Collaborative Recommendation Method Based on Knowledge Graph for Cloud Services

Weijia Huang[1], Qianmu Li[1], Xiaoqian Liu[2], and Shunmei Meng[1](✉)

[1] School of Computer Science and Technology, Nanjing University of Science and Technology, Nanjing, China
{weijia,qianmu,mengshunmei}@njust.edu.cn
[2] Jiangsu Police Institute, Nanjing, China
liuxiaoqian@jspi.edu.cn

Abstract. As the number of cloud services and user interest data soars, it's hard for users to find suitable could services within a short time. A suitable cloud service automatic recommendation system can effectively solve this problem. In this work, we propose KGCF, a novel method to recommend users cloud services that meet their needs. We model user-item and item-item bipartite relations in a knowledge graph, and study property-specific user-item relation features from it, which are fed to a collaborative filtering algorithm for Top-N item recommendation. We evaluate the proposed method in terms of Top-N recommendation on the MovieLens 1M dataset, and prove it outperforms numbers of state-of-the-art recommendation systems. In addition, we prove it has well performance in term of long tail recommendation, which means that more kinds cloud services can be recommended to users instead of only hot items.

Keywords: Cloud services · Recommendation systems · Knowledge graph · Collaborative filtering

1 Introduction

In recent years, the number of cloud services and user interest data have exploded. Users often don't know which one to choose when facing a large number of cloud service instances, and even many cloud service resources are not known to users. To solve this problem, recommendation system was proposed and has been springing up in the last years. Not only the general public benefit from it, but also service providers do. Greg Linden, Former Amazonian scientist, once said in his blog that at least 20% of Amazon's sales come from recommendation algorithms when he leaves Amazon. Netflix's recommendation algorithm has been well received, and the company has claimed that about 60% of its members customize the rental order based on the recommended list.

Recommendation systems can be classified into two main approaches: collaborative filtering methods and content-based methods. Filtering methods recommend a user items based on the human experience that similar people have similar preferences. Another method provides a user with items that have the similar features with those preferred

X. Zhang et al. (Eds.): CloudComp 2019/SmartGift 2019, LNICST 322, pp. 255–265, 2020.
https://doi.org/10.1007/978-3-030-48513-9_21

by this user. These methods also can be merged into a hybrid system [1]. Furthermore, many works [2–5] focus on making use of semantics when face content-based methods.

Although the usage of semantic information can improve the performance of the recommendation systems, the cost of manually tagging semantic information is very large. Knowledge graph, which represents a wealth of freely available multi-domain ontological knowledge [15], can provide semantic information of items. And we can make use of it to handle the large cost problem.

In this work, we automatically apply semantic analysis into the collaborative filtering approach. Consider the assumption that the semantic information, such as stars, subjects or feedback, extracted from items can represent users' preferences better than items can, and the same kind of semantic information can cover more items, which means collaborative filtering methods making use of it will perform better in the term of long tail. For example, compared with films, its semantic information, such as subjects or stars, may represent a user's taste better. However, the cost of manually tagging semantic information is very large and this information is usually not comprehensive. Therefore, we proposed an approach that using knowledge graph to automatically extract semantic information from items and representing users' preferences through it. Our work consists that: a) use knowledge graph to extract and expand items' semantic information. b) use semantic information that extracted from items to represent a user's preference. We have verified that this method can improve the performance of recommendation, and have the ability to recommend long tail items.

The reminder of this paper is organized as follows: Sect. 2 introduces the UserCF and knowledge graph. In Sect. 3, a collaborative recommendation method based on knowledge graph for cloud services is presented. Section 4 empirically studies the empirical performance and accuracy of our method. Section 5 reviews some related researches. Finally, Sect. 6 concludes this paper and provides some future work.

2 Preliminary Knowledge

2.1 User-Based Collaborative Filtering

User-based Collaborative Filtering (UserCF) is the most popular algorithm in Recommendation System Field, which was proposed for E-mail Filtering System in 1992, and applied to News Filtering by GroupLens two years later.

