ReDyAl: A Dynamic Recommendation Algorithm based on Linked Data*

Iacopo Vagliano*, Cristhian Figueroa*§, Oscar Rodríguez Rocha†,
Marco Torchiano*, Catherine Faron-Zucker‡, Maurizio Morisio*

*Dept. Control and Computer Engineering, Politecnico di Torino, Turin, Italy
{iacopo.vagliano, cristhian.figueroa, marco.torchiano, maurizio.morisio}@polito.it
§ Universidad del Cauca, Popayán, Colombia
† INRIA Sophia Antipolis Méditerranée, Sophia Antipolis, France
oscar.rodriguez-rocha@inria.fr
‡ Univ. Nice Sophia Antipolis, CNRS, I3S, UMR 7271, Sophia Antipolis, France
faron@i3s.unice.fr

ABSTRACT

The Web of Data is an interconnected global dataspace in which discovering resources related to a given resource and recommend relevant ones is still an open research area. This work describes a new recommendation algorithm based on structured data published on the Web (Linked Data). The algorithm exploits existing relationships between resources by dynamically analyzing both the categories to which they belong to and their explicit references to other resources. A user study conducted to evaluate the algorithm showed that our algorithm provides more novel recommendations than other state-of-the-art algorithms and keeps a satisfying prediction accuracy. The algorithm has been applied in a mobile application to recommend movies by relying on DBpedia (the Linked Data version of Wikipedia), although it could be applied to other datasets on the Web of Data.

Keywords

Recommender System, Linked Data, DBpedia, Semantic Web

1. INTRODUCTION

The Web is evolving from an information space for sharing textual documents into a medium for publishing structured data. Linked Data[1] is a set of best practices to publish and interlink data on the Web and it is the base of the Web of Data, an interconnected global dataspace where data providers publish their content publicly.

Due to the increase in the amount of structured data published on the Web through the principles of Linked Data, it is more likely to find resources that describe or represent real life concepts. The information provided by these resources may be used in many different domains. However, finding and recommending related resources is still an open research area [19]. The work presented in this paper holds on the results obtained from an our previous study and is its continuation[6]. The study stated that the problem of finding existing relationships between resources can be addressed by analyzing the categories they belong to, their explicit references to other resources and/or by combining both these approaches. The study also showed that many works aimed at resolving this problem by focusing on a specific application domain and dataset. In this paper, we address this issue and we focus on the following research questions: (i) *How can we design a recommendation algorithm that exploits existing relationships between resources on the Linked Data, is independent from the application domain and may be used on different datasets on the Web of Data?* (ii) *How can we design a recommendation algorithm that provides novel recommendations, i.e., recommendations of resources not previously known from the user, without affecting the prediction accuracy of the algorithm?*

We propose a new algorithm based on Linked Data which exploits existing relationships between resources in order to recommend related resources. It dynamically analyzes the categories they belong to and their explicit references to other resources, then combines the results. The algorithm has been applied to DBpedia[2], but it could as well be applied to other datasets on the Web of Data and it is not bound to any specific application domain.

We conducted a user study to comparatively evaluate its accuracy and novelty against three state-of-the-art algorithms, which showed that our algorithm provides a higher number of novel recommendations, while keeping a satisfying prediction accuracy. An implementation of our recommendation algorithm has been integrated into a mobile application suggesting movies based on DBpedia, which was developed in collaboration with Telecom Italia, the major network operator in Italy.

*The mobile application presented in Section 5 was done at the Joint Open Lab MobiLAB and was supported by a fellowship from TIM.

[1] `http://linkeddata.org`

[2] `http://dbpedia.org`

CBRecSys 2016, September 16, 2016, Boston, MA, USA.

The paper is organized as follows: Section 2 reviews related works; Section 3 presents our algorithm; Section 4 describes the evaluation method and provides the results; Section 5 shows the application of our algorithm for recommending movies; Section 6 provides conclusions.

2. RELATED WORK

This work began by conducting a systematic literature review [6] on the subject, which allowed us to lay the groundwork for this research. Such review listed the different approaches to exploit Linked Data in order to recommend resources. Some studies found, infer relationships between resources by taking into account the existing links between them in a dataset, and use these relationships to measure the semantic similarity of the resources. Such relationships can be direct links, paths, or shared topics between sets of items. The most important related works are summarized in the following.

Damljanovic et al. [4] recommended experts in an open innovation scenario. Their approach, named *HyProximity*, takes as input a description of a problem in natural language and extracts a set of relevant words that are linked with resources of DBpedia. Then it generates recommendations by combining two techniques. The first one consists in discovering resources related through hierarchical relationships, while the second one is based on traversal relationships, which connect resources without establishing a classification or hierarchy. By exploiting these two kinds of relationships, the approach identifies a set of direct or indirect topics related with potential experts to solve an innovation problem.

