Similarity-based Matrix Factorization for Item Cold-Start in Recommender Systems

Eduardo P. Fressato, Arthur F. da Costa and Marcelo G. Manzato
Institute of Mathematics and Computer Science
University of São Paulo - São Carlos, SP, Brazil
eduardofressato@usp.br, {fortes, mmanzato}@icmc.usp.br

***Abstract*—In recommender systems (RS) one of the most used approaches is collaborative filtering (CF), which recommends items according to the behavior of similar users. Among CF approaches, those based on matrix factorization are generally more effective because they allow the system to discover the underlying characteristics of interactions between users and items. However, this approach presents the cold-start problem, which occurs because of the system's inability to recommend new items and/or accurately predict new users' preferences. This paper proposes a novel matrix factorization approach, which incorporates similarity of items using their metadata, in order to improve the rating prediction task in an item cold-start scenario. For this purpose, we explore semantic descriptions of items which are gathered from knowledge bases available online. Our approach is evaluated in two different and publicly available datasets and compared against content-based and collaborative algorithms. The experiments show the effectiveness of our approach in the item cold-start scenario.**

I. Introduction

Recommender systems are software tools and techniques that provide suggestions for items to be used by a particular user based on his behavior. Among the most popular techniques are content-based and collaborative filtering approaches [1] [2]. Content-based filtering (CBF) generates recommendations based on the items' attributes (metadata) that the user previously accessed, and recommendations are calculated so that selected items' descriptions are preferred by the user [3]. On the other hand, collaborative filtering depends on the behavior and feedback of users to make recommendations [2]. In general, it analyzes the relationships between items and users to identify new associations among them. The two main collaborative filtering approaches are neighborhood-based methods and latent factor models [1] [2]. Neighborhood methods are centered on the calculation of relationships between items or between users [4]. In turn, latent factor models, also known as matrix factorization models, attempt to explain the judgments that users assigned to items, characterizing items and users by their underlying characteristics [2].

Neighborhood-based models are less scalable and efficient than matrix factorization (MF), since they use association inferences, which have a very high complexity of time and poor scalability [2] [1]. On the other hand, MF allows the usage of a wide range of machine learning algorithms, whose models are trained according to available data [2]. However, some problems are found in both collaborative filtering approaches, such as sparsity, overfitting and cold-start. In particular, item cold-start occurs when the algorithm is unable to recommend new products, because it does not have enough information about them [2] [1]. Although some authors suggest the use of content-based techniques for cold-start recommendations [3] [1], usually items lack detailed and rich descriptions of their content, resulting in poor and/or overspecialized recommendations.

Such problem is known as limited content analysis [3], and a promising solution is to collect semantic information from the Linked Open Data (LOD) cloud [5]–[7]. The LOD cloud provides a variety of semantic knowledge bases for free on the Web in a machine-readable format, covering a wide range of application domains. Using semantic information in RS can provide a better knowledge of items and users, allowing richer representation of data [6] [7]. However, further research is needed, so that both problems of limited content analysis and cold-start can be solved for the majority of applications. In particular, this paper exploits two issues in this scenario: i) how the similarity of items can be computed based on semantic information gathered from knowledge sources; and ii) how semantic similarities among items can be combined in a matrix factorization technique, so that cold-start can be effectively addressed.

In response to these issues, we propose Item-MSMF, a novel hybrid recommender technique based on matrix factorization, that incorporates similarities of items which are calculated based on semantic descriptions from knowledge bases. Our approach is intended to address the item cold-start through a shared latent factor vector representation of similar items based on those items which have enough interactions with users. In this way, as new items representations are not accurate in terms of rating prediction, we propose replacing them with a weighted average of the latent factor vectors of the most similar items which have enough interactions. In this study, we adopt a pure item cold-start scenario [8] [9], i.e., we consider as new items those that have not received any evaluation in the dataset. In the evaluation, we compare our proposal against different approaches of recommender systems in the pure item cold-start scenario to demonstrate the accuracy of the proposal. The experiments are performed on two real datasets in the movie and music domains. Our experiments show that the proposed approach is able to provide better results than baseline algorithms, specially those which incorporate metadata to address cold-start.

