Exploiting Linked Data-based Personalization Strategies for Recommender Systems

Gabriela Oliveira Mota Da Silva[1,2][a], Lara Sant'Anna do Nascimento[1][b] and Frederico Araújo Durão[1][c]

[1]*Institute of Computer Science, Federal University of Bahia, Avenida Milton Santos, s/n, Salvador, Brazil*

[2]*Federal Institute of Education, Science and Technology of Bahia, Avenida Centenário, 500, Jacobina, Brazil*

Keywords: Recommender Systems, Linked Data, Personalization, Semantic Similarity, Feature Selection.

Abstract: People seek assertive and reliable recommendations to support their daily decision-making tasks. To this end, recommendation systems rely on personalized user models to suggest items to a user. Linked Data-driven systems are a kind of Web Intelligent systems, which leverage the semantics of links between resources in the RDF graph to provide metadata (properties) for the user modeling process. One problem with this approach is the sparsity of the user-item matrix, caused by the many different properties of an item. However, feature selection techniques have been applied to minimize the problem. In this paper, we perform a feature selection preprocessing step based on the ontology summary data. Additionally, we combine a personalization strategy that associates weights with relevant properties according to the user's previous interactions with the system. These strategies together aim to improve the performance and accuracy of the recommender system, since only latent representations are processed by the recommendation engine. We perform several experiments on two publicly available datasets in the film and music domains. We compare the advantages and disadvantages of the proposed strategies with non-personalized and non-preprocessed approaches. The experiments show significant increases in top-n recommendation tasks in Precision@K (K=5, 10), Map, and NDCG.

1 INTRODUCTION

Asking for recommendations is part of everyday life for many people in different areas, whether it is shopping, reading a book, listening to a song, or even looking for a romantic couple. Thus, one searches for a reliable source of recommendation so that the object sought is following their tastes (Ricci et al., 2015). To this end, recommender systems (RS) increasingly rely on personalized user models by exploiting metadata from the items that a user has previously interacted with to suggest similar items they might be interested in. The user model plays a significant role in many personalized applications. For example, an RS analyzes the 1 to 5-star ratings given by a user to movies and music in the past to establish their general interests in art. The system then evaluates and suggests other items that it expects the user will genuinely like in the future.

Linked Data (LD) or Linked Open Data (LOD) is a powerful source of diverse data structured under open Semantic Web standards, such as Resource Description Framework (RDF) and SPARQL (Berners-Lee, 2009). DBpedia[1] is a notable example of a LOD set built by a strong community effort to extract structured information from Wikipedia and make this information openly available on the Web.

In the past years, LD datasets such as DBpedia have been proposed as a valuable source of information to increase the predictive power of RSs (Di Noia et al., 2018). In such Linked Data-driven RS, links between resources in the RDF graph play the primary role of providing metadata (properties or features) for the user modeling process and the recommendation model (Passant, 2010; Di Noia et al., 2012; Piao and Breslin, 2016). These systems use a semantic similarity algorithm that calculates the degree of matching between pairs of Linked Data resources. Because RDF represents data as a graph, these algorithms, in general, count the number of direct and indirect links,

[a] https://orcid.org/0000-0001-8799-5642
[b] https://orcid.org/0000-0001-7428-9992
[c] https://orcid.org/0000-0002-7766-6666

[1] https://www.dbpedia.org/

Da Silva, G., do Nascimento, L. and Durão, F.
Exploiting Linked Data-based Personalization Strategies for Recommender Systems.
DOI: 10.5220/0011591300003318
In *Proceedings of the 18th International Conference on Web Information Systems and Technologies (WEBIST 2022)*, pages 226-237
ISBN: 978-989-758-613-2; ISSN: 2184-3252

the length of the path between two resources, or their place in the hierarchy of classes (Passant, 2010; Piao et al., 2016; Cheniki et al., 2016).

One problem of this approach is the sparsity of the user-item matrix, provided by the numerous different properties one item could have. However, various feature selection techniques have been applied to discard irrelevant properties and minimize the problem. These techniques filter properties that can cause the system's reliability to decrease and only consider the most relevant subset of the original data in the recommendation process step.

Nevertheless, some effort has been made to select subsets of features that seem more helpful in computing similarity between items of a graph dataset (Catherine and Cohen, 2016; Musto et al., 2016), reducing the matrix dimension. Recent studies cite several methods to find and exclude or rank features based on diverse goals — see Section 4 for examples.

This work investigates LD-based personalization strategies to provide more representativeness to the user model. The proposed algorithm associates weights to the properties of items, according to their relevance to the user. The relevance is calculated by identifying among the items that the user has previously liked (user model), which properties link to the same item repeatedly. We assume that properties that appear more often in a particular user's model are more important for the calculation of future recommendations. We also perform a filtering preprocessing step in which we apply an ontology-based summarization method (Di Noia et al., 2018) to filter properties according to the domains. In summary, these strategies together aim to reduce the dimensionality of the spaces and enhance the recommender system performance.

We perform several rounds of experiments with two different domain datasets - MovieLens and LastFM, enriched with DBpedia semantic information, to evaluate the system's effectiveness and tune its parameters. The similarities are calculated with the Personalized Linked Data Semantic Distance (PLDSD) method (da Silva et al., 2019). We present the advantages and disadvantages of the proposed strategies by comparison with non-personalized and non-preprocessed approaches and discuss the results in terms of performance and accuracy.

The remainder of the paper is organized as follows: Section 2 explains about Linked Data driven RS; Section 3 discusses feature selection ad other personalization methods; Section 4 lists the work related to this research; Section 5 introduces and describes the proposed personalization approaches; Section 6 sets forth the experimental evaluation in the context of two LD-based recommender systems simulation; Section 7 includes the discussion on the results and some points of improvement; and Section 8 concludes the paper and provides suggestions for future work.

2 LD-BASED RECOMMENDER SYSTEMS

Recommender Systems (RSs) are software tools and techniques that aim to solve the information overload problem by suggesting items that are most likely of interest to a particular user (Ricci et al., 2015). Literature essentially discusses two recommender methods: Content-Based (CB), in which the similarity of items is calculated by analyzing features associated with the compared items; and Collaborative Filtering (CF), in which recommendations to one user are based on items that other users with similar tastes liked in the past. Regardless of the chosen technique, the user model plays a central role in recommender systems since it encodes the user's preferences and needs (Ricci et al., 2015). For instance, in considering a CB approach, the user can be profiled directly by its ratings of items. Thus, in a LD enriched Recommender System context, resources and their relations are mixed to the user model (ratings over items) to calculate similarity and generate personalized user recommendations (da Silva et al., 2019).