Passant [17] described *dbrec*, a recommender targeted for the music domain, which mainly relies on a distance measure named Linked Data Semantic Distance (LDSD). It takes into account the number of direct or indirect links between resources (related with the music domain) represented in DBpedia. Unlike HyProximity it does not distinguish between traversal and hierarchical links. Both Damljanovic et al. and Passant had to reduce the set of resources and links of the dataset to those belonging to a specific domain (innovation problems and music respectively), which involves a huge effort to manually define which resources or links should be considered.

Other works combine Linked Data based algorithms with other techniques of recommendation in order to improve the results. These techniques include collaborative filtering [10, 14, 16, 18], information aggregation [2, 9, 12] and statistical methods like Random Indexing (RI) [23], Vector Space Model (VSM) [1, 16], Latent Dirichlet Allocation (LDA) [11], implicit feedback [16] and structure-based statistical semantics [3]. De Graaff et al. [5] proposed a knowledge-based recommender system that derives the user interests from the users social media profile, which is enriched with information from DBpedia. Musto et al. [15] compared several techniques to automatically feed a graph-based recommender system with features extracted from Linked Data. However, these techniques usually require additional information from the user in order to produce accurate recommendations.

We propose a new recommendation algorithm, which is cross-domain and cross-dataset. It relies only on Linked Data and does not require to reduce the set of resources and links of the dataset to those belonging to a specific domain.

3. ReDyAl

ReDyAl is a recommendation algorithm which takes into account the different types of relationships between the data published according to the Linked Data principles. It aims at discovering related resources from datasets that may contain either *well-linked* resources as well as *poorly-linked* resources. A resource is said to be well-linked if it has a number of links higher than the average number of links in the dataset; otherwise it is poorly-linked. The algorithm is able to dynamically adapt its behavior in order to find a set of candidate resources to be recommended, relying on the implicit knowledge contained in the Linked Data relationships.

3.1 Principles

Any dataset on the Web of Data may be seen as a tuple *(R, T, L)* composed by resources (R), categories (T), and relationships (L). Categories denote types, concepts or classes. Resources are instances of concepts; they are Web resources or real world resources identified by a URI. Relationships are also known as links or properties; they are the links connecting resources or categories along the whole dataset graph. Categories often are hierarchically organized. For example, DBpedia provides information about hierarchical relationships in three different classification schemata: Wikipedia Categories, YAGO[3] [24] classes, and WordNet Synsets[4]. Relationships can be of three types:

Resource-Resource (*R-R*) These are the traversal relationships between resources, i.e. the links between resources that do not refer to hierarchical classifications.

Resource-Category (*R-T*) These are relationships between a resource and a category. They can be represented by the RDF[5] property `rdf:type` or the `dcterms:subject` property from the Dublin Core vocabulary[6].

Category-Category (*T-T*) These are hierarchical relationships between categories within a hyponymy structure (a category tree). They can be represented by using the RDFS[7] property `rdfs:subClassOf` or the SKOS[8] properties `skos:broader` (isSubCategoryOf) and `skos:narrower` (isSuperCategoryOf).

Considering this model of a dataset, ReDyAl consists in three stages:

1. The first stage discovers resources by analyzing the links between the given initial resource and other resources. Only R-R relationships are considered at this stage, although they can be indirect, i.e. they can connect two resources through a third one.

[3] `http://www.mpi-inf.mpg.de/yago-naga/yago/`
[4] `https://wordnet.princeton.edu`
[5] `http://www.w3.org/TR/rdf11-concepts/`
[6] `http://dublincore.org/documents/dcmi-terms/`
[7] `http://www.w3.org/TR/rdf-schema/`
[8] `https://www.w3.org/TR/skos-reference`

2. The second stage analyzes the categorization of the given initial resource and discovers similar resources located in the same categories. It finds indirect relationships between resources through direct R-T and T-T relationships. It is possible to specify to the algorithm which specific R-T and T-T relationships to consider in this step: the choice for R-T relationships is between `dcterms:subject` or `rdf:type`, while `skos:broader` and `skos:narrower` or `rdfs:subClassOf` are acceptable T-T relationships.

3. The last stage intersects the results of both the previous stages and ranks them by giving priority to those found in the first stage. The algorithm computes the similarity of the initial resource with respect to any of the discovered resources, based on a similarity function which combines the Linked Data Semantic Distance (LDSD) [17] and HyProximity distance [4], opportunely adapted and generalized.

The algorithm can be applied to any dataset on the Web of Data. In the first step, it relies only on R-R relationships: any relationship of this kind may be used, independently of the data stored on the dataset. In the second step, the algorithm can be configured to use the `dcterms:subject` or `rdf:type` properties, which are R-T relationships. DBpedia uses both to enable different categorizations; for example to rely on the Wikipedia categories, it is necessary to set `dcterms:subject` as R-T relationship and `skos:broader` and `skos:narrower` as T-T relationships. Any other dataset uses at least `rdf:type` to indicate the class which a resource is instance of. Thus, `rdf:type` can be used to find resources in the same class and then `rdfs:subClassOf` can be used to retrieve more general classes (or `skos:broader` and `skos:narrower`, if the categories are organized through SKOS properties).