978-1-5386-8023-0/18/$31.00 ©2018 IEEE
DOI 10.1109/BRACIS.2018.00066

This paper is structured as follows: in Section II we depict the related work; Section III describes related techniques; in Section IV the proposal is explained in details; Section V describes the system evaluation; and finally, in Section VI we present the final remarks.

II. Related Work

In order to minimize the cold-start problem, several studies incorporate side information about the content of items in matrix factorization models. Usually they adopt one of two approaches: descriptions are incorporated and factorized in the same prediction rule; or similarities of items are incorporated in the factorization.

By combining a collaborative method with a content-based method, Forbes and Zhu [10] propose the Content-Boosted Matrix Factorization (CBMF) algorithm to incorporate metadata of items directly into the MF approach. The CBMF is based on the assumption that the latent factor vector of each item is a function of its attributes, generating three matrices which are used to compute the final predictions: item vs. attribute matrix, attribute vs. latent factor matrix and user vs. latent factor matrix. A similar approach is the gSVD++ [11] algorithm, which also considers implicit feedback of users to enhance item and user representations.

Although both approaches can reduce item cold-start by incorporating items' attributes in the factorization model, training is performed over users' interactions and items' metadata, which can be computationally expensive. Furthermore, such descriptions are expected to be available, which may not be the case for certain domains.

Such problem is known as limited content analysis, which consists in the difficulty of extracting information regarding the content of the items. Observing this limitation, Peska and Vojtas [7] proposed an extension of the CBMF model that uses semantic metadata, which are collected from the LOD (Linked Open Data) cloud. A more robust approach that also exploits semantic information is proposed by Rowe [12], which is an extension to the SVD++ model [4]. In this approach, the evolution of users' tastes is modeled by means of semantic descriptions, which is then combined within the factorization process. In both approaches, the items' metadata are exploited in the training phase of the model, which falls back to the same problem of high computational complexity. In addition, they do not consider the relations among items (similarity), which could be useful for the pure item cold-start scenario.

Regarding items' similarities in a factorization model, Koren [13] proposed a recommendation model that integrates two approaches to collaborative filtering: neighborhood and matrix factorization. In the neighborhood module, a new model is proposed, whose parameters are learned by solving the least-squares problem. In the MF module, the SVD++ prediction rule is incorporated. In this way, the integrated model will sum up the predictions of neighborhood and matrix factorization approaches, allowing them to enrich themselves. Another work that includes the similarity between items in MF is the CoFactor, proposed by Liang et al. [14]. The work proposes a co-factorization, which decomposes the interaction matrix along with a co-occurrence matrix of the item, the co-occurrence between two items codifies the number of users that consumed both items. This co-occurrence matrix can be considered an matrix of similarity between items. The decomposition of the matrices uses the latent factors of the items in a shared way.

These works use the similarities between items, but are not explicitly intended for the item cold-start problem. For instance, items with no interaction cannot be correlated unless metadata are available. In addition, these works add complexity to the prediction rule in order to model such similarities. In our proposal, as opposite, the prediction is kept as simple as possible, so computational cost is reduced at training and prediction phases.

III. Matrix Factorization

This work involves an extension of the well-known matrix factorization algorithm using a neighborhood-based approach in order to smooth the cold-start problem and improve the prediction results.

In the matrix factorization algorithms, each item i is associated with a vector $q_i \in \mathbb{R}_f$ and each user u is associated with a vector $p_u \in \mathbb{R}_f$. Each component of the item's vector (q_i) represents the degree of relevance of the factor in item i. This relevance occurs in the presence or absence of a certain factor for item i. For user vector components (p_u), each factor represents the degree of interest (appreciation or depreciation) that user u has in the factor. The final rating prediction is made based on the product of the transpose of the item's vector q_i with the user's vector p_u, defined by [15]:

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T p_u. \tag{1}$$

where μ represents the average evaluation of all known evaluations, b_u indicates the observable deviation of the user u from the global mean, and b_i indicates the observable deviation from item i. During the training process, the parameters are trained and adjusted for better accuracy in prediction, minimizing the quadratic error:

$$\min_{p_*, q_*, b_*} \sum_{(u,i) \in K} (r_{ui} - \mu - b_u - b_i - q_i^T p_u)^2 + \lambda(b_u^2 + b_i^2 + ||q_i||^2 + ||p_u||^2) \quad , \tag{2}$$

where λ is the regularization parameter, usually defined by cross-validation. Solving Equation 2 is typically accomplished through Stochastic Gradient Descent [15].

