Using Semantic Technology to Improve Recommender Systems Based on Slope One

Rui Yang, Wei Hu, and Yuzhong Qu

Abstract Slope One is a family of algorithms proposed for collaborative filtering and has been widely deployed on websites' recommender systems. Compared to SVD, LSI, Similarity Fusion, or some other commonly used algorithms, Slope One often gives better performance in usability, realizability, and efficiency. However, its prediction accuracy is sometimes lower than other expensive methods, because it is a collaborative filtering model only based on average rating difference and cannot meet some special or individual requirements. The user's and item's features are also not well considered. In this paper, we propose a new approach for enhancing Slope One using semantic technologies. We explore the implicit relationships between items based on the Linked Data and some measures for computing the semantic distances. The relevance information can be utilized to adjust the weighting when computing the prediction ratings. The approach is easy to be implemented and does not increase the complexity of Slope One hardly. A preliminary experiment is conducted and shows that our approach outperforms the traditional weighted Slope One scheme.

1 Introduction

Recommender systems [1] have made contributions to the success of personalized websites as they can automatically and efficiently choose the items or services suitable to user's interest from huge datasets. Among recommendation approaches,

R. Yang
Department of Computer Science and Technology, Nanjing University, China
e-mail: ryang@smail.nju.edu.cn

W. Hu (✉) • Y. Qu
Department of Computer Science and Technology, Nanjing University, China

State Key Laboratory for Novel Software Technology, Nanjing University, China
e-mail: whu@nju.edu.cn; yzqu@nju.edu.cn

J. Li et al. (eds.), *Semantic Web and Web Science*, Springer Proceedings in Complexity,
DOI 10.1007/978-1-4614-6880-6_2,

collaborative filtering (CF) [9, 15], which tries to predict the utility of items for a particular user based on the items previously rated by other users, is most popular used nowadays.

As the most concise form of nontrivial and rating-based CF models, Slope One algorithm [4, 11] shows quite an easy way to build a CF model based on average rating difference, and it has been widely deployed in some successful real-world Web applications such as inDiscover (an MP3 recommender system) and Value Investing News (a stock market news site). But it does not perform as well as some other expensive algorithms in the rating prediction (the most commonly used evaluation method in a large share of the papers in the field of recommender system) [1, 8]. As studies indicate, the core idea of Slope One is that the average difference may cover up the personality. Unfortunately, the user's personality cannot be covered in some especial cases.

Since Tim Berners-Lee, father of the Semantic Web, published the famous article in Scientific American [2], the Semantic Web (SW) has become a hot research field. Nowadays, more and more theoretical researches are converting to practical applications. In the meanwhile, the Linking Open Data (LOD) cloud [3] has grown considerably. The room for end-user applications instead of the professional tools like sematic search engines and APIs that consume Linked Data is being paid more and more attention. Thanks to the LOD, in the research field of recommender system, we can easily get much more open domain knowledge than before. How to use these semantic data to improve the recommender systems has become a new research hot spot, and it is the main focus of this paper.

In this paper, we propose a new approach to enhance recommender systems based on Slope One. A movie recommender system, whose original data source is MovieLens dataset, is developed to illustrate this approach particularly. In specific, the movie recommender system uses Linked Data (in particular, DBtropes [10]), which offers a vast amount of information of various types of film or television works to compute the prediction ratings. We provide some methods to show how to map resources to Linked Data from a relational database. With these mapping methods, we combine these closed and open data and compute the semantic distance (can also be called the implicit relationship between the items) [5, 7]. We borrow ideas from content-based recommender systems [1] to improve the original Slope One prediction algorithm. Our goal in this paper is not to propose a method to replace the original Slope One in the field recommender system or compare the accuracy of a wide range of CF algorithms. We just want to make an attempt to combine the semantic technologies with traditional CF and increase the number of accuracy without reducing computationally efficiency and simplicity. Results on a preliminary experiment show that our approach outperforms the original Slope One scheme and the scheme only based on semantic distance.

The rest of this paper is structured as follows. Section 2 discusses related studies. In Sect. 3, we introduce the semantic distance calculation method and our approach to map semantic distance to traditional datasets and improve the original Slope One scheme. Experimental results on the MovieLens dataset are reported in Sect. 4. Finally, Sect. 5 concludes the paper with future work.