The algorithm is independent on the application domain because it relies only on R-R, R-T or T-T links. If in the dataset on which the algorithm is applied there are relationships among resources in different domains the algorithm may generate cross-domain recommendations. For example, DBpedia is a general dataset which represents resources of different kind and there may be a relationship between a song and a city because the song was recorded in that city, or because is about the city. Alternatively, there may be a link between a song and a movie because the song was part of the soundtrack of the movie. Thus, a city or a movie may be recommended starting from a song. Also R-T links may generates cross-domain recommendations if resources which belong to different domains are included into the same category.

3.2 Reducing the search space

Additionally, the algorithm may be configured with a set of forbidden links in order to restrict the kind of links the algorithm should consider. This is useful to prevent the algorithm to obtain resources over links pointing to empty nodes (i.e. resources without a URI), literals that are used to identify values such as numbers and dates, and other nodes that are not desired for the recommendation. In other words, it is a way to limit the results of the algorithm. For example the DBpedia resource `dbr:Turin` contains the link `dbpprop:populationTotal` that points to the integer value 911823: we can confgure this link as forbidden link since it does not point to a resource which can be recommended. This is also useful to increase the performance of the algorithm because limiting the number of results decreases the ranking time. All the links which are not explicitly specified as forbidden are allowed links and define a domain of interest. This may be useful when the algorithm is applied to a generic dataset as DBpedia. This dataset contains millions of links between resources, and if a developer is creating an application in the music domain then he/she may be interested only in resources of that domain, so he/she may want to consider only links pointing to those resources i.e., a set of allowed links. In fact the algorithm is cross-domain, thus it may recommend a city or a movie starting from a song, as we have already explained. While this may be an advantage in some applications, it may be confusing in others, especially if not properly explained to the user. To limit the recommendations to specific categories of resources (for example to consider only tracks and artists) it is sufficient to "allow" only the relationships which point to these kinds of resources, i.e. which have such desired category as range.

3.3 Parameter Settings

ReDyAl receives as input an initial resource by specifying its corresponding URI ($inURI$), and three values ($minT$, $minC$, $maxDistance$) for configuring its execution. The selection of $minT$ and $minC$ is arbitrary and depends on the dataset and the convenience of the user who is setting up the algorithm. $minT$ is the minimum number of links (input and output links involving the initial resource) necessary to consider a resource as well-linked. The proper value of $minT$ depends on the dataset: if it contains resources with a high number of links between them it is expected to be higher, while if the resources have only few links it should be set to a lower value. However, this parameter impacts on the algorithm: if the initial resource is well-linked, traversal interlinking has a higher priority in the generation of candidate resources, otherwise the algorithm gives priority to the hierarchical relationships. For example, a user may consider the use of the hierarchical algorithms only if the resources are connected with less than 10 links by setting $minT$ to 10. In a similar way, the user may arbitrary fix the value of $minC$, which is the minimum number of candidate resources that the algorithm is expected to generate, i.e. the number of candidate resources the user is expecting.

The value of $maxDistance$ limits the distance (i.e. the number of hierarchical levels) that the algorithm considers in a category tree. $maxDistance$ may be defined manually; this is particularly useful when there are not enough candidate resources from the categories found at a certain distance (i.e. the number of candidate resources retrieved is lower than $minC$). In this case, the algorithm increases the distances in order to find more resources and if the $maxDistance$ value is reached with less than $minC$ candidate resources, the algorithm ranks only the candidate resources found until that moment. Additionally, the algorithm may receive a list of forbidden links (FL) to avoid searching for candidate resources over a predefined list of undesired links.

3.4 Algorithm

Algorithm 1 ReDyAl algorithm

Require: $inURI$, $minT$, $minC$, FL, $maxDistance$,
Ensure: A set of candidate resources CR

```
1: L_in = readAllowedLinks(inURI, FL)
2: if |L_in| ≥ minT then
3:    for all l_k ∈ L_in do
4:       DRl_k = getDirectResources(l_k)
5:       IRl_k = getIndirectResources(l_k)
6:       Add DRl_k to CR_tr
7:       Add IRl_k to CR_tr
8:    end for
9:    if |CR_tr| ≥ minC then
10:      return CR_tr
11:   else
12:      currentDistance = 1
13:      Gc = createCategoryGraph(inURI, currentDistance)
14:      while currentDistance ≤ maxDistance do
15:         CR_hi = getCandidateResources(Gc)
16:         if |CR_hi| ≥ minC then
17:            Add CR_tr and CR_hi to CR
18:            return CR_hi
19:         end if
20:         increase currentDistance
21:         updateCategoryGraph(currentDistance)
22:      end while
23:      Add CR_tr and CR_hi to CR
24:   end if
25: end if
26: return CR
```