The data that describe items in an RS are derived from many sources, most often private databases belonging to companies. However, the availability of open sources of knowledge is growing, increasing the emergence of semantic-based applications, for example, Semantic-Aware SRs (de Gemmis et al., 2015). Linked Open Data (LOD) is a widely known project that aims to connect open datasets over the Web, and facilitate their use by applications (Bizer et al., 2009). The LOD project consists of 1,255 datasets with 16,174 links, as of May 2020[2], that covers an extensive collection of statements related to resources — such as people, places, songs, movies and so on. A typical LOD dataset is DBpedia, which incorporates links to other datasets, such as DailyMed, Geonames, and LinkedGeoData. As these additional links are provided, applications can exploit knowledge from interconnected datasets for developing semantic applications.

Semantics-aware applications rely on an RDF graph architecture. The RDF statement (also known as *triple*) has the form of *subject-predicate-object*, each part uniquely identifiable by Uni-

[2]https://lod-cloud.net/

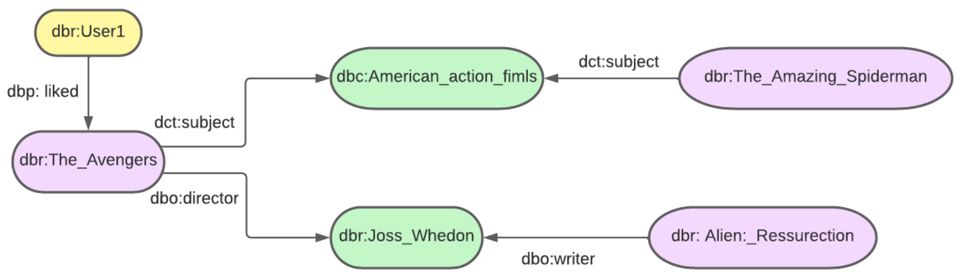

Figure 1: A DBpedia movie context example.

form Resource Identifiers (URIs). For example, in the DBpedia movie segment shown in Figure 1, dbr:The_Avengers is the subject of one triple, dbo:director is the predicate, and dbr:Joss_Whedon is the object, forming the sentence: The Avengers was directed by Joss Whedon. Another triple links the subject dbr:Alien_Ressurection to the same object dbr:Joss_Whedon by a different property — dbo:writer — forming, in turn, the sentence: Alien Ressurection was written by Joss Whedon.

This work focuses on CB Recommender Systems, which compute the similarity between items in the system by comparing their characteristics (also called features). For instance, in a movie Recommender System, algorithms reason about the degree of similarity between two movies by comparing their directors, actors, the main subject, and so on. Although several studies address different ways to build recommendations, RS researchers still discuss some problems. One is the sparsity problem, when there is much less feedback data in comparison to the whole data matrix size; or the well-known cold-start problem, that happens when a new item or a new user is added to the system. Another common issue is how to offer more personalized recommendations to users. LD are often used to add semantics to the system in order to help address those well-known RS problems (Di Noia et al., 2012; de Gemmis et al., 2015; Joseph and Jiang, 2019).

In LD-based RSs, items' features are expressed by links between nodes in the RDF graph, that represent the resources' *properties*. Thus, similarity algorithms need to consider the semantics brought by those links. Different similarity measures were proposed to handle this task through LD-based systems (Passant, 2010; Di Noia et al., 2012; Piao and Breslin, 2016). Most of them can manage the sparsity problem well. Nevertheless, one typical characteristic of the previous work on LD similarity measures is that most research considers all the links on an RDF graph as having equal importance. Retaking the movie RS context, genre is considered an important feature to the movie domain. One user although likes to watch only to recent released titles. Given this hypothetical scenario, is it correct that algorithms consider the property genre as having the exact weight of the release year of a movie? Based on this issue, we will discuss some ways to rank features regarding a given RS context and/or select the best combination of properties that fits a particular user model.

3 FEATURE SELECTION AND PERSONALIZATION

Many machine learning tasks, such as recommender systems, demand a preprocessing step to filter a smaller and more significant portion of the dataset. When working with Linked Data, a single item in the original dataset may have hundreds or thousands of variables (as many as the number of links), causing the feature matrix to be oversized. Studies covering this issue focus mainly on two branches of research: selecting subsets of features that are useful to build a good predictor — Feature Selection (FS), and ranking all potentially relevant variables in a context — Feature Ranking (FR) (Guyon and Elisseeff, 2003).

One common way of performing FS tasks is through hand-crafted work: manually selecting the most relevant features according to simple heuristics or domain knowledge. Nevertheless, there are attempts to automatize the process, for example, Musto et al. (Musto et al., 2016) assess the impact of some classic feature selection techniques on recommendations accuracy, such as Principal Component Analysis (PCA), Information Gain Ratio (GR), PageRank (PR) and Support Vector Machines (SVM).

In an LD-Based Recommender System, properties are considered as features of a given node in the knowledge graph. In a different approach, Di Noia et al. (Di Noia et al., 2018) show how LD-based summarization can drive FS and FR tasks by comparing an automated feature selection method based on ontology data summaries with other classical ones, like manual selection.

Although FS and FR preprocessing techniques are commonly applied in recommended models, most of them consider links as having equal importance from the user's point of view. In this work, we further perform an FR task personalized. We not only use ontology summarization to select the most relevant features from the LD graph, but we rank them by adding weights to each link according to the user model.

This step is called graph personalization and is responsible for automatically ranking the properties that most influenced the ratings given by a user of a recommender system. For example, in a movie RS, the genre, artists, and director properties are usually considered important by automated FS algorithms such as ontology summarization. However, if a particular user has given positive ratings to movies of various genres and directors, but most are short movies, our personalization method will classify the movie length as more important than the other properties, giving it a higher weight.