UserCF is based on the experience that birds of a feather flock together. For example, if you like "Batman", "Mission in the Dish", "Interstellar", "Source Code" and other movies, and some people like these movies, and he also likes "Iron Man", it is very likely that you also like the movie "Iron Man". Therefore, when the system recommends for a user A, it will first find a user group G similar to his interest, and then recommend an item that G likes and A has not heard of to A, which is UserCF.

Based on the above basic principles, we can split the user-based collaborative filtering recommendation algorithm into two steps: First, find a collection of users with similar interests to the target user. Then, find items in the collection that the user likes and that the target user has not heard of are recommended to the target user.

Discover Users with Similar Interests. The proximity between two users is usually measured by the Jaccard formula or cosine similarity. Here, *N(u)* represents the collection of items which user *u* likes, and *N(v)* represents the collection of items that user *v* likes, so the similarity between *u* and *v* is:

Jaccard formula:

$$w_{uv} = \frac{|\mathrm{N}(u) \cap N(v)|}{|\mathrm{N}(u) \cup N(v)|} \tag{1}$$

Cosine similarity:

$$\mathrm{w_{uv}} = \frac{|N(u) \cap \mathrm{N(v)}|}{\sqrt{|N(u)| \cdot |\mathrm{N(v)}|}} \tag{2}$$

Suppose there are currently 4 users: A, B, C, D, and there are 5 items: a, b, c, d, e. A likes a, b and c, B likes a and c, C likes b and e, D likes c, d and e. Taking cosine similarity as an example, the user-similarity matrix is represented as Fig. 1.

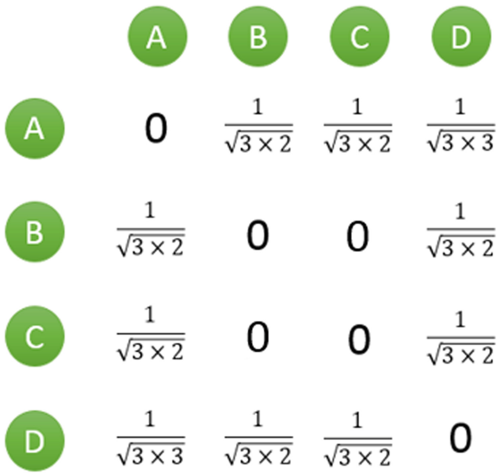

Fig. 1. The user-similarity matrix based on cosine similarity

Recommend. First, we need to find the *K* users most similar to the target user *u* from the matrix, which can be represented as the set *S(u, K)*. Then, we extract all items that users in *S* like, and remove the items that *u* already likes. For each candidate item *i*, the degree of which user *u* is interested in it is calculated by the following formula:

$$\mathrm{p(u, i)} = \sum\nolimits_{v \in S(u,K) \cap N(i)} w_{uv} \times r_{vi} \tag{3}$$

Where r_{vi} is the degree of which user *v* likes *i*.

For example, suppose that we need to recommend user A items, and select $K = 3$ similar users, so similar users are B, C, D. They liked and A did not like the items are c and e. Then respectively calculate *p(A, c)* and *p(A, e)*:

$$p(A, c) = w_{AB} + w_{AD} = \frac{1}{\sqrt{6}} + \frac{1}{\sqrt{9}} = 0.7416 \tag{4}$$

$$p(A, \mathrm{e}) = w_{AC} + w_{AD} = \frac{1}{\sqrt{6}} + \frac{1}{\sqrt{9}} = 0.7416 \tag{5}$$

It seems that user A may have the same level of preference for c and e. In a real recommendation system, just sort by score and take the first few items.