It is worth mentioning that in cases of pure item cold-start, the new item i will not have any interaction during training, which causes factor vector q_i to be filled with zero or random values, depending on implementation. Thus, the prediction rule in Equation 1 for item i will be reduced to $\hat{r}_{ui} = \mu + b_u$, which does not consider any additional information about i. In next section, such deficiency is exploited by means of a new matrix factorization model, which uses semantic information to correlate items in order to create shared latent representations of similar items.

IV. Proposed Approach

As mentioned in previous subsections, because the traditional matrix factorization algorithms construct representations of items, their factors will be represented as characteristics based on users' behavior, with scores representing users' overall appreciation or depreciation in relation to those items. However, if an item does not have enough ratings, its representation will be weak, so when the algorithm tries to predict a rating for a particular user, the accuracy of the recommendation will be low.

In order to reduce this problem, this paper proposes Item-MSMF (Items Most Similar based on Matrix Factorization), which is an extension of the prediction process in matrix factorization models. The algorithm consists in a post-processing technique, which is applied after the training process of the user (p_u) and items (q_i) representations. Firstly, an items' similarity matrix is constructed using a correlation metric which is applied in semantic attributes collected from knowledge bases. Next, we compute a weighted average of the latent factors of the k most similar items of each new item. This process aims to generate an accurate representation of these items. Finally, we replace the vector of latent factors of the new items, by their respective new vectors.

In contrast to well-known matrix factorization algorithms that attempt to alleviate the cold-start problem by means of additional complexity of the model, our approach introduces changes only in the prediction step rather than in the training process. This is an advantage because we add only one additional step at the end of the matrix factorization approach. As shown in the experiments, this approach is able to provide better accuracy in the cold-start scenario, and runs much faster with lower computational resources. The following subsections present the algorithm in details.

A. Finding new items

In this work, our focus is the problem of pure item cold-start [8] [9], thus, we consider as new items those that do not have any interaction of the users during training. The main problem with these items is that they are not trained in matrix factorization models, so the latent factor vectors of these items are not adjusted to predict the final classification. This is because the algorithms are usually trained based only on the known interactions available. In this context, the more evaluations an item receives from users, the better the predicted assessments of that item will be for a new user.

B. Finding the most similar neighbors

We extract enriched information that characterizes the items, calculate and identify the most similar items with the items selected in the previous stage. The enriched information used in this work to compute the similarity between two items is the metadata of the items that are extracted from the LOD cloud. This process aims to alleviate the problem of limited content analysis in CBF, since information about items is available and structured ontologically [5].

In this paper, the similarity between a pair of items is obtained through measures of semantic similarity. Such measures are used in a variety of applications, such as detecting communities in social networks, suggesting Web pages, and recommending resources (items). It can be said that two resources are similar by means of semantic similarity measures, which is based on the similarity of their semantic content. It is assumed that if two items share attributes (e.g. genres and actors), then they are related to each other [5] [6]. Thus, the more metadata two items share, the more similar they will be.