4 RELATED WORK

This work addresses the problem of building a recommender system over a LOD dataset, where resources' properties form sparse matrices. Literature is plenty of work that draws on a feature-based definition of Linked Data, such as Meymandpour & Davis (Meymandpour and Davis, 2016) which proposes a generalized information content-based approach with systematic assessment of semantic similarity between entities. They extensively reviewed existing semantic similarity measures and proposed a hybrid method made of feature-based and statistical techniques.

One example of a semantics-aware application is when a Knowledge Graph (KG) is used to provide better user recommendations (Catherine and Cohen, 2016). In most works following this line of research, the authors enrich benchmark databases with semantics given from the interconnections between KG nodes. Musto et al. (Musto et al., 2016) studies the impact of using knowledge coming from Linked Data on the overall performance of a recommendation algorithm. They reduced the matrices dimension by testing several feature selection techniques against two datasets. They investigate whether the integration of LD-based features improves the effectiveness of the algorithm and to what extent the choice of different feature selection techniques influences its performance in terms of accuracy and diversity.

Furthermore, Di Noia et al. (Di Noia et al., 2018) show how LD-based summarization can drive feature selection tasks by proposing a fully automated method based on semantics provided by the ontology. They evaluate the accuracy of each strategy used and compare them to classic methods such as manual selection and statistical distribution methods. Finally, they aggregate diversity to the recommender system by exploiting the top-k selected features. In the inverse direction, Van Rossum & Frasincar (van Rossum and Frasincar, 2019) investigate the incorporation of graph-based features into path-based similarities. They proposed two normalization procedures that adjust user-item path counts by the degree of centrality of the nodes connecting them. It's based on the idea that a user liking one movie tells us more about the popularity of this movie than about the particular user. Our approach has similar premises, however, the methodology diverges as we exploit the link information from the perspective of each specific user.

Some studies were made in this line of user-based personalization. However, approaches do not embrace graph-based systems. On Singh et al. (Singh et al., 2018), user-based collaborative filtering is used to generate recommendations by utilizing both items and user preferences based on splitting criteria for movie recommendation applications. Every single item and every single user are split into two virtual items and two virtual users based on contextual values. The recommender technique is then applied to the new dataset of split items and users. A new approach is proposed by Yi et al. (Yi et al., 2018) by combining the tasks of rating prediction from a rating-based system with a review-based RS. In this manner, they made a user-item rating relation from latent feature representations and fuse it to the user-item review relation extracted from users' reviews.

Latest research addresses new approaches to improve the user experience in RS, as in Blanco. et al. (Blanco. et al., 2021). They propose a recommender recovery solution with an adaptive filter to deal with the failed recommendations, which after a recommendation failure, filters out all items that are similar to the one disliked by the user. The objective is to keep the user engagement and allow the recommender system to become a long-term application. Same as Yi et al., this approach is not adapted to LD-based systems. Another interesting approach is the use of long-tail algorithms to generate recommendations that can lead users to explore less popular but highly relevant items in the RDF graph (de Sousa Silva et al., 2020). Our approach aims to improve the user experience as well, with the advantage of being LD-driven. Gan et al. (Gan et al., 2021) proposed an EM-model that alternates between a general item diversity learning and knowledge graph embedding learning for user and item representation, which helps to achieve better results in comparison to the state-of-art baselines on datasets MovieLens and Anime. Although this work takes advantage of auxiliary information along with historical interactions between user and item from knowledge graphs, it does not cover a personalization method.

Cao et al. (Cao et al., 2019) proposed a research based on the idea that a KG has commonly missing facts, relations, and entities. Thus, they argue that it is crucial to consider the incomplete nature of KG when incorporating it into recommender system. The main difference between the related work cited above and our approach fundamentally is that neither of them really leverages the knowledge present in the user model (KG) to make personalized recommendations. Among the related works, the one that approaches most closely the idea of this work is Cao et al., because they explore the KG trying to understand the reasons why a user likes an item. They provide an example that if a user has watched several movies directed by (relation) the same person (entity), it is possible to infer that the director relation plays a critical role when the user makes the decision, thus help to understand the user's preference at a finer granularity (Cao et al., 2019). We also started our research from this idea and developed a weighting algorithm that ranks properties by importance from the user's point of view. Before the ranking step, we also address the concept of KG completion, although we use the rationale of direct and indirect links coming from Passant (Passant, 2010). We generate the missing direct links by counting the indirect links between the items the user interacted with, as we demonstrate below.

5 PROPOSAL

This section is dedicated to explaining the strategies proposed in this work, beginning with the user graph personalization, and going through how the weights are applied together with a similarity measure for recommending items to the user.

5.1 Notation

A user model or user profile is how a recommender system represents the users' preferences about items, as discussed in Section 2. In this work, we model the user profile as part of the Linked Data graph that describes the system's domain. Therefore, we assume a set of RDF resources identified by their URI $R = \{r_1, r_2, ..., r_n\}$, and a set of users identified by their URI $U = \{u_1, u_2, ..., u_m\}$, in which each instance of u_k is also itself a resource in the LD graph, in other words, $U \subset R$. Figure 2 shows an example of a user resource u_1 = dbr:User1, which is represented in yellow, to distinguish it from other resources.

We also assume a set of properties $P = \{p_1, p_2, ..., p_h\}$ that link together pairs of resources from R molding triples in the form of $T = \{t_1, t_2, ..., t_q\}$. Thus, as an RDF triple is a statement in the format ⟨subject, predicate, object⟩, then $t_i = \langle r_a, p_j, r_b \rangle \in T$ means that there is an instance of a property $p_j \in P$ linking the subject $r_a \in R$ to the object $r_b \in R$, like in ⟨dbr:Toy_Story, dbo:director, dbr:John_Lasseter⟩, back to Figure 2.

Following the DBpedia notation, we elaborated one property $p_x = dbr:liked$ that represents all the positive feedback from users over the items, with $p_x \in P$. For example, a triple $t_1 = \langle u_1, dbr:liked, r_1 \rangle$ means that the user $u_1 \in U$ has rated the item $r_1 \in R$ as a positive feedback.