ReDyAl (Algorithm 1) starts by retrieving a list of allowed links from the initial resource. Allowed links are those that are not specified as forbidden (FL) or that are explicitly defined in the initial resource. If there is a considerable number of allowed links (more than $minT$, i.e., the initial resource is well-linked) the algorithm obtains a set of candidate resources located through direct (DRl_k) or indirect traversal links (IRl_k), starting from the links explicitly defined in the initial resource (Lines 1-8). A resource is indirectly linked to the initial resource if it is linked through another resource. A resource directly linked is located at traversal distance 1 from the initial resource, while a resource indirectly linked is located at traversal distance 2 from the initial resource. With regards to the traversal links, a maximum distance of 2 is considered because for distances higher than 2 (i.e 1 direct heap plus 1 indirect heap) the number of retrieved resources is dramatically increased, therefore increasing also the number of resources that are not relevant or related with the initial resource.

Next, if the current number of candidate resources generated (CR_{tr}) is greater than or equal to $minC$, the algorithm terminates returning the results (Lines 9-10). Otherwise, the algorithm generates a category graph (Gc) with categories of the first distance and applies iterative updates over the category graph over n distances from the initial resource, obtaining broader categories (i.e. more generic categories that are located in a higher level in a classification) until at least one of two following conditions is fulfilled: the number of candidate resources is sufficient ($|CR| > minC$), or the maximum distance is reached ($currentDistance > maxDistance$). At each iteration, candidate resources (CR_{hi}) are extracted from the broader categories of maximum distance (Lines 14-23). In any case, the algorithm combines these results with the results obtained in Lines 3-8 (adding CR_{tr} and CR_{hi} to CR). Finally, the set of candidate results is returned (Line 23).

3.5 Ranking of the recommended resources

The final operation is ranking the sets of candidate resources. The ranking process receives as input the candidate resources retrieved by the ReDyAl algorithm and ranks them according to their degree of similarity with the initial resource. This similarity is computed based on a combination of two distance measures: LDSD and HyProximity.

The LDSD distance, initially proposed by Passant [17], is based on the number of indirect and direct links between two concepts. In this measure, the similarity of two resources (r_1, r_2) is measured by combining four properties: the input/output direct links or the input/output indirect links between them. Equation 1 presents the basic form of the $LDSD$ distance. Cd_{out} is the number of direct output links (from c_1 to c_2), Cd_{in} is the number of direct input links, Ci_{in} is the number of indirect input links, and Ci_{out} is the number of indirect output links. The implementation developed by Passant is limited to links from a specific domain, while the LDSD function implemented in ReDyAl takes into account all the concepts of the dataset unless forbidden links are specified.

$$LDSD(c_1, c_2) = \frac{1}{1 + Cd_{out} + Cd_{in} + Ci_{out} + Ci_{in}} \quad (1)$$

HyProximity is a similarity measure defined by Stankovic et al. [4], which can be used to calculate both traversal and hierarchical similarities. The measure in its general form is shown in Equation 2 as the inverted distance between two concepts, balanced with a pondering function. In this equation $d(r_1, r_2)$ is the distance function between the resources r_1 and r_2, while $p(r_1, r_2)$ is the pondering function, which is used to weight different distances. Based on the structural relationships (hierarchical and traversal), different distance and pondering functions may be used to calculate the HyProximity similarity. ReDyAl reuses the HyProximity hierarchical measure, which is the quotient of a pondering function (p) and a distance (d). The distance was calculated using $maxDistance$ such that: $d(ir, r_i) = maxDistance$, where ir is the initial resource and r_i is a candidate resources generated by the recommendation algorithm. The pondering function was calculated with an adaptation of the informational content function (Equation 3) defined by Seco et al. [21]. In this equation $hypo(C)$ is the number of descendants of the category C and $|C|$ is the total number of categories in the category graph. This function was selected because it minimizes the complexity of calculation of the informational content, compared to other functions that employ an external corpus [8]. Nonetheless, in ReDyAl, this measure is not limited to a specific property, and optionally can be configured to support a set of forbidden links.

$$hyP(r_1, r_2) = \frac{p(r_1, r_2)}{d(r_1, r_2)} \quad (2)$$

$$p(C) = 1 - \frac{\log(hypo(C)+1)}{log(|C|)} \quad (3)$$

$$Hybrid_{sim} = (1 - LDSD)\alpha + (hyP(r_1, r_2))\beta \quad (4)$$

Finally, the measure that combines LDSD and HyProximity used by ReDyAl is defined in Equation 4, where α and β may be set according to the convenience of the user: α is the weight for the traversal algorithm and β is the weight for the hierarchical algorithm. In this way, resources are ranked in descending order, arranged from the largest to the smallest value of $Hybrid_{sim}$.