2 Related Work

State-of-the-art recommender systems can be generally categorized as content-based, collaborative, and hybrid [1, 18]. We focus on the collaborative methods in this paper because they are trendy and not a lot of researchers have combined them with Semantic Web technologies to provide recommendations until now. In the following of this section, we will briefly introduce some mainstream collaborative methods and some Semantic Web technologies which can be used in recommendation systems.

2.1 Collaborative Methods

User-Based Collaborative Filtering. Collaborative filtering systems predict the utility of items for a particular user based on the items previously rated by other users. User-based collaborative filtering is said to be the oldest algorithm in this field [1, 15]. It was proposed in 1992 and first applied to spam filtering system. Nowadays, Digg.com uses it to recommend suitable news to every particular user to solve the information overload problem.

For the online recommendation systems, when a user a needs personalized recommendation service, we can look for some other users who have similar interests with a first and then push the items these users favored (but a did not browse before) to a as recommendations. This idea is the core of user-based collaborative filtering. So we can summarize the above into two steps:

1. Find out the set S of users having similar interests with the target user.
2. Find out the items that users in S like and make a recommendation list based on them.

Step 1's key is to compute the interest similarity of any pair of users. Collaborative filtering usually calculates interest similarity based on behavior similarity, i.e., let $N(u)$ be the set of items that user u has positive feedback and $N(v)$ be the set of items that user v has positive feedback, we can compute the similarity between user u and user v by the Jaccard coefficient:

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

or by the cosine similarity:

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

After we get the similarity between the users, user-based CF finds out K users whose interest similarity with the target user is the lowest, then recommend their favorite items to the target user. A general equation can help us measure user u's interest in item i:

$$p_{ui} = \sum_{v \in S(u,K) \cap U(i)} w_{uv} r_{vi}, \tag{3}$$

where $S(u, K)$ contains K users whose interest is most similar to the user u and $U(i)$ contains users who have rated item i. r_{vi} is the rating of user v to item i or some other value that represents how user v likes item i, sometimes it can also be simplified as 0 and 1 (dislike or like).

Item-Based Collaborative Filtering. Item-based CF is the most widely used algorithm in industry. The world's largest e-commerce site Amazon and online video-sharing site YouTube are using recommender systems based on it. Unlike user-based, item-based CF tries to recommend items to users that are similar to items they like before [1, 15]. For example, this algorithm recommends *The Hobbit* to a user just because *The Lord of the Rings* is in the user's favorite list. It is worth noting that item-based CF calculates the item-to-item similarity based on user behavior instead of items' features. Item A and item B are very similar because most of the users who like A also like B.

After we get the item-to-item similarity, a general equation can help us measure user u's interest in item i:

$$p_{uj} = \sum_{v \in S(j,K) \cap I(u)} w_{ji} r_{ui}, \tag{4}$$

where $S(j, K)$ contains K items which are most similar to item j while $I(u)$ contains the items user u like. w_{ji} here is the similarity between item j and i.

2.2 Weighted Slope One

There is a kind of widely deployed item-based CF algorithms called Slope One in industry. The Slope One algorithms are based on predictors in the form of $f(x) = x + b$. According to user ratings on items, we can get the regression line between any pair items. It is a very simple algorithm because it just uses average difference between the two items' ratings as the single free parameter b [11]. Its final prediction is calculated as follows:

$$p_{ui} = \frac{\sum_{j \in I(u)-i} (\sum_{x \in S_{ji}} \frac{v_{xi} - v_{xj}}{|S_{ji}|} + v_{ui}) |S_{ji}|}{\sum_{j \in I(u)-i} |S_{ji}|}, \tag{5}$$

where S_{ji} is the set of users that have rated both items j and i, I_u is the set of items that user u has rated, and v_{ui} represents the rating of user u to item i.

Different from the original Slope One algorithm, the above equation underlines the number of ratings available from each user. Thus, it is also called the weighted Slope One scheme. Apart from that, a third variant, bipolar Slope One, has also been studied. It tries to resolve another problem that praise and negative feedback on the user's decision-making influence is different. It firstly divides the items into those that the user has rated positively and those that have been rated negatively and then applies the weighted Slope One scheme separately and uses the weighted average as the final result.