In essence, a LOD graph is a directed graph $G = (R, P, T)$, that encompass the sets of resources $R = \{r_1, r_2, ..., r_n\}$, properties $P = \{p_1, p_2, ..., p_h\}$ and triples $T = \{t_1, t_2, ..., t_q\}$, in combination with a set of special resources called users $U = \{u_1, u_2, ..., u_m\}$, and one special property $p_x = dbr:liked$, which together provide the notation for modeling the user's preferences over the dataset items.

5.2 Graph Personalization

Our proposed strategy is composed of a user model personalization approach (da Silva et al., 2019), and of an ontology-based feature selection approach adapted for LD contexts from Di Noia et al. (Di Noia et al., 2018). In the personalization step, any resource r_a from R that is linked to a user u_k by the particular property p_x is considered as part of the user model. These resources are always of the same type, which is the main class of the domain dataset. For example, Figure 2 models a portion of a movie dataset, where we can notice the resources from the movie class highlighted in lavender color. The three of them are in the model because the user u_1 = dbr:User1 has liked them through the property $p_x = dbr:liked$.

The personalization method intends to assign weights to every property p_j from P in the graph G that links a user model item — colored in lavender in Figure 2 and represented in equations by r_a — to any other resource — mostly the ones colored in green, represented by r_b. The goal is to rank properties by applying weights so that the aspects of the film that influenced the user the most are at the top of the list. It means that we assign higher weights to the properties p_j that occur more among the liked movies. Equation

1 shows how the weights calculation is made for each user u_k.

$$W(p_j,u_k) = \frac{1}{1+\frac{\sum_{t_i} Freq(r_b)}{q}} \quad (1)$$

Where $t_i = \langle r_a, p_j, r_b \rangle \in T$, and $Freq(r_b)$ calculates the frequency of the resource r_b, by iterating over the triples $t_i = \langle r_a, p_j, r_b \rangle \in T$ with i varying from 1 to q.

Since the resource r_a always refers to an item that was positively evaluated by a user u_k, the W equation is performed for every p_j until a rank of properties is generated from the perspective of a particular user u_k. In future recommendations, the ranking will be taken into account when calculating the similarity of new items. In resume, the graph personalization process is based on the concept that the more one property is associated with appreciated items, the greater the importance of this property in future recommendations for the user.

5.3 Personalized LDSD

In this work, we show that ranking the most relevant features from the user's point of view, instead of considering just the context model, increases the effectiveness of the RS. The classic LDSD (Passant, 2010) measure considers every link p_j as having the same weight on calculations. In order to make the recommender system personalized, we add the function $W(p_j u_k)$ to the PLDSD Equation. As W represents the weight to one property for a given user's point of view, its value is multiplied with each function $C(p_j, r_a, r_b)$, which are iterated to the j number of properties linking r_a and r_b, as shown in Equation 2.

The PLDSD distance counts direct and indirect links from resource r_a to resource r_b and vice versa. All functions $C(p_j, r_a, r_b)$ compute whether there is a link p_j between resources r_a and r_b. They return 1 in case there is, 0 otherwise. Whenever there is an n, the function is calculating the total number of links between a resource r_a or r_b to all other resources. C_d only takes into consideration direct links from r_a to r_b; reverse links, from r_b to r_a, count as different links.

The function C_{ii} represents indirect incoming links and returns 1 only if there is a resource r_c that satisfies both $\langle p_j, r_a, r_c \rangle$ and $\langle p_j, r_b, r_c \rangle$, 0 if not. In the case of returning 1, a virtual direct link is created between the two resources that share the incoming link. The function C_{io} represents indirect outgoing links and equals to 1 only if there is a resource r_c that satisfies both $\langle p_j, r_c, r_a \rangle$ and $\langle p_j, r_c, r_b \rangle$, 0 if not. In the case of returning 1, a virtual direct link is created between the two resources that share the outgoing link.

In Figure 2 the highlighted property dct:subject was created because The Amazing Spider Man and The Avengers share the same outgoing link. Similarly, the property dbo:product was created because The Avengers and Toy Story share the same ingoing link. It is possible to infer from the figure that these two properties will have a greater weight after the W equation is calculated, because they appear more frequently in the graph, influencing the PLDSD result.

5.4 Semantic Feature Selection

We adapted the ABSTAT Feature Selection method (Di Noia et al., 2018) by applying the LDSD measure to it and implemented a preprocessing step that filters out the most important features for each domain. That method originally uses the Jaccard index, a well-known but non-specific similarity measure for LD-based systems.

ABSTAT is the name of the original implementation (Di Noia et al., 2018), which consists of an automatic FS method, the ontology-based data summarization framework. This framework identifies patterns and frequencies and computes cardinality descriptors from the RDF triples of one domain. The authors conduct experiments in three different domains: movies, music and books.

6 EXPERIMENTAL EVALUATION

The goal of the experimental evaluation is to validate the hypothesis that personalized user models can lead to more effective recommendations in Linked Data-based systems. To prove this, we implement a personalization strategy that weighs and ranks each property in the user model according to the user's past interactions with the system, called PLDSD.

We performed two comparative experiments against the classical LDSD method, one using a movie RS and one using a music RS. Results show greater accuracy of the system when using PLDSD instead of the non personalized LDSD. Additionally, we run the PLDSD method with and without the preprocessing step — the semantic Feature Selection method — as additional validation experiments. We demonstrate that using a filter technique appropriate to the application domain — both by being LD-based

$$sim_{PLDSD}(r_a,r_b,u_k) = \frac{1}{1+\sum_j \frac{C_d(p_j,r_a,r_b).W(p_j u_k)}{1+\log C_d(p_j,r_a,n)} + \sum_j \frac{C_d(p_j,r_b,r_a).W(p_j u_k)}{1+\log C_d(p_j,r_b,n)} + \sum_j \frac{C_{ii}(p_j,r_a,r_b).W(p_j u_k)}{1+\log C_{ii}(p_j,r_a,n)} + \sum_j \frac{C_{io}(p_j,r_a,r_b).W(p_j u_k)}{1+\log C_{io}(p_j,r_a,n)}} \quad (2)$$