4. USER EVALUATION

We comparatively evaluated the prediction accuracy and the novelty of the resources recommended with ReDyAl with respect to three state-of-the-art recommendation algorithms relying exclusively on Linked Data to produce recommendations: dbrec [17], HyProximity traversal and HyProximity hierarchical [4]. This evaluation aimed to answer the following questions: (RQ1) *Which of the considered algorithms is more accurate?* (RQ2) *Which of the considered algorithms provides the highest number of novel recommendations?*

We decided to rely on a user study because we were interested in evaluating the novelty of proposed recommendations over the accuracy. Since we cannot expect that users rated all the items they already know, a user study can measure novelty more precisely than an offline study. On the other side, user studies are more expensive to conduct than an offline studies, for this reason we focus on recommendation algorithms based only on Linked Data and we did not consider algorithm exploiting traditional techniques, or combining Linked Data with traditional techniques. We plan to conduct other experiments to compare our method with other techniques and investigate on the effectiveness of our approach combined with traditional techniques.

Although our algorithm is not bound to any particular dataset, we applied it to DBpedia because it is a general dataset that offers the possibility to evaluate the results in a number of scenarios. DBpedia is one of the biggest datasets in the Web of Data and the most interlinked [20]. Furthermore, it is frequently updated and continuously grows.

4.1 Experiment

A user study was conducted involving 109 participants. The participants were mainly students of Politecnico di Torino (Italy) and University of Cauca (Colombia) enrolled in IT courses. The average age of the participants was 24 years old and they were 91 males, 14 females, and 4 of them did not provide any information about their sex. Although the proposed algorithm is not bound to any particular domain, this evaluation focused on movies because we aimed at applying our algorithm in the mobile application presented in Section 5 (which suggest movies) and in this domain a quite large amount of data is available on DBpedia. Additionally, it was easier to find participants, since no specific skills are required to express an opinion about movies. The algorithms were compared within subjects [22] since each participant evaluated recommendations from different algorithms, as it is explained in the following.

The evaluation was conducted as follows. A list of 20 recommendations generated from a given initial movie was presented to the participants. For each recommendation two questions were asked: *(Q1) Did you already know this recommendation? Possible answers were: yes, yes but I haven't seen it (if it is a movie) and no. (Q2) Is it related to the movie you have chosen? Possible answers were: I strongly agree, I agree, I don't know, I disagree, I strongly disagree. Each answer was assigned respectively a score from 5 to 1.*

We developed a website[9] to collect the answers from the participants. The participants were able to choose an initial movie from a list of 45 movies selected from the IMDB top 250 list[10]. The first 50 movies were considered and 5 movies were excluded because they were not available in DBpedia. Choosing these movies ensured participants to know them, but was also a limitation: the corresponding DBpedia resources are very well-linked, thus we could not properly evaluate the algorithm on poorly linked initial movies. The movies were presented to the user in a random order to avoid having most of the participants evaluating recommendations for the same initial movies (e.g. the first in the lists). When a participant selected an initial movie the tool provided the corresponding list of recommendations with the questions mentioned above. The recommendations were presented in a randomized order. Each participant was able to evaluate recommendations from as many initial movies as he wanted, but he had to answer the questions for all the recommendations, i.e. was not possible to answer only to part of the questions for the initial movie chosen. As a result, the recommendations of the lists for 40 out of 45 initial movies were evaluated by at least one participant and each movie was evaluated by an average of 6.18 participants. The dataset with the initial movies and the lists of recommendations is available online[11].

Each list of 20 recommendations was pre-computed. In particular, recommendations were generated for each of the 45 initial movies with each of the four different algorithms. Then, the recommendations generated by each algorithm were merged in a list of 20 recommendations to be shown to the participants. To do this, we generated a list of 40 recommendations by selecting the first 10 pre-computed recommendations for each algorithm and we ordered them by the similarity computed by each algorithms, since each algorithm ranks its recommendations by using its semantic similarity function with values between 0 and 1. Then we eliminated eventual duplicates, since the same recommendation could be provided by more than one algorithm. The final list was obtained considering the first 20 recommendations of the merged list.

With regard to the questions stated at the beginning of this section, to answer RQ1, the Root Mean Squared Error (RMSE) [22] was computed, and to answer RQ2 the ratio between the number of evaluations was computed in which the recommended item was not known by the participants and the total number of evaluations. For the RMSE measure, scores given by the participants when answering to