2.2 Knowledge Graph Based on Ontology

Essentially, the knowledge graph is a network with semantics. It is a graphic data structure which consists of Points and Edges. In the knowledge graph, each edge represents a Relationship between two entities, and each node is an Entity existing in the real world. Knowledge graphs are the most effective representation of relationships. In general, a knowledge graph is a network of relationships that connects all the different kinds of information together. Knowledge graphs can help us analyze problems in terms of the relationship perspective.

When knowledge is obtained from various data sources, it is necessary to provide a unified term to merge the knowledge acquired from various data sources into a large knowledge base. The structure or data that provides unified term is called ontology. The ontology not only provides a unified term dictionary, but also builds relationships and restrictions between terms. The ontology allows users to easily create and modify data models based on their own business. The data mapping technology is used to establish the mapping relationship between the terms in the ontology and the vocabulary in different data sources, and then the data of different data sources are merged together.

The knowledge graph based on ontology can help us to add more semantics to the items' description in our approach.

3 Recommendation Approach

3.1 Overview of Our Model

Usually, in our overall model, collaborative filtering is used to predict users' preferences, where user similarity is calculated by user-semantic information relatedness, which are obtained from knowledge maps. As can be seen from Fig. 2, the operations in our model can be broken down into two main phases.

1) Property-specific user-item relations: User-feedback are linked to DBpedia resources to create a new knowledge graph. Then extract user preferences vectors from the graph, which is introduced in Sect. 3.2.
2) Recommend algorithm: Fed preferences vectors into collaborative filtering. Here, we make use of user-semantic information, instead of user-items, to calculate user similarity matrix. We will introduce its details in Sect. 3.3.

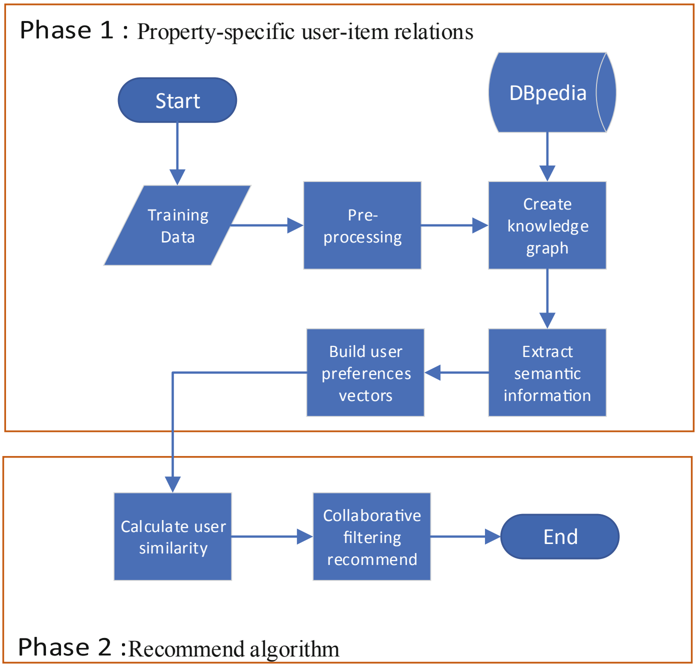

Fig. 2. Framework of our proposed model

3.2 Property-Specific User-Item Relations

As shown in Fig. 3, users have feedback relation with items, and items have property-specific relations with semantic information. We can learn it from the knowledge graph. For example, we can create vector $\rho_{subject}(u)$ which represents subjects that user u prefers. The dimension of $\rho_{subject}(u)$ is the number of nodes in knowledge map with the incoming edge as subjects. The initial values of all the dimensions are 0. On the graph, the traversal starts from user u, and stops at another user. If a dimension in the vector is traversed to the corresponding node on the graph, the value is incremented by 1. In Fig. 3, $\rho_{subject}(Emma) = (2, 1)$, the nodes corresponding to each dimension are Buddy_films and 1941_films.