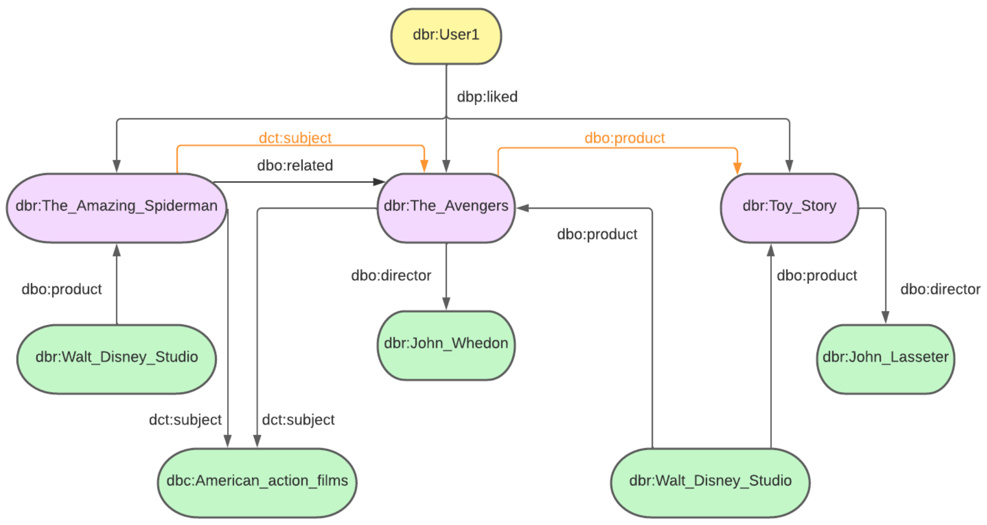

Figure 2: Example of weight calculation in PLDSD.

and by taking into account the semantics of the domain elements — also increases the accuracy of the recommendations when combined with the personalization strategy. We critically discuss all these results in Section 7.

The experiments were implemented in Java using JENA, an open source framework for building Semantic Web and Linked Data applications[3]. We also store similarity values and other data in a MySQL database and leverage the linked data environment with the Virtuoso Server [4]. The complete implementation project has been made available on an Anonymous Github under the url https://anonymous.4open.science/r/lodweb-pldsd-40BD/. Researchers interested in replicating and conducting further comparative studies can find the necessary documentation at this location.

6.1 Dataset Setup

Two datasets were configured to run the experimental evaluations. The first dataset is the Movielens 1M dataset[5], which has: 1 million ratings from 6000 users on 4000 movies (1-5 stars rating). The Dataset rating density is 4.26%. From this start point, we modeled a database that stores the user model — users, movies, and user ratings over movies, and also stores the content of movies in the form of RDF triples. For this purpose, we used the MappingMovielens2DBpedia project[6], which provides RDF identifiers — URIs — for each movie in MovieLens 1M. Therefore, this initial setup allows us to access resources and all their connections from DBpedia through SPARQL queries online.

The second dataset is the Last.FM Million Song[7]. It contains listening information from almost 2 thousand users about approximately 1 million songs. Nonetheless, we take the listening data summarized to user-artist for this work, rather than considering the songs played. For this reason, during the dataset setup step, we used another set of data released during the HETRec 2011 Workshop and published as another project of raw data mapping to Linked Data[8]. As is in the movie mapping project, this one provides DBpedia URIs for each musical artist or band in the Last.FM dataset.

6.2 Methodology

In this work, we simulate two recommender systems that retrieve the Top-N recommendations to the user: a Movie RS and a Music RS. User preferences about movies in the MovieLens dataset are expressed on a 5-star Likert scale. Some work on the MovieLens dataset use methods for making the user ratings binary (Di Noia et al., 2012; Musto et al., 2016). The PLDSD methodology considers 3 different scenarios to represent a user's positive feedback: $PLDSD_{rat=5}$, that only ratings equal to 5 are considered; $PLDSD_{rat>=4}$ that considers movies rated as 5 and 4; and $PLDSD_{rat>=3}$ that takes ratings equal to 5,

[3]https://jena.apache.org/

[4]https://virtuoso.openlinksw.com/

[5]https://grouplens.org/datasets/movielens/1m/

[6]https://github.com/sisinflab/LODrecsys-datasets/blob/master/Movielens1M/MappingMovielens2DBpedia-1.2.tsv

[7]http://millionsongdataset.com/lastfm/

[8]https://github.com/sisinflab/LinkedDatasets/blob/master/last_fm/mappingLinkedData.tsv

4 and 3 as positive feedback. Previous work (da Silva et al., 2019) demonstrated that the $PLDSD_{rat=5}$ strategy performed the most accurate results for all scenarios. For this reason, we adapted our methodology to consider ratings 5 as positive feedback and all the other ratings — 1, 2, 3, and 4 — as negative feedback.

Similarly, in the Last.FM dataset, user preferences about artists are expressed through the number of times that each user has listened to their songs. Consequently, the concept of positive feedback for this work had to be constructed by isolating an expressive number that represents a user listening to a well-liked artist throughout their daily routine. We chose the number 500 to represent positive feedback and therefore artists tracks heard less than 500 times are considered negative feedback given by a user. Based on this, we have built the user models from the set of items positively evaluated by each user. We also personalized the user models by applying the PLDSD methodology as explained in section 5.

The personalization step results in lists of ranked properties according to the user's past choices in the system. These lists may contain all the properties involved in the user model, or they may be limited by a cutoff point, which is the value of the parameter k in the ontology summarization approach (section 3). In other words, the parameter k in the feature selection preprocessing step defines how many properties will be ranked by the personalization step.

From this point on, we will refer to movies and musical artists simply as items, since the methodology applies to any domain and we would like to describe it as generally as possible. First, we take items that the user has given positive feedback and build the training dataset or the dataset of known data. Then we take a subset of 90 items randomly (discarding those that are already in the user model subset) and build the test dataset. After that, we perform the PLDSD algorithm on the test dataset and also perform the LDSD algorithm as the baseline method. The user model is then used twice during each round of experiments: first to perform the weigh preprocessing step described in Subsection 5; and secondly as the validation dataset to which we compare the ranked result list coming from the recommendation method.

Moreover, we performed some statistical tests to assess the significance of the experimental results. First, we ran the chi-squared goodness-of-fit test over the ranked lists of movies and music. As this test revealed the normal distribution of the data, we opted to apply a paired T-test using a p-value $<< 0.0001$ to each PLDSD sample — with 3 variations of k values — against its correspondent baseline method (LDSD) results' sample — with 3 more variations of k values. As the length of the sample varies according to how many items the user has rated, we ran statistical tests for each user tested.

6.3 Metrics

We compute the results under 3 evaluation metrics to assess the quality of the generated ranks. These metrics assess the relevance and accuracy of the similarity algorithm. In other words, they measure how accurately a system can predict similar items to a specific user.

- **Precision@K** - Precision measures the fraction of retrieved items from a dataset that are relevant, as shown in Equation 3. Precision@K is a variation that represents the amount of relevant results at the position K (Equation 4). We have adopted in our evaluation two classic values of K: 5 and 10 (Davoodi et al., 2013).

$$Precision = \frac{|RelevantItems \cap RetrievedItems|}{|RetrievedItems|} \tag{3}$$

$$Precision@K = \frac{|RelevantItems@K|}{K} \tag{4}$$

- **Mean Average Precision (MAP)** - It is a combination of averages that has good discrimination and stability (Manning et al., 2008). MAP is an extension of the Average Precision (AP) which takes the average of all AP as shown in Equation 5.

$$MAP(Q) = \frac{1}{|Q|} + \sum_{j=1}^{|Q|} \frac{1}{m_j} \sum_{k=1}^{m_j} Precision(R_{jk}) \tag{5}$$

  Given the set $\{d_1, ..., d_{m_j}\}$ of relevant documents for an information need $q_j \in Q$ and R_jk is the set of ranked results from the top result until document d_k.

- **Normalized Discounted Cumulative Gain (NDCG)** - It is a metric based on the notion that items in a rank have varying degrees of relevance, either once relevant items appear in a low-rank position or when less relevant items appear in a greater position. (Järvelin and Kekäläinen, 2002). Thereby, the gained value (without discounts) is obtained as the relevance score of each item is progressively summed from the rank position 1 to n. Discounted Cumulative Gain (DCG) is then obtained through Equation 6, which penalizes highly relevant documents appearing lower on a search result list as the graded relevance value is logarithmically reduced proportionally to the result's position.

$$DCG[i] = \begin{cases} CG[i], & \texttt{if } i < b \\ DCG[i-1] + G[i]/\log_b i, & \texttt{if } i \geq b. \end{cases} \tag{6}$$

DCG values can be compared to the theoretically best possible score vector, the ideal vector, which is represented by Equation 7 for relevance scores 0 and 1, where m is the number of relevant items.

$$BV[i] = \begin{cases} 1, & \texttt{if } i \leq m \\ 0, & \texttt{otherwise.} \end{cases} \tag{7}$$

Normalizing the values is a required step when comparing two techniques under DCG. This is made by dividing the DCG vectors by the corresponding ideal DCG vectors (iDCG). The NDCG is then calculated using Equation 8 which, likewise Precision@K, only evaluates the top k results (Manning et al., 2008).

$$NDCG(k) = \frac{DCG(k)}{iDCG(k)} \tag{8}$$

7 RESULTS

This section presents and discusses the results of each evaluation metric, considering the methodology and objectives proposed by this work. Before the tests, we built the user models from the set of items positively evaluated by each user. The concept of positive rating was constructed differently for each domain dataset, movies, and music, according to the methodology explained in section 6.

The personalization step results in lists of ranked properties, that are used to feed the recommender system, which runs two similarity algorithms: LDSD and Personalized LDSD (PLDSD). The ranked lists may be limited by a k value due to the feature selection preprocessing step. As we consider 3 k values for each strategy — in addition to the turn without limiting the value of k —, and 2 similarity strategies, the experimental evaluation consists of 8 rounds of testing for each user on the movie dataset and another 8 rounds for each user on the music dataset.

7.1 Discussion

In order to facilitate the understanding of the results tables, we use the acronym *LDSD* to represent the recommender model built under plain LDSD, and the acronym *PLDSD* to represent the recommender model that considers our personalized approach, as presented in Section 5. Tests that consider the feature selection step are identified by the value assigned to k in the result tables: Table 1 and Table 2. For example, PLDSD k=10 means that results encompass the personalization method and the preprocessing step, that selects the 10 most relevant properties. On the other hand, LDSD k=0 demonstrates the results for when neither customization nor feature summary are applied.

Table 1 summarizes the average (Avg) results of a hundred users from the movie dataset, considering the LDSD scenarios with and without feature selection and PLDSD with and without feature selection. In the movie scenario, each of the 100 user models is composed of at least 7 and at most 52 positively rated movies. The test set is composed of 200 not evaluated movies. The value of k varies from 10 to 535, which is the maximum number of properties for the movie domain in DBpedia. LDSD and PLDSD without feature selection (k=0) are used for comparison purposes.

Results of Table 1 show a statistically significant increase on all metrics when applying the PLDSD with k=10. We can notice that the LDSD approach presents worse results in comparison with the personalized and filtered rounds.

Table 1: Results for the movie dataset considering diverse scenarios.

Strategy	Precision@5	Precision@10	MAP	NDCG
LDSD k=0	0.450	0.450	0.501	0.743
LDSD k=10	0.500	0.450	0.565	0.798
LDSD k=100	0.450	0.425	0.550	0.751
LDSD k=535	0.450	0.425	0.499	0.744
PLDSD k=0	0.450	0.450	0.511	0.757
PLDSD k=10	**0.550**	**0.450**	**0.582**	**0.809**
PLDSD k=100	0.550	0.450	0.581	0.794
PLDSD k=535	0.550	0.425	0.504	0.783

Table 2 summarizes the average (Avg) results of a hundred users from the music dataset, considering LDSD scenarios with and without feature selection and PLDSD with and without feature selection. In the music scenario, each of the 100 user models consists of at least 14 and at most 50 positively rated artists. The test set is composed of other 200 not evaluated artists. The value of k varies from 10 to 512, which is the maximum number of properties for the music domain in DBpedia. LDSD and PLDSD without feature selection (k=0) are used for comparison purposes.

As with the movie domain, the music results shown in Table 2 also achieved higher values when applying PLDSD with k=10. And again the results with LDSD without personalization and without filters show worse results among the rounds of experiments.

The results for the movie dataset experiments show that MAP and NDCG metrics obtained the best values. Additionally, those values are higher when the similarities are calculated using the feature selection step. Figure 3 presents the MAP and NDCG values for both LDSD and PLDSD approach with different

Table 2: Results for the music dataset considering diverse scenarios.