In this paper, we will pay more attention to the weighted Slope One because it outperforms the others in MovieLens's datasets and is also the most popular one in industry [4, 11].

2.3 Recommender Systems in the Semantic Web

In the Semantic Web area, researchers try to improve recommender systems with semantic technologies mainly based on the Linking Open Data (LOD) cloud and the content-based recommender system model (different from the item-based model, the content-based model usually only uses items' features to compute the recommendations). We can easily fetch much useful attribute information about the items in our system from LOD, and sometimes this information is not readily available especially when we are starting to build recommender systems. Then we use it to compute the similarity between the items and build the item-based recommender system. For example, the datasets offered by MovieLens only give out titles, years, IMDB's URLs, and genres about movies, while DBtropes, a Linked Data wrapper for TVTropes.org, provides much more information about directors, casts, writers, summaries, and even the users' comments. The information wrapped by Linked Data manifests in the form of RDF triples. So mining the connections between the items would not be a very difficult task with such a rich source of data and the good form. Researchers have convinced that they can calculate a more accurate similarity between items through Linked Data compared with the traditional method. In this paper, we also reuse some of the excellent algorithms.

However, CF-based recommender systems in general perform much better than content-based ones. This is mainly because content-based algorithms ignore the user behavior, thus ignoring the law contained in the items popularity and user behavior. Therefore, its accuracy is relatively low [4].

3 Our Approach

In this section, we will introduce how to use semantic technologies to improve the weighted Slope One scheme. In order to build the movie recommender system which uses both traditional datasets and Linked Data, we follow four steps (see Fig. 1):

1. Identify the relevant subset from LOD (in the form of RDF triples).
2. Map the URIs in RDF triples to the items in traditional datasets and reduce the RDF data for computational optimization.
3. Use the LDSD algorithm to compute the semantic distances and insert them into traditional datasets as item-to-item similarities.
4. Integrate the similarities into the original weighted Slope One scheme and compute the recommendations.

3.1 Linked Data Used in Our Recommender System

Technically, Linked Data is a model for publishing structured data online and dereferenceable URIs are used as identifiers. We give a definition of Linked Data dataset firstly in order to define algorithms using it later [12].

Definition 1 *A Linked Data dataset is a graph* $G = (R, L, I)$*, where* $R = \{r_1, r_2, r_3, \ldots, r_n\}$ *is a set of resources and* $L = \{l_1, l_2, l_3, \ldots, l_n\}$ *is a set of typed links, both resources and typed links are identified by dereferenceable URIs.* $I = \{i_1, i_2, i_3, \ldots, i_n\}$ *is a set of instances of these links between resources such as* $i_k = \langle l_j, r_a, r_b \rangle$.

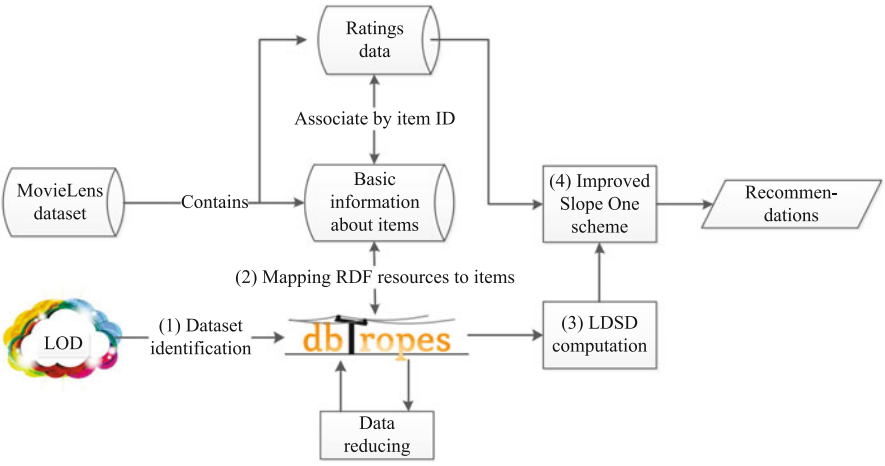

Fig. 1 Overview of the approach

```
http://dbtropes.org/.../TheMatrix
  rdf:type                          http://dbtropes.org/ont/TVTItem ;
  rdfs:label                        The Matrix ;
  rdfs:comment                      What is the Matrix?... ;
  http://skipforward.../hasFeature  http://dbtropes.../int_101087a ;
                                    http://dbtropes.../int_10435f29 ;
                                    http://dbtropes.../int_108bef7e ;
  rdfs:seeAlso                      http://dbtropes.../EnterTheMatrix ;
                                    http://dbtropes.../Starwars ;
```