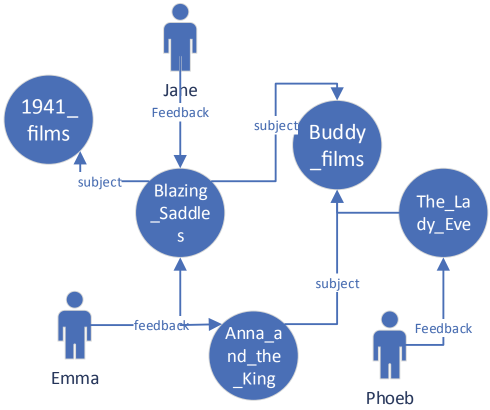

Fig. 3. A sample example for knowledge graph

3.3 Recommend Algorithm

There are two steps in this section: First, find a collection of users who have similar interests to the target user. Then, find items in the collection, which the similar user likes and that the target user has not heard of, and recommend them to the target user.

Discover Users with Similar Interests. In Our approach, we have improved the UserCF algorithm. When measure the proximity of user *u* and *v*, instead of using user-item relations, we make use of the property-specific vector in Sect. 3.1. We take cosine similarity of two vectors as users' similarity. The equation is:

$$\text{similarity}(\mathrm{u}, \mathrm{v}) = \frac{\rho u \cdot \rho v}{||\rho u|| \cdot ||\rho v||} \tag{6}$$

For the graph shown in Fig. 3, the user-similarity matrix is represented as Fig. 4.

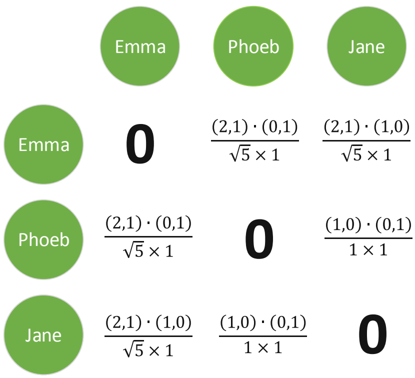

Fig. 4. The user-similarity matrix

Then the steps of finding the *K* most similar users of user *u* and generating the Top-N recommendation list of user *u* are the same as UserCF mentioned in Sect. 2.

Recommend. First, we need to find the K users most similar to the target user u, which can be represented as the set $S(u, K)$. Then, we extract all items that users in S like, and remove the items that u already likes. For each candidate item i, the degree of which user u is interested in it is calculated by the following formula:

$$\mathrm{p(u, i)} = \sum\nolimits_{v \in S(u,K) \cap N(i)} w_{uv} \times r_{vi} \tag{7}$$

Where r_{vi} is the degree of which user v likes i.

Then sort candidate items and recommend u top N items.

4 Experiment

4.1 Experimental Setup

Dataset: In this work, we use MovieLens 1M [10] to evaluate the proposed approach. It is a prevalent dataset in the field of RS evaluation, with 1,000,209 anonymous ratings made by 6,040 MovieLens users on approximately 3,900 movies. Items in Movie 1M have been mapped to the corresponding DBpedia entities in [11], and make use of these publicly available mappings to build knowledge graph.

4.2 Performance Metrics

Evaluation of recommendation is important, especially facing the problem that obtaining users' feedback on recommendations is difficult for researchers, and RS researchers need quality measures to evaluate the quality of algorithms in terms of predictions and recommendations. Many works [6–9] focus on quality measures and propose diverse evaluations. Considering this work is Top-N recommendation, we take classification accuracy measures, including precision, recall and F1. For a user u, in the list of recommended lengths of N, the Precision and Recall of u are:

$$\mathrm{Precision}_u = \frac{|R(u) \cap T(u)|}{|R(u)|} \tag{8}$$

$$\mathrm{Recall_u} = \frac{|\mathrm{R(u)} \cap \mathrm{T(u)}|}{|\mathrm{T(u)}|} \tag{9}$$

where, $R(u)$ is the set of items that recommended for user u, and $T(u)$ represents the set of items that u likes in the test dataset.