Strategy	Precision@5	Precision@10	MAP	NDCG
LDSD k=0	0.433	0.450	0.491	0.622
LDSD k=10	0.462	0.450	0.520	0.671
LDSD k=100	0.425	0.425	0.503	0.621
LDSD k=512	0.420	0.425	0.488	0.603
PLDSD k=0	0.436	0.450	0.509	0.635
PLDSD k=10	**0.470**	**0.556**	**0.535**	**0.731**
PLDSD k=100	0.451	0.445	0.508	0.647
PLDSD k=512	0.450	0.440	0.500	0.620

values of k. This graphical representation emphasizes the significant growth of approximately 7% in the NDCG for LDSD values and of 8% in the NDCG for PLDSD values. Concerning the MAP metric, experiments achieved 13% of growth in the LDSD rounds and 14% in the NDCG for PLDSD values when applying the feature selection step.

The results of Precision@5 and Precision@10 in the movie experiments show positive and negative oscillations between values, highlighting a positive gain of Precision@5 when calculating PLDSD with the feature selection. Furthermore, all metrics results are higher when the number of properties is reduced in the pre-selection step, both for LDSD and PLDSD, especially when $k = 10$.

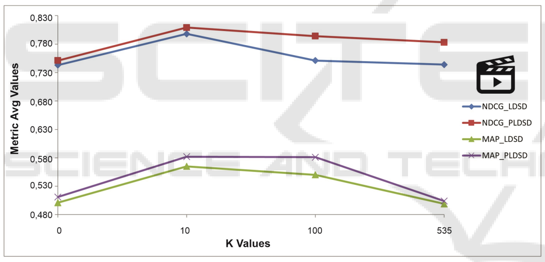

Figure 3: NDCG and MAP results *versus* the k value of pre-selected features from the movie domain.

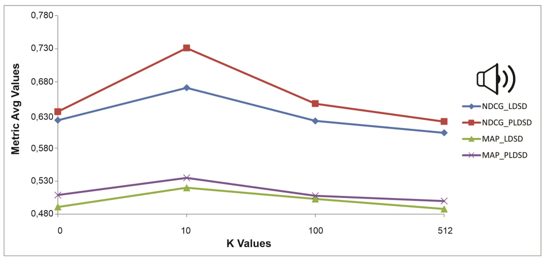

Figure 4: NDCG and MAP results *versus* the k value of pre-selected features from the music domain.

The results of the experiments with the music dataset also showed improvements, although not as expressive as the movie domain, especially for the MAP metric. Figure 4 shows a growth of 8% in the NDCG for LDSD values and of 15% in the NDCG for PLDSD values. Regarding the MAP metric, experiments achieved 6% of growth in the LDSD rounds and 5% in the NDCG for PLDSD values when applying the feature selection step.

Originally, the ABSTAT method behave different depending on the selected knowledge domain (Di Noia et al., 2018). A comparative analysis leads us to conclude that ABSTAT summaries are strongly grounded in the ontological nature of the knowledge graph, while our approach emphasizes user preferences drawn from their previous interactions with the system. This means that PLDSD applied to different domains will perform similarly, although it may vary slightly. Nevertheless, the combined use of the two techniques has been shown to be more efficient than either one separately.

However, further analysis is required to investigate if the nature of the music subject does not allow an accurate prediction of its properties weigths. The maximun number of properties retrieved from DBpedia is similar to both domains — 535 and 512 — although the modeling is very different. A musical artist can be linked to many different songs by the property is dbo:artist of, while an actor is apt to be linked to fewer movies by the property is dbo:star of. For example, the band The Rolling Stones is linked to over 300 songs, while Anthony Hopkins, the actor with one of the most solid careers, could act in only 137 movies so far.

Conversely, the overall performance of the system shows a significant reduction in the processing time, both for the movie and song datasets, since the number of properties being computed is reduced by approximately 80% when considering $K = 10$. Taking the example of user #1 — who positively rated 246 artists — the computation time was reduced from 240 to 67 minutes by adding the selection step, a reduction of 72%.

7.2 Points of Improvement

During the evaluation phase, one difficulty was to define the number of properties that should be used in the pre-selection process, that is, to define the value of k. Due to this fact, the tests were performed considering three distinct values for k. We defined the values so that the algorithm could explore a low ($k = 10$), a medium ($k = 100$) and a high ($k = max$) value of k for pre-selection. The highest value of k is the maximum number of properties returned by the pre-selection step with the dataset used in the tests, which is $k = 535$ for movies and $k = 512$ for music.

Although most of the results were more relevant when $k = 10$, both for LDSD and PLDSD, it is necessary to establish a method that identifies the optimal amount of properties to be analyzed for each data set.

One suggestion for future work is to apply the Elbow Curve, a popular method for finding the optimal number of clusters when working with the K-Means classification algorithm (Kaufman and Rousseeuw, 1990). We propose to investigate the possibility of finding the ideal value to K by adapting the Elbow Method, with the purpose of improving the results personalization strategies of this work.

8 CONCLUSION

In this paper, we proposed personalization strategies for LD-based recommender systems. We use a user modeling process that analyses the past interactions of the user with the system to rank the properties that are used in the recommender model. After ranking the properties, we applied the Personalized Linked Data Semantic Distance (PLDSD) similarity measure, which generates a rank of items to recommend to the user. We run experiments comparing the PLDSD results to the classic LDSD measure using 3 different metrics. We also performed comparative experiments using an adapted implementation of an LD-oriented feature selection strategy, so that only the most relevant properties for the system were considered in part of the calculations.

The evaluation results show the best values for PLDSD combined with a $k = 10$ choice of feature selection strategy, that outperforms the unweighted and not filtered baseline method LDSD. We can state that this work achieved the goal of obtaining better accuracy and performance of LD-based RS when using movie and music datasets from DBpedia. An improvement in the results was noticed when the number of items evaluated was increased and the number of selected properties was reduced with the application of the filtering step. The results of the PLDSD metrics combined with the filter properties stood out from the others in all the tests performed.