Fig. 2 An example for an RDF sentence in DBtropes

Linkeddata.org, a widely known and used website, exists to provide a home for, or pointers to, resources across the Linked Data community. In order to meet our needs, we mainly use DBtropes as the source data. There are more than 10,000,000 RDF statements, 22,000 items, 22,000 feature types and 1,750,000 feature instances in DBtropes and DBTropes.org provides a download link to all Internet users. All data are in the form of RDF triples.

Example 1. Figure 2 depicts some RDF triples in DBtropes. From it we can see that there is a direct link between the resource *The Matrix* and the resource *Starwars*, and the links like http://skipforward.net/.../hasFeature can help us to find the relationships. All the information like this is useful to calculate similarities between items.

3.2 Mapping Semantic Distance to Traditional Datasets

In order to use Linked Data dataset and the traditional dataset like MovieLens dataset (providing with rating data and basic information about items and users) together, we have to establish the correspondence between the two datasets that associates each resource in a set with a resource in another set. In this paper, we use the simplest method, string matching, to accomplish this task because DBtropes published its resources in a very structured form. Every film entity is identified by a dereferenceable URI like http://dbtropes.org/.../TheMovieTitle. The last fragment of the URI is the movie title. Thus, we can get the movie titles from MovieLens dataset and easily change the spelling style to make the two successfully matched.

It is hard to give out a general mapping method. We believe that the right method is the best method. For example, if we want to build a book recommender system instead of the movie recommender system, we can use ISBN numbers which can be easily found in both traditional datasets and Linked Data to accomplish the mapping task. For a paper recommender system, the paper titles, authors, journals, and institutes may be helpful. When two or more features are considered, we can also use the Vector Space Model (VSM) and set support threshold to do this work.

3.3 Semantic Distance

As said before, MovieLens' dataset does not provide us with plenty of information about movie's features or some other "noncritical" details. So we have to mine the implicit relationships between items from semantic datasets (Linked Data). Based on Definition 1, the work in [12] defined a *Linked Data Semantic Distance* (LDSD) measure to compute the distance between two resources published as Linked Data. The series of algorithms have been used in some item-based recommender systems (also called content-based) and convinced to be efficient and well performed in some cases. Since [12, 13] has discussed a lot about LDSD, we directly give out the equations here instead of describing and interpreting the details of it. At a glance, the following definitions identify four dimensions to compute the semantic distance between two resources r_a and r_b:

Definition 2 *C_d is a function that computes the number of direct and distinct links between resources in a graph G. $C_d(l_i, r_a, r_b)$ equals 1 if there is an instance of l_i from resource r_a to resource r_b, 0 if not. By extension, C_d can be used to compute the total number of direct and distinct links from r_a to r_b ($C_d(n, r_a, r_b)$) as well as the total number of distinct instances of the link l_i from r_a to any node ($C_d(l_i, r_a, n)$).*

Definition 3 *C_{io} and C_{ii} are two functions that compute the number of indirect and distinct links, both outcoming and incoming, between resources in a graph G. $C_{io}(l_i, r_a, r_b)$ equals 1 if there is a resource n that satisfies both $\langle l_i, r_a, n\rangle$ and $\langle l_i, r_b, n\rangle$, 0 if not. $C_{ii}(l_i, r_a, r_b)$ equals 1 if there is a resource n that satisfies both $\langle l_i, n, r_a\rangle$ and $\langle l_i, n, r_b\rangle$, 0 if not. By extension, C_{io} and C_{ii} can be used to compute the total number of indirect and distinct links between r_a and r_b ($C_{io}(n, r_a, r_b)$ and $C_{ii}(n, r_a, r_b)$, outcoming resp. incoming) as well as the total number of resources n linked indirectly to r_a via l_i ($C_{io}(l_i, r_a, n)$ and $C_{ii}(l_i, r_a, n)$, outcoming resp. incoming).*

We select two measures [12] from the *LDSD* series to compute the similarity (can be also called the semantic distance here). Equation (6) is the first similarity measure named $LDSD_d$ and it only considers direct incoming and outcoming links. Equation (7) is the second similarity measure named $LDSD_i$ and it takes indirect links into account.

$$LDSD_d(r_a, r_b) = \frac{1}{1 + C_d(n, r_a, r_b) + C_d(n, r_b, r_a)}, \tag{6}$$

$$LDSD_i(r_a, r_b) = \frac{1}{1 + C_{io}(n, r_a, r_b) + C_{ii}(n, r_a, r_b)}. \tag{7}$$

3.4 Recommendation Algorithm

Based on the definitions and equations above, we can finally define the prediction algorithm. The prediction rating of user u to item i is computed as follows:

$$p_{ui} = \frac{\sum_{i \in I_u - i_j} \left(\sum_{x \in S_{ji}} \frac{v_{xi} - v_{xj}}{|S_{ji}|} + v_{ui}\right) \log \frac{|S_{ji}|}{LDSD(i_i, i_j)}}{\sum_{i \in I_u - i_j} \log \frac{|S_{ji}|}{LDSD(i_i, i_j)}}, \tag{8}$$

where $LDSD(i_i, i_j)$ here can be $LDSD_d(i_i, i_j)$, $LDSD_i(i_i, i_j)$ or $LDSD_{cw}(i_i, i_j)$. We verify their quality according to the experiment in the next section. Equation (8) is a nonlinear transformation method based on weighted Slope One; it directly changes the calculation method of weights in the original equation. The new weights consider the influence of relationships between items and the relationships are just semantic distance which we mined from the Linked Data. We try to add the weight if there is a firm relationship between item i_i and i_j and decrease it in the opposite situation. In fact, we learn from Term Frequency-Inverse Document Frequency(TF-IDF) [17] when computing the weights.

Corresponding to the nonlinear method above, we also defined a so-called linear method. Formally speaking:

$$\begin{aligned} p_{ui} = (1-\alpha) \times & \frac{\sum_{i \in I_u - i_j} \left(\sum_{x \in S_{ji}} \frac{v_{xi} - v_{xj}}{|S_{ji}|} + v_{ui}\right) |S_{ji}|}{\sum_{i \in I_u - i_j} |S_{ji}|} \\ + \alpha \times & \frac{\sum_{i \in I_u - i_j} \left(\sum_{x \in S_{ji}} (v_{xi} - v_{xj}) + v_{ui} |S_{ji}|\right) \frac{1}{LDSD(i_i, i_j)}}{\sum_{i \in I_u - i_j} \frac{1}{LDSD(i_i, i_j)}}. \end{aligned} \tag{9}$$

The first half of Eq. (9) is the original weighted Slope One, and the latter half takes the LDSD instead of the number of ratings available from each user into account. Free parameter α is used to adjust the proportion. From some angles this method is more scalable. By adjusting the parameter, we can observe the influence of semantic distance in more detail.

4 Evaluation

There are three main experimental methods for evaluating the effects of recommender systems—offline experiment, user study and online experiment [14]. Because our system is not a real commercial system, we mainly focus on the offline experiment.

Table 1 RMSE and MAE results

Methods	RMSE	MAE	Run time
Weighted Slope One (WSO)	1.077	0.789	16.518s
Nonlinear transformation of WSO with $LDSD_d$	1.077	0.789	17.100s
Nonlinear transformation of WSO with $LDSD_i$	1.092	0.802	17.079s
Linear transformation of WSO with $LDSD_d$	1.067	0.764	16.573s
Linear transformation of WSO with $LDSD_i$	1.068	0.771	16.612s

4.1 Prediction Accuracy

Prediction accuracy is usually used to evaluate the capability of predicting user behavior by a recommender system or a recommendation algorithm. This indicator is the most important indicator of recommender system offline experiment.