[9] `http://natasha.polito.it/RSEvaluation/`

[10] `http://www.imdb.com/chart/top`

[11] `http://natasha.polito.it/RSEvaluation/faces/resultsdownload.xhtml`

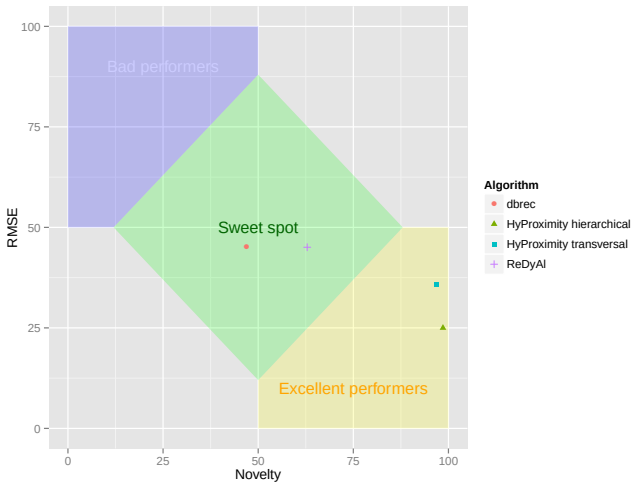

Figure 1: Prediction accuracy and novelty of the algorithms evaluated

Q2 were considered as reference and were normalized in the interval [0, 1], and these scores were compared with the similarities computed by each algorithms, since each algorithm ranks its recommendations by using its semantic similarity function.

4.2 Results

The results of the evaluation are summarized in Figure 1, which compares the algorithms with respect to their RMSE and novelty. The "sweet spot" area represents the conditions in which an algorithm has a good trade-off between novelty and prediction accuracy. In effect, presenting a high number of recommendations not known to the user is not necessarily good because it may prevent him to assess the quality of the recommendations: for example having in the provided recommendation a movie which he has seen and which he liked may increase the trust of the user in the RS.

Regarding RQ1, HyProximity accounts for the lowest RMSE measures (with 25% and about 36% for the hierarchical and traversal versions respectively), but these results are less significant due to the low number of answers to Q2 for these algorithms (this means that the RMSE was computed over a low number of recommendations). For both ReDyAl and dbrec the RMSE is roughly 45%. Concerning RQ2, the two versions of HyProximity account for the highest values (hierarchical roughly 99%, while traversal about 97%). However, such a high rate of novel recommendations may confuse the user and prevent him to judge recommendations, as we have already explained. ReDyAl has a larger rate of novel recommendations than dbrec. These two algorithms account respectively for about 60% and 45%.

The recommendations generated by HyProximity in both traversal and hierarchical version collected a low number of answers to Q2 because most of the recommendations generated by these algorithms were unknown as illustrated in Table 1. Consequently the RMSE was computed over a low number of recommendations. Thus, the results of these two algorithms related to RQ1 are less definitive than for the others, since for measuring the prediction accuracy only the evaluations for which the answer to Q1 was either "yes" or "yes but I haven't seen it (if it is a movie)" were considered.

We computed the Fleiss' kappa [7] measure for assessing the agreement of the participants in answering Q2. We considered the recommendations and in particular we considered as different the same recommendation when related to a different initial movie (i.e. when appearing in different lists of recommendations). We excluded recommendations not evaluated or evaluated by only one participant. The Fleiss' kappa is 0.79; according to Landis and Koch [13], this corresponds to a substantial agreement.

In conclusion, Figure 1 illustrates that ReDyAl and dbrec provides a good trade-off between prediction accuracy and novelty (sweet spot area), although ReDyAl performs better in novelty. HyProximity hierarchical and HyProximity traversal seem to be excellent performers since the RMSE is low and the novelty is high, but the RMSE was computed on few evaluations. An additional analysis of these two algorithms is needed to verify if the user can benefit from such a high novelty and if novel recommendations are relevant. In addition, further investigation is needed on poorly-linked resources, since the choice of the initial movies focused on selecting well known movies to make easier the evaluation from participants, but the related resources were well-linked. On poorly-linked resources we expect ReDyAl and Hyproximity hierarchical keeping good recommendations since they can rely on categories, while dbrec and HyProximity traversal are likely to provide much less recommendations since they rely on direct links between resources.

5. MOBILE MOVIE RECOMMENDATIONS

An implementation of ReDyAl has been integrated into a mobile application developed in collaboration with Telecom Italia (the major network operator in Italy). This application recommends movies based on DBpedia: when the user enters the title of a movie, the application provides the Wikipedia categories to which the initial movie is related to. In this way, the user may focus on a specific scope and can receive recommendations of related resources for any category. In addition, it is possible to view any recommendation to obtain additional information.

Our algorithm can provide cross-domain recommendations because it is independent on the domain and is applied on DBpedia, which is a general dataset. Thus, the recommended resources can be movies but also other relevant entities such as actors, directors, places of recording, books on which the movie is inspired, etc. Other advantages of using DBpedia as dataset are the high number of resources that it represents, the variety of domains addressed and the continuous update and growth, since it is extracted from Wikipedia.