The average Precision, average Recall and F1 of the system can be defined as follows:

$$\mathrm{Precision} = \frac{\sum_u Precision_u}{N} \tag{10}$$

$$\mathrm{Recall} = \frac{\sum_u Recall_u}{N} \tag{11}$$

$$\mathrm{F1} = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{12}$$

The Ref. [8] proposes several methods measuring the ability of leveraging long tail. In this experiment, we take Coverage which can be defined as follow:

$$\text{Coverage} = \frac{\left|\bigcup_{u} R(u)\right|}{|T|} \tag{13}$$

where T is the set of items in test dataset.

4.3 Performance Evaluation

Figures 5, 6, 7 and 8 report the performance of different approaches on the MovieLens 1M. It is clear that our method significantly outperforms the other methods in terms of recall, precision, F1 and coverage. Figures 5, 6 and 7 show the accuracy results in terms of recommended items number N. where it can obtain 0.345 in terms of precision at $N = 5$, 0.065 in terms of recall at $N = 5$, and 0.109 in terms of F1 at $N = 5$. The result shows that our approach has a good Top-N recommendation accuracy. As far as its own results are concerned, the precision decreases with the increase of N, and the recall increases with the increase of N, which is in line with our expectations. Within a certain interval, as N increases, $|R(u) \cap T(u)|$ increases, but $|R(u)|$ increases more, which makes Precision decrease. And with $|T(u)|$ unchanged, Recall increases.

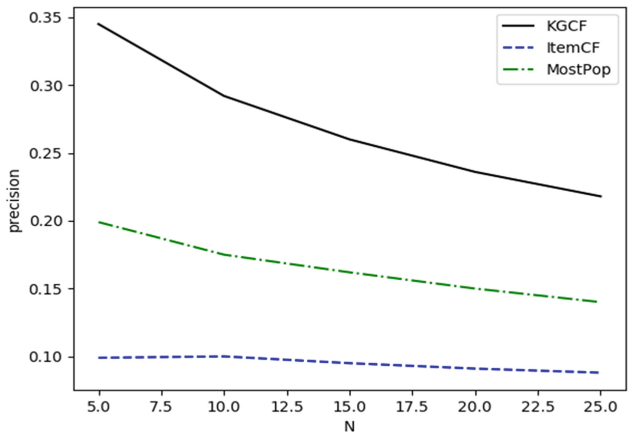

Fig. 5. Precision under different N

Figure 8 shows the coverage result in terms of recommended items number N. We can see that KGCF has well performance in term of long tail recommendation, where it can obtain 0.1320 in terms of coverage.

Obviously, our approach is able to improve the performance of personalized recommendation, and can explore long tail items.