Rating Prediction. The accuracy of rating prediction is commonly calculated by Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) [1, 14]. For a user u and an item i, let r_{ui} be the actual rating that u gives i while $\hat{r}_{ui}$ be the prediction rating given by recommendation algorithm, RMSE and MAE are defined as follows:

$$RMSE = \sqrt{\frac{\sum_{u,i \in T} \left(r_{ui} - \hat{r}_{ui}\right)^2}{|T|}}, \quad MAE = \frac{\sum_{u,i \in T} |r_{ui} - \hat{r}_{ui}|}{|T|}, \tag{10}$$

where T is the testing set. Netflix thinks RMSE increases penalties for wrong prediction (by square). Therefore, RMSE is more demanding. Researches have shown that if a scoring system is set up based on the integer rating, rounding the prediction ratings would improve MAE results [16].

We select five methods to do the comparison. The methods include the original weighted Slope One (WSO), nonlinear and linear transformation of WSO combined with two kinds of $LDSD$. We predict more than 5,500 ratings given by 442 users in each test.

The results shown in Table 1 can be analyzed from three perspectives. Firstly and most obviously, linear transformation of WSO with $LDSD_d$ has the best performance. Linear transformation of WSO with $LDSD_i$ also has a significant improvement. It indicates that our approach has capability to make rating prediction more accurate. Secondly, two nonlinear methods do not perform well. From this point we can see that replacing the weight directly does not improve the prediction and even declines sometimes. At last, results of run time tell us none of the transformations change the computational complexity.

By comparing the second and third rows (or the fourth and fifth), we found that $LDSD_d$ contains more effective information than $LDSD_i$. Direct links give firmer relationships, while indirect links may bring about some "noises." For linear transformation of WSO with $LDSD_d$, we also observed that the change of results

Fig. 3 RMSE—α

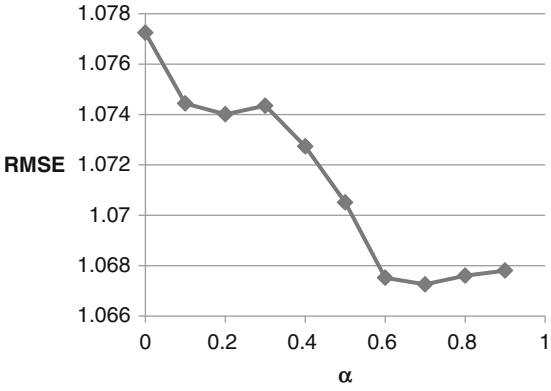

Table 2 Precision results

Method	Precision	Recall	Coverage
Weighted Slope One (WSO)	36.93%	7.14%	48.27%
Soft trans.WSO with $LDSD_d$	39.72%	7.68%	48.19%

when free parameter α varies. In Fig. 3, RMSE is 1.077, which equals the result of original WSO when α is zero and reaches the lowest point when α is in the vicinity of 0.65. This leads us to conclude that the linear integration indeed has a positive effect. The decreasing RMSE shows that $LDSD$ has found the correct implicit relations and is helping us to make more accurate predictions. Separately using semantic distances or original weight is not in line with our expectations and the actual results have convinced it.

Top-N Recommendation. Another important evaluation method is the Top-N recommendation. When a website provides a recommendation service, it in general gives the user a personalized recommendation list, and this is so-called Top-N recommendation [6]. Top-N recommendation's prediction accuracy can be measured by precision/recall.

Let $R_N(u)$ be the recommendation list which contains N items, $T(u)$ be the set of items actually chosen by user and U be the set of all users:

$$Recall = \frac{\sum_{u \in U} |R_N(u) \cap T(u)|}{\sum_{u \in U} |T(u)|}, \quad Precision = \frac{\sum_{u \in U} |R_N(u) \cap T(u)|}{\sum_{u \in U} |R_N(u)|}. \quad (11)$$

Movie recommender systems' ultimate goal is to recommend the movies that users most want to watch but not to predict the ratings. So Top-N recommendation is more in line with the actual requirements of applications. The results in the Table 2 show that the linear transformation of WSO with $LDSD_d$ performs best in this test, i.e., our approach improves the accuracy of the recommendations. Also, our approach does not ignore the influence of user behavior.

4.2 Coverage

Coverage describes a recommender system's ability to explore the items long tail. The simplest definition of the coverage is the proportion of the collection of items that the recommender system can recommend to total items [14]. Let I be the set of all items, the coverage is defined as follows:

$$Coverage = \frac{|\bigcup_{u \in U} R(u)|}{|I|}. \tag{12}$$

The content providers may pay more attention to the coverage. One hundred percent coverage means that the recommender system can recommend each item to at least one user. So a good recommendation system not only needs to have high user satisfaction but also has high coverage. The results in Table 2 indicate that our approach improves the prediction accuracy on the basis of not reducing the coverage. So the diversity of recommendations is preserved.