As future work, we aim to test our model using other LD-based similarity measures in order to compare and determine which one performs better. We also plan to conduct more studies regarding the feature selection task, by using other LD-driven approaches and comparing them to the baseline methods used so far. Another possible future work is to evaluate this approach using a cross-domain dataset, which would enable the development of multi-domain recommendations for general use in linked datasets.

ACKNOWLEDGEMENTS

This work was partially funded by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) – Grant number 001.

REFERENCES

Berners-Lee, T. (2009). Linked-data design issues. w3c design issue document. *The World-Wide Web Consortium W3C*.

Bizer, C., Heath, T., and Berners-Lee, T. (2009). Linked data - the story so far. *Int. J. Semantic Web Inf. Syst.*, 5(3):1–22.

Blanco., J., Ge., M., and Pitner., T. (2021). Recommendation recovery with adaptive filter for recommender systems. In *Proceedings of the 17th International Conference on Web Information Systems and Technologies - WEBIST,*, pages 283–290. INSTICC, SciTePress.

Cao, Y., Wang, X., He, X., Hu, Z., and Chua, T.-S. (2019). Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences. In *The World Wide Web Conference*, WWW '19, page 151–161, New York, NY, USA. Association for Computing Machinery.

Catherine, R. and Cohen, W. (2016). Personalized recommendations using knowledge graphs: A probabilistic logic programming approach. In *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16, pages 325–332, New York, NY, USA. ACM.

Cheniki, N., Belkhir, A., Sam, Y., and Messai, N. (2016). Lods: A linked open data based similarity measure. *2016 IEEE 25th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)*, pages 229–234.

da Silva, G. O. M., Durão, F. A., and Capretz, M. (2019). Pldsd: Personalized linked data semantic distance for lod-based recommender systems. In *Proceedings of the 21st International Conference on Information Integration and Web-Based Applications and Services*, iiWAS2019, page 294–303, New York, NY, USA. Association for Computing Machinery.

Davoodi, E., Kianmehr, K., and Afsharchi, M. (2013). A semantic social network-based expert recommender system. *Applied Intelligence*, 39(1):1–13.

de Gemmis, M., Lops, P., Musto, C., Narducci, F., and Semeraro, G. (2015). Semantics-aware content-based recommender systems. In Ricci, F., Rokach, L., and Shapira, B., editors, *Recommender Systems Handbook*, pages 119–159. Springer US, Boston, MA.

de Sousa Silva, D. V., de Oliveira, A. C., Almeida, F., and Durão, F. A. (2020). Exploiting graph similarities with clustering to improve long tail itens recommendations. In *Proceedings of the Brazilian Symposium on Multimedia and the Web*, WebMedia '20, page 193–200, New York, NY, USA. Association for Computing Machinery.

Di Noia, T., Magarelli, C., Maurino, A., Palmonari, M., and Rula, A. (2018). Using ontology-based data summarization to develop semantics-aware recommender systems. In Gangemi, A., Navigli, R., Vidal, M.-E., Hitzler, P., Troncy, R., Hollink, L., Tordai, A., and Alam, M., editors, *The Semantic Web*, pages 128–144, Cham. Springer International Publishing.

Di Noia, T., Mirizzi, R., Ostuni, V. C., Romito, D., and Zanker, M. (2012). Linked open data to support content-based recommender systems. In *Proceedings of the 8th International Conference on Semantic Systems*, I-SEMANTICS '12, pages 1–8, New York, NY, USA. ACM.

Gan, L., Nurbakova, D., Laporte, L., and Calabretto, S. (2021). Emdkg: Improving accuracy-diversity trade-off in recommendation with em-based model and knowledge graph embedding. In *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, WI-IAT '21, page 17–24, New York, NY, USA. Association for Computing Machinery.

Guyon, I. and Elisseeff, A. (2003). An introduction to variable and feature selection. *J. Mach. Learn. Res.*, 3:1157–1182.

Järvelin, K. and Kekäläinen, J. (2002). Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446.

Joseph, K. and Jiang, H. (2019). Content based news recommendation via shortest entity distance over knowledge graphs. In *Companion Proceedings of The 2019 World Wide Web Conference*, WWW '19, pages 690–699, New York, NY, USA. ACM.

Kaufman, L. and Rousseeuw, P. J. (1990). *Finding Groups in Data: An Introduction to Cluster Analysis.* John Wiley.

Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Evaluation in information retrieval*, page 139–161. Cambridge University Press.

Meymandpour, R. and Davis, J. G. (2016). A semantic similarity measure for linked data. *Know.-Based Syst.*, 109(C):276–293.

Musto, C., Lops, P., Basile, P., de Gemmis, M., and Semeraro, G. (2016). Semantics-aware graph-based recommender systems exploiting linked open data. In *Proceedings of the 2016 Conference on User Modeling Adaptation and Personalization*, UMAP '16, pages 229–237, New York, NY, USA. ACM.

Passant, A. (2010). Measuring semantic distance on linking data and using it for resources recommendations. In *AAAI Spring Symposium: Linked Data Meets Artificial Intelligence*. AAAI.

Piao, G., Ara, S. s., and Breslin, J. G. (2016). Computing the semantic similarity of resources in dbpedia for recommendation purposes. In Qi, G., Kozaki, K., Pan, J. Z., and Yu, S., editors, *Semantic Technology*, pages 185–200, Cham. Springer International Publishing.

Piao, G. and Breslin, J. G. (2016). Measuring semantic distance for linked open data-enabled recommender systems. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, SAC '16, pages 315–320, New York, NY, USA. ACM.

Ricci, F., Rokach, L., and Shapira, B. (2015). Recommender systems: introduction and challenges. In *Recommender systems handbook*, pages 1–34. Springer International Publishing.

Singh, M., Sahu, H., and Sharma, N. (2018). *A Personalized Context-Aware Recommender System Based on User-Item Preferences*, pages 357–374. Springer International Publishing, Singapore.

van Rossum, B. and Frasincar, F. (2019). Augmenting lod-based recommender systems using graph centrality measures. In Bakaev, M., Frasincar, F., and Ko, I.-Y., editors, *Web Engineering*, pages 19–31, Cham. Springer International Publishing.

Yi, J., Huang, J., and Qin, J. (2018). Rating prediction in review-based recommendations via adversarial auto-encoder. In *2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*. IEEE.