In this study, we evaluated a range of similarity metrics, and selected the two which provided the best results. One uses the vector space model (VSM) and the other uses the co-occurrence (COO) of metadata between two items:

1) Similarity VSM for LOD: In order to compute the similarities between the resources, Di Noia et al. [5] have adapted for a LOD-based configuration one of the most popular information retrieval models, VSM. In this model, non-binary weights are assigned to index terms in queries and in documents, and are used to calculate the degree of similarity between each document and query. The authors adapted this model to deal with RDF graphs. The proposed model represents the graph in a three-dimensional matrix in which each slice refers to a property of the ontology and represents its adjacency. Given a property, each item is viewed as a vector whose components refer to the term frequency-inverse document frequency (TF-IDF). For each property, the degree of similarity is the correlation between the two vectors of the items. The authors use the cosine of the angle to compute the similarity between the two vectors, but in our experiments, we also evaluated the Pearson correlation [2]. The three-dimensional matrix can be decomposed into a smaller matrix, where each array refers to a specific RDF property. The rows of each matrix represent the domain (subject) of the property, and the columns represent the range (object). With respect to a pr property, an item i is represented by a vector of weights w containing all terms (objects) related to i through pr. Thus, the degree of similarity between two items i and j by means of the cosine angle is [5]:

$$sim^{pr}_{vsm}(i,j) = \frac{\sum_{n=1}^{t} w_{n,i,pr} \cdot w_{n,j,pr}}{\sqrt{\sum_{n=1}^{t} w^2_{n,i,pr}} \cdot \sqrt{\sum_{n=1}^{t} w^2_{n,i,pr}}} \quad , \quad (3)$$

where t is the total number of terms for property pr.

2) Similarity COO for LOD: The co-occurrence similarity metric is an extension of [14], which aims to explain co-occurrence patterns for the consumption of items through the implicit feedback available in the data set. In our work, the similarity between two items i and j is computed by:

$$sim_{coo}(i,j) = \frac{\#(i,j)}{log(\#i)log(\#j)} \quad , \quad (4)$$

where $\#(i,j)$ is the number of metadata belonging to both i and j, $\#(i) = \sum_{j \in I} \#(i,j)$, $\#(j) = \sum_{i \in I} \#(i,j)$, and I is the set of all items in the collection.

After the similarity between the items is computed, the next step is to identify the most similar items to the new item. In this step, we separate the k most similar items from each new item discovered as described in Subsection IV-A, and are stored for the prediction phase, which is detailed next.

C. Rating Prediction

In this step, we replace the latent factor vector of the new item i by the weighted average of the latent factor vectors of the k most similar items to i: I_i^k. The vectors of the similar items are weighted by the similarity between the new item i and the related items j. To perform the substitution, we consider that the most similar items cannot be new, that is, they must be in the training set. The substitution of the latent factor vectors is performed in the prediction stage based on the set of items not seen by the users in the matrix factorization model. This process can be seen in the Algorithm 1.

Algorithm 1 Item-MSMF

1: **Input:** *train set, test set, similarity matrix*
2: **Output:** *predicted ratings of test set*
3: **for** $i \in$ *test set* **do**
4: **if** $i \notin$ *train set* **then**
5: *new_items_vector* $\Leftarrow i$
6: **end if**
7: **end for**
8: $M \Leftarrow$ *Similarity matrix*
9: *Train* p_u *and* q_i *according to Section III*
10: **for** $(u, i) \in$ *test set* **do**
11: **if** $i \in$ *new_items_vector* **then**
12: $q_i \Leftarrow \frac{\sum_{j \in I_i^k} q_j \cdot sim(i,j)}{\sum_{j \in I_i^k} sim(i,j)}$
13: **end if**
14: *Calculate* $\hat{r}_{ui}$ *with Equation (1)*
15: **end for**

The selection of new items (Lines 1-4) and the calculation of similarity between items (Line 6) is done in a step before matrix factorization training. As can be seen, in the initial process we do not remove the pairs (u, i) that contain new items, we only select items for the prediction process. Thus, we usually train the latent factors with the traditional matrix factorization model and use the vectors p and q for the prediction step. In the rating prediction step (Lines 8-13), we check if item i is new, and if so, we altered its vector of latent factors by the weighted average of the latent factors vectors of the k most similar items. After this process, the prediction is usually performed through the Equation 1.