5 Conclusion

Using semantic technologies to improve recommender system is an emerging research area. In this paper, we proposed a method to integrate recommender system based on Slope One with semantic technologies, which outperforms the original schemes and does not undermine the simplicity and efficiency. We also provided an implementation of our approach and conducted experiments on a well-known movie dataset.

In future work, we look forward to proposing more general mapping methods especially at the instance level, in order to meet more kinds of application requirements. We will study new approaches for incremental semantic distance computation to support dataset update. Additionally, we hope to integrate not only Slope One CF algorithm but also some sophisticated CF algorithms with Linked Data and Semantic Web technologies.

Acknowledgements This work is supported in part by the National Natural Science Foundation of China under Grant Nos. 61003018 and 61021062, in part by the National Research Foundation for the Doctoral Program of Higher Education of China under Grant No. 20100091120041, and also in part by the Natural Science Foundation of Jiangsu Province under Grant No. BK2011189.

References

1. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Trans. Knowl. Data Eng. **17**(6), 734–749 (2005)
2. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Sci. Am. **5**, 29–37 (2001)

3. Bizer, C., Heath, T., Idehen, K.U., Berners-Lee, T.: Linked data on the web. In: Proceedings of WWW, 2008
4. Cacheda, F., Carneiro, V., Fernandez, D., Formoso, V.: Comparison of collaborative filtering algorithms: Limitations of current techniques and proposals for scalable, high-performance recommender systems. IEEE Trans. Web **5**(1), 2:1–2:33 (2011)
5. Cross, V.: Fuzzy semantic distance measures between ontological concepts. In: Proceedings of IEEE Annual Meeting of the Fuzzy Information, 2004
6. Cremonesi, P., Koren, Y., Turrin, R: Performance of recommender algorithms on top-n recommendation tasks. In: Proceedings of RecSys, 2010
7. Ge, J., Qiu, Y.: Concept similarity matching based on semantic distance. In: Proceedings of SKG, 2008
8. Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating collaborative filtering recommender systems. ACM Trans. Inform. Syst. **22**(1), 5–53 (2004)
9. Hill, W., Stead, L., Rosenstein, M., Furnas, G.: Recommending and evaluating choices in a virtual community of use. In: Proceedings of CHI, 1995
10. Kiesel, M., Grimnes, G.A.: DBTropes – A linked data wrapper approach incorporating community feedback. In: Proceedings of EKAW, 2010
11. Lemire, D., Maclachlan, A.: Slope One predictors for online rating-based collaborative filtering. In: Proceedings of SDM (2005)
12. Passant, A.: Measuring semantic distance on linking data and using it for resources recommendations. In: Proceedings of AAAI Spring Symposium Linked Data Meets, 2010
13. Passant, A.: dbrec – Music Recommendations Using DBpedia. In: Proceedings of ISWC, 2010
14. Shani, G., Gunawardana, A.: Evaluating recommendation systems. In: Recommender Systems Handbook, pp. 257–298, 2011
15. Su, X., Khoshgoftaar, T.M.: A survey of collaborative filtering techniques. Adv. Artif. Intell. **2009**(5), 1–19 (2009)
16. Willmott, C.J., Matsuura, K.: Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. Climate Res. **30**(1), 79–82 (2005)
17. Wu, H.C., Luk, R.W.P., Wong, K.F., Kwok, K.L.: Interpreting TF-IDF term weights as making relevance decisions. ACM Trans. Inform. Syst. **26**(3), 13:1–13:37 (2008)
18. Yao, J., Yao, J., Yang, R., Chen, Z.: Product recommendation based on search keywords. In: Proceedings of WISA, 2012

http://www.springer.com/978-1-4614-6879-0

Semantic Web and Web Science

Li, J.; Qi, G.; Zhao, D.; Nejdl, W.; Zheng, H.-T. (Eds.)

2013, IX, 404 p., Hardcover
ISBN: 978-1-4614-6879-0