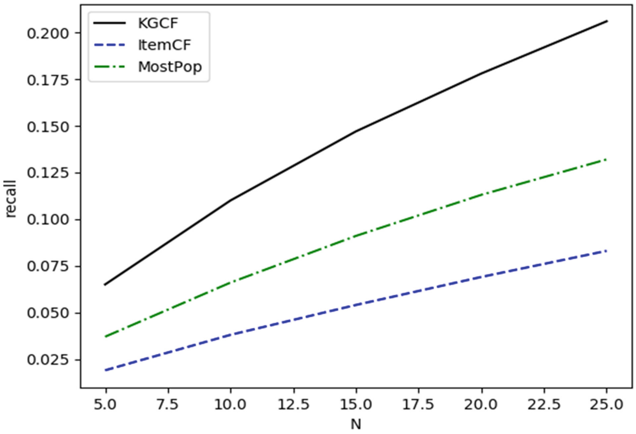

Fig. 6. Recall under different N

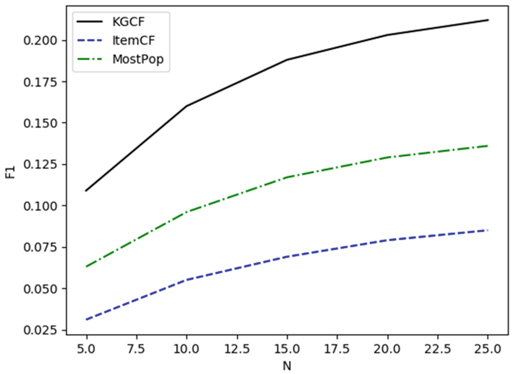

Fig. 7. F1 under different N

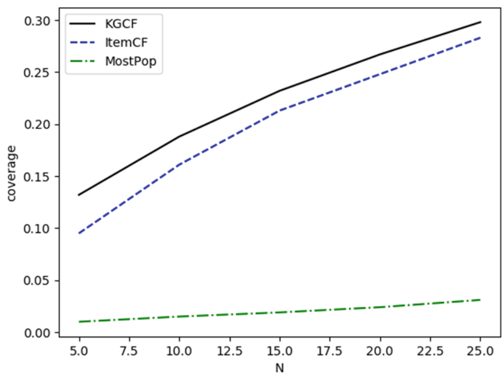

Fig. 8. Coverage under different N

5 Related Work

Recommendation systems based on ontology and semantics have been proposed in many of the past work. [16] proposes an ontology recommendation system that calculates collaborative recommendations by semantic user profiles, thereby improving recommendation accuracy and reducing cold start. [17, 18] use a hybrid graph-based data model to extract meta-path-based features using Linked Open Data, which were used to learn the ranking framework. In [19], entity2rec is proposed to learn user-item correlation from knowledge graph. [20] makes use of the linked data source to calculate the top N recommendations. [21] uses link data and user diversity for an event recommendation system. [22] proposes SemanticSVD++, which is the semantic-aware extension of the SVD++ model and incorporates the semantics of the project into the model.

6 Conclusion

In this work, we propose KGCF, a novel method to recommend users cloud service instances that meet their needs. We model user-item and item-item relatedness in a knowledge graph, and study property-specific user-item relations from it, which are fed to a collaborative filtering algorithm for Top-N item recommendation. This method has two major benefits: it can automatically extract semantic information from items which saves a lot of manpower and time, and it effectively digs long tail items. The experiment on the MovieLens 1M dataset proves that the proposed approach outperforms collaborative filtering approaches based on items and the Most Popular items strategy, and it has well performance in term of long tail recommendation, which means that more kinds cloud service instances can be recommended to users instead of only hot items. We will consider a more comprehensive evaluation in future work, and improve the feasibility of the proposed method in a cloud services recommendation scenario.

Acknowledgment. This paper is supported by the National Science Youth Foundation of China under Grant No. 61702264, the Fundamental Research Funds for the Central Universities No. 30919011282, the Fundamental Research Funds for the Central Universities No. 30918014108, the Natural Science Foundation of the Jiangsu Higher Education Institutions of China No. 19KJB510022.

References

1. John, K., Niu, Z., Kalui, D.: A hybrid recommender system for e-learning based on context awareness and sequential pattern mining. Soft. Comput. **22**(8), 2449–2461 (2018)
2. Nilashi, M., Ibrahim, O., Bagherifard, K.: A recommender system based on collaborative filtering using ontology and dimensionality reduction techniques. Expert Syst. Appl. **92**, 507–520 (2018)
3. Ahmad, N., Ghauth, K.I., Chua, F.-F.: Utilizing learners' negative ratings in semantic content-based recommender system for e-learning forum. J. Educ. Technol. Soc. **21**(1), 112–125 (2018)

4. Cai, G., Lee, K., Lee, I.: Itinerary recommender system with semantic trajectory pattern mining from geo-tagged photos. Expert Syst. Appl. **94**, 32–40 (2018)
5. John, K., Niu, Z., Mustafa, G.: Knowledge-based recommendation: a review of ontology-based recommender systems for e-learning. Artif. Intell. Rev. **50**(1), 21–48 (2018)
6. Shani, G., Gunawardana, A.: Evaluating Recommendation Systems. In: Ricci, F., Rokach, L., Shapira, B., Kantor, Paul B. (eds.) Recommender Systems Handbook, pp. 257–297. Springer, Boston, MA (2011). https://doi.org/10.1007/978-0-387-85820-3_8
7. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Analysis of recommendation algorithms for e-commerce. In: ACM Conference on Electronic Commerce, pp. 158–167 (2000)
8. Park, Y.J., Tuzhilin, A.: The long tail of recommender systems and how to leverage it. In: Proceedings of the 2008 ACM Conference on Recommender Systems, pp. 11–18 (2008)
9. Hurley, N., Zhang, M.: Novelty and diversity in top-N recommendations analysis and evaluation. ACM Trans. Internet Technol. **10**, 1–29 (2011)
10. Harper, F.M., Konstan, J.A.: The MovieLens datasets: history and context. ACM Trans. Interact. Intell. Syst. (TiiS) **5**(4) (2015). Article 19, 19 pages
11. Ostuni, V.C., Noia, T.D., Sciascio, E.D., Mirizzi, R.: Top-n recommendations from implicit feedback leveraging linked open data. In: Proceedings of the 7th ACM conference on Recommender systems, pp. 85–92. ACM (2013)
12. Xu, X., Liu, Q., Zhang, X., Zhang, J., Qi, L., Dou, W.: A blockchain-powered crowdsourcing method with privacy preservation in mobile environment. IEEE Trans. Comput. Soc. Syst. **6**, 1407–1419 (2019)
13. Qi, L., Chen, Y., Yuan, Y., Fu, S., Zhang, X., Xu, X.: A QoS-aware virtual machine scheduling method for energy conservation in cloud-based cyber-physical systems. World Wide Web J. **23**, 1275–1297 (2019)
14. Qi, L., et al.: Finding all you need: web apis recommendation in web of things through keywords search. IEEE Trans. Comput. Soc. Syst. **6**, 1063–1072 (2019)
15. Bizer, C., Heath, T., Berners-Lee, T.: Linked data-the story so far. In: Semantic Services, Interoperability and Web Applications: Emerging Concepts, pp. 205–227 (2009)
16. Middleton, Stuart E., Roure, D.D., Shadbolt, Nigel R.: Ontology-based recommender systems. In: Staab, S., Studer, R. (eds.) Handbook on Ontologies. IHIS, pp. 779–796. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-540-92673-3_35
17. Di Noia, T., Ostuni, V.C., Tomeo, P., Di Sciascio, E.: SPrank: semantic path-based ranking for top-n recommendations using linked open data. ACM Trans. Intell. Syst. Technol. (TIST) **8**(1), 9 (2016)
18. Ostuni, V.C., Di Noia, T., Di Sciascio, E., Mirizzi, R.: Top-n recommendations from implicit feedback leveraging linked open data. In: Proceedings of the 7th ACM Conference on Recommender Systems, pp. 85–92. ACM (2013)
19. Palumbo, E., Rizzo, G., Troncy, R: entity2rec: learning user-item relatedness from knowledge graphs for top-N item recommendation, pp. 32–36 (2017)
20. Ostuni, V.C., Di Noia, T., Di Sciascio, E., Mirizzi, R.: Top-N recommendations from implicit feedback leveraging linked open data. In: Proceedings of the 7th ACM Conference on Recommender Systems, pp. 85–92. ACM, New York (2013)
21. Khrouf, H., Troncy, R.: Hybrid event recommendation using linked data and user diversity. In Proceedings of the 7th ACM Conference on Recommender Systems, pp. 185–192. ACM, New York (2013)
22. Rowe, M.: SemanticSVD++: incorporating semantic taste evolution for predicting ratings. In: 2014 IEEE/WIC/ACM International Conferences on Web Intelligence, WI (2014)