For example, given *The Matrix* as initial movie the categories which it belongs to are presented. The user may be more interested in martial arts, post-apocalyptic movies or he may prefer to consider all the movies from American directors, thus he can choose a category accordingly. By selecting Post-apocalyptic films, a number of resources are recommended. For each recommendation it is possible to open

Algorithm	Yes	Yes but I haven't seen it	No
ReDyAl	27.95	9.17	62.88
dbrec	41.10	11.95	46.95
HyProximity hierarchical	1.08	0.36	98.56
HyProximity traversal	1.32	1.89	96.79

Table 1: Percentage of answers for Q1 by algorithm

a detailed view, which contains three tabs: the first contains a brief textual description, the second presents a graph view of the resource in order to show the main properties and the third summarizes the main information in a tabular form. The graph view is illustrated in Figure 2. The graph is paginated and few properties per page are presented in order to avoid information overload, since the resource can have a very high number of properties. This view can be useful also to explain the recommendation: for instance the user can understand that the recommended resource has the same director or the same main actor as the initial movie. The graph view is based on DBpedia Mobile Explorer [25], a Linked Data visualization framework for the mobile environment, which enables the application to hide the underlying complexity of the Linked Data to the users by processing the resources to be presented received from DBpedia.

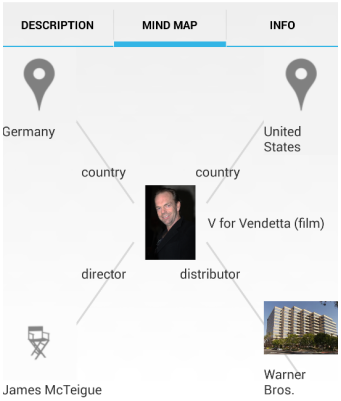

Figure 2: The graph view of *V for Vendetta*

The application is based on a client-server architecture and the main modules are DBpedia, a RESTful recommender service[12] which exposes our algorithm, and the mobile user interface. The main flow of interactions is represented in Figure 3. The mobile application asks for recommendations specifying an initial resource and optionally a scope such as a Wikipedia category (1). The recommender service answers with a list of scopes if no scope was provided or with a list of recommendations in the scope specified, otherwise (2). The recommender service relies on DBpedia to provide recommendations (3, 4) and the mobile application retrieves the resources to be visualized from the dataset (5, 6). The recommender service is developed in Java, while the client is an Android mobile application. The two modules use JSON as data-interchange format, while the mobile application retrieves resources from DBpedia serialized in JSON-LD[13]. The mobile application is going to be published on Google Play, but the Android Package (APK) of the first version is already available on the Web[14].

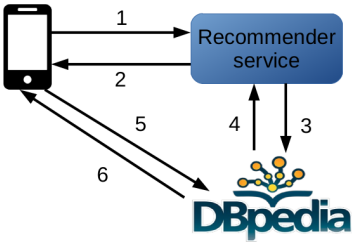

Figure 3: The interactions between the main modules of the application

6. CONCLUSIONS AND FUTURE WORK

We presented ReDyAl which is a hybrid algorithm that dynamically uses both the traversal and hierarchical approach for discovering resources. It is independent from the application domain and, although we applied it to DBpedia, it could be easily adapted to other dataset in the Web of Data. It relies only on Linked Data and does not require to reduce the set of resources and links of the dataset to those belonging to a specific domain.

We evaluated and compared our algorithm against three state-of-the-art algorithms by conducting a user study and we also showed a practical application of the algorithm by presenting a mobile application that provides movie recommendations relying on DBpedia. Although the algorithm could be applied to other datasets in the Web of Data, we selected DBpedia because it is a general dataset, thus cross-domain recommendations were possible. In addition, there is a high number of resources represented, a variety of domains addressed and it is continuously updated, since it is extracted from Wikipedia. The user study demonstrated that ReDyAl improves in the novelty of the results discovered, although the accuracy of the algorithm is not the highest (due to its inherent complexity). Although ReDyAl is not bound to any particular domain, the study focused on movies as for this domain there is a quite large amount of data available on DBpedia and participants were not required to have specific skills.

Future work includes studying the relevance under different domains and improving the accuracy of ReDyAl while maintaining its novelty. We plan to conduct other studies to compare it with traditional techniques and with approaches which combine Linked Data with traditional techniques. We are also working on combining ReDyAl with collaborative filtering techniques in order to take user preferences into account while providing recommendations.

[12] http://natasha.polito.it/LDRecommenderWeb/
[13] http://json-ld.org/
[14] https://www.dropbox.com/sh/0q8d2mcbko9e2oj/AAASh-YHGz0MmG_Z8hH6mfWOa?dl=0

7. REFERENCES

[1] S. Baumann, R. Schirru, and B. Streit. Towards a storytelling approach for novel artist recommendations. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 6817 LNCS, pages 1–15, 2011.

[2] I. Cantador, I. Konstas, and J. M. Jose. Categorising social tags to improve folksonomy-based recommendations. *Web Semantics: Science, Services and Agents on the World Wide Web*, 9(1):1–15, Mar. 2011.

[3] G. Cheng, S. Gong, and Y. Qu. An Empirical Study of Vocabulary Relatedness and Its Application to Recommender Systems. In *10th International Conference on The Semantic Web - Volume Part I*, pages 98–113. Springer, 2011.