V. Evaluation

In order to evaluate our approach, we compared our algorithm with the traditional matrix factorization (MF), presented in Section III, and with a content-based approach [16], which uses a similarity matrix based on item attributes to make predictions (called ItemAttr in our experiments). In addition, we compare our approach with gSVD++ [11], which was described in Section II. In this paper, we used the semantic similarities presented in Subsection IV-B to generate the similarity matrix of items. We evaluate our approach in a cold-start scenario using two public available datasets.

A. Simulating Pure Item Cold-Start

To simulate the pure item cold-start, the available items in the datasets were divided into 10 different partitions randomly, (e.g. on a dataset with 1,000 items, each partition will have 100 distinct items). In this way, each partition will be composed by the triples $(u, i, r_{u,i})$ of their respective items. In the following step, we created the training and test sets, whose test set is composed by one partition and the training set by the other nine. Therefore, the items present in the test set will not be present in the training set. We swap these partitions 10 times, getting 10 different folds, which are used in a 10-fold cross-validation protocol.

B. Datasets

We evaluate our approach on two public available datasets from movie and music domains, containing explicit users' interactions.

- **MovieLens** $100k$[1]**:** consists of 100,000 ratings (between 1 to 5) to 943 users and 1,682 movies. As explicit information, we used the ratings that users assigned to items. In our experiment, we used a subset of 1,621 items that had their metadata found, so the subset consists of 99,432 ratings of 943 users.
- **R1-Yahoo! Music**[2]**:** contains users' ratings for musical artists (version 1.0). The data set consists of 115,579,440 ratings of 98,213 music artists per 1,948,882 anonymous users. Evaluations are integers from 0 to 100. From this data set, we filter a subset of items that had their metadata found. Thus, the subset consists of 36,456,966 ratings from 1,713,158 users to 8,486 items. Evaluations were converted to a scale of 0.5 to 5. In our experiment, we used a sample of this dataset, which consists of 217,370 ratings applied by 10,000 users on 3,678 items.

C. Experimental Settings

We evaluate the accuracy of our proposal in a rating prediction scenario. In this scenario, the goal is to predict ratings to unobserved pairs (u, i). We compute the root mean square error (RMSE) of every predicted rating $\hat{r}_{ui}$ in relation to its real rating r_{ui} found in the test set.

All experiments were executed using 10-fold cross-validation, and the average results are presented. We also applied the Wilcoxon [17] test with a $99,99\%$ confidence level to check whether the results are statistically significant or not.

Regarding the parameters of the algorithms, we defined a set of values which performed well for all datasets. For matrix factorization algorithms, we ran experiments for factors equal to 10, 30, 50, and chose 10 as it provided the best results in all datasets. For the ItemAttr, we defined each item's number of neighbors as 20, because it presented better results. For other

[1] http://grouplens.org/datasets/movielens/100k/
[2] https://webscope.sandbox.yahoo.com/catalog.php?datatype=r

parameters, we used default values of each recommender tool and library chosen.

Regarding the LOD cloud, we used the DBpedia[3] knowledge base as it provides extensive coverage across multiple domains, easing the replication of our experiments across multiple datasets. From this knowledge base, we used the *dcterms:subject* property, as it is a general property present in most features. This property relates a resource to its categories. Such categories are used in Wikipedia to help users to structure the knowledge base [5].

To perform queries in DBpedia, we use the SPARQLWrapper[4] which is a Python wrapper that helps to perform SPARQL queries. Such queries were performed using the endpoint[5] available from DBpedia. The search of the item in the knowledge base was performed using the name available in the dataset, so it follows the SPARQL query:

```
SELECT DISTINCT ?uri_item WHERE {
    ?uri_item a dbo:Film.
    ?uri_item rdfs:label ?name.
    FILTER(REGEX(?name, '^name_of_item$|
         ^name_of_item ', "i"))
    FILTER(lang(?name) = 'en') }
```

In addition to filtering by name, in the query we use *rdf:type* to filter the types of resources returned. For MovieLens movies we use the *dbo:Film* and for the musical artists of Yahoo *dbo:Band*. After extracting the URIs from the items, we extracted the metadata using the *dcterms:subject* property, as follows:

```
SELECT DISTINCT ?subject WHERE {
    <uri_item> dct:subