1007/978-3-319-11955-7_58)
4. Caraballo, A.A.M., Nunes, B.P., Lopes, G.R., Leme, L.A.P.P., Casanova, M.A.: Automatic creation and analysis of a linked data cloud diagram. In: Cellary, W., Mokbel, M.F., Wang, J., Wang, H., Zhou, R., Zhang, Y. (eds.) WISE 2016. LNCS, vol. 10041, pp. 417–432. Springer, Cham (2016)

5. Ellefi, M.B., Bellahsene, Z., Dietze, S., Todorov, K.: Dataset recommendation for data linking: an intensional approach. In: Sack, H., Blomqvist, E., d'Aquin, M., Ghidini, C., Ponsetto, S.P., Lange, C. (eds.) ESWC 2016. LNCS, vol. 9678, pp. 36–51. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-34129-3\\_3](https://doi.org/10.1007/978-3-319-34129-3_3)
6. Ellefi, M.B., Bellahsene, Z., Dietze, S., Todorov, K.: Beyond established knowledge graphs-recommending web datasets for data linking. In: Bozzon, A., Cudre-Marouf, P., Pautasso, C. (eds.) ICWE 2016. LNCS, vol. 9671, pp. 262–279. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-38791-8\\_15](https://doi.org/10.1007/978-3-319-38791-8_15)
7. Emaldi, M., Corcho, O., López-De-Ipiña, D.: Detection of related semantic datasets based on frequent subgraph mining. In: Proceedings of the Intelligent Exploration of Semantic Data (IESD 2015) (2015)
8. Harris, S., Seaborne, A.: SPARQL 1.1 query language. Technical report, W3C (2013)
9. Leme, L.A.P.P., Lopes, G.R., Nunes, B.P., Casanova, M.A., Dietze, S.: Identifying candidate datasets for data interlinking. In: Daniel, F., Dolog, P., Li, Q. (eds.) ICWE 2013. LNCS, vol. 7977, pp. 354–366. Springer, Heidelberg (2013). [https://doi.org/10.1007/978-3-642-39200-9\\_29](https://doi.org/10.1007/978-3-642-39200-9_29)
10. Liu, H., Wang, T., Tang, J., Ning, H., Wei, D.: Link prediction of datasets sameAS interlinking network on web of data. In: Proceedings of the 3rd International Conference on Information Management (ICIM 2017), pp. 346–352 (2017)
11. Lopes, G.R., Leme, L.A.P.P., Nunes, B.P., Casanova, M.A., Dietze, S.: Two approaches to the dataset interlinking recommendation problem. In: Proceedings of the 15th International Conference on Web Information Systems Engineering (WISE 2014), pp. 324–339 (2014)
12. Martins, Y.C., da Mota, F.F., Cavalcanti, M.C.: DSCrank: a method for selection and ranking of datasets. In: Garoufallou, E., Subirats Coll, I., Stellato, A., Greenberg, J. (eds.) MTSR 2016. CCIS, vol. 672, pp. 333–344. Springer, Cham (2016). [https://doi.org/10.1007/978-3-319-49157-8\\_29](https://doi.org/10.1007/978-3-319-49157-8_29)
13. Nentwig, M., Hartung, M., Ngonga Ngomo, A.C., Rahm, E.: A survey of current link discovery frameworks. Semant. Web **8**(3), 419–436 (2016)
14. Neves, A.B., Leme, L.A.P.P.: Dataset Descriptions. figshare (2017). <https://doi.org/10.6084/m9.figshare.5211916>
15. Ngomo, A.C.N., Auer, S.: LIMES - a time-efficient approach for large-scale link discovery on the web of data. In: Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI 2011), pp. 2312–2317 (2011)
16. Nikolov, A., Uren, V., Motta, E.: KnoFuss: a comprehensive architecture for knowledge fusion. In: Proceedings of the 4th International Conference on Knowledge Capture (K-CAP 2007), pp. 185–186 (2007)
17. Oliveira, R.G.G., Neves, A.B., Leme, L.A.P.P., Lopes, G.R., Nunes, B.P., Casanova, M.A.: Empirical Analysis of Ranking Models for an Adaptable Dataset Search: Complementary Material. figshare (2017). <https://doi.org/10.6084/m9.figshare.5620651>
18. Volz, J., Bizer, C., Gaedke, M., Kobilarov, G.: Discovering and maintaining links on the web of data. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) ISWC 2009. LNCS, vol. 5823, pp. 650–665. Springer, Heidelberg (2009). [https://doi.org/10.1007/978-3-642-04930-9\\_41](https://doi.org/10.1007/978-3-642-04930-9_41)
19. Witten, I.H., Frank, E., Hall, M.A., Pal, C.J.: Data Mining: Practical Machine Learning Tools and Techniques, 4th edn. Morgan Kaufmann Publishers Inc., Burlington (2016)