[4] D. Damljanovic, M. Stankovic, and P. Laublet. Linked data-based concept recommendation: Comparison of different methods in open innovation scenario. 7295:24–38, 2012.

[5] V. de Graaff, A. van de Venis, M. van Keulen, and R. A. de By. Generic knowledge-based analysis of social media for recommendations. In *CBRecSys 2015: New trends on content-based recommender systems*. CEUR-WS.org, Sept 2015.

[6] C. Figueroa, I. Vagliano, O. Rodríguez Rocha, and M. Morisio. A systematic literature review of linked data-based recommender systems. *Concurrency and Computation: Practice and Experience*, 2015.

[7] J. Fleiss et al. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382, 1971.

[8] M. Hadj Taieb, M. Ben Aouicha, M. Tmar, and A. Hamadou. New information content metric and nominalization relation for a new wordnet-based method to measure the semantic relatedness. In *Cybernetic Intelligent Systems (CIS), 2011 IEEE 10th International Conference on*, pages 51–58, Sept 2011.

[9] H. Hamdan. Experiments with DBpedia , WordNet and SentiWordNet as re- sources for sentiment analysis in micro-blogging. In *Seventh International Workshop on Semantic Evaluation (SemEval 2013) - Second Joint Conference on Lexical and Computational Semantics*, volume 2, pages 455–459, Atlanta, Georgia, 2013.

[10] Y. Kabutoya, R. Sumi, T. Iwata, and T. T. Uchiyama. A topic model for recommending movies via linked open data. In *2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, pages 625–630. IEEE, Dec. 2012.

[11] H. Khrouf and R. Troncy. Hybrid event recommendation using linked data and user diversity. In *Proceedings of the 7th ACM conference on Recommender systems - RecSys '13*, RecSys '13, pages 185–192. ACM Press, 2013.

[12] K. Kitaya, H.-H. Huang, and K. Kawagoe. Music Curator Recommendations Using Linked Data. In *Second International Conference on the Innovative Computing Technology (INTECH 2012)*, pages 337–339. IEEE, Sept. 2012.

[13] J. R. Landis and G. G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174, 1977.

[14] E. Mannens, S. Coppens, T. De Pessemier, H. Dacquin, D. Van Deursen, R. De Sutter, and R. Van de Walle. Automatic news recommendations via aggregated profiling. *Multimedia Tools and Applications*, 63(2):407–425, 2013.

[15] C. Musto, P. Basile, M. de Gemmis, P. Lops, G. Semeraro, and S. Rutigliano. Automatic selection of linked open data features in graph-based recommender systems. In *CBRecSys 2015: New trends on content-based recommender systems*, pages 10–13. CEUR-WS.org, Sept 2015.

[16] V. C. Ostuni, T. Di Noia, E. Di Sciascio, and R. Mirizzi. Top-N recommendations from implicit feedback leveraging linked open data. In *Proceedings of the 7th ACM conference on Recommender systems - RecSys '13*, RecSys '13, pages 85–92. ACM Press, 2013.

[17] A. Passant. dbrec - Music Recommendations Using DBpedia. In *The Semantic Web - ISWC 2010*, pages 209–224. Springer Berlin Heidelberg, 2010.

[18] L. Peska and P. Vojtas. Enhancing Recommender System with Linked Open Data. In *10th International Conference on Flexible Query Answering Systems (FQAS 2013)*, pages 483–494, Granada, Spain, 2013. Springer Berlin / Heidelberg.

[19] F. Ricci, L. Rokach, and B. Shapira. Introduction to recommender systems handbook. In F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, editors, *Recommender Systems Handbook*, pages 1–35. Springer US, 2011.

[20] M. Schmachtenberg, C. Bizer, and H. Paulheim. Adoption of the linked data best practices in different topical domains. In *The Semantic Web - ISWC 2014*, volume 8796 of *Lecture Notes in Computer Science*, pages 245–260. Springer International Publishing, 2014.

[21] N. Seco, T. Veale, and J. Hayes. An Intrinsic Information Content Metric for Semantic Similarity in WordNet. In *European Conference on Artificial Intelligence*, pages 1089–1090, 2004.

[22] G. Shani and A. Gunawardana. Evaluating recommendation systems. *Recommender Systems Handbook*, pages 257–297, 2011.

[23] M. Stankovic, W. Breitfuss, and P. Laublet. Discovering Relevant Topics Using DBPedia: Providing Non-obvious Recommendations. In *2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, pages 219–222. IEEE, Aug. 2011.

[24] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: A core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 697–706, New York, NY, USA, 2007. ACM.

[25] I. Vagliano, M. Marengo, and M. Morisio. DBpedia Mobile Explorer. In *1st International Forum on Research and Technologies for Society and Industry Leveraging a better tomorrow (RTSI)*, pages 181–185, Sept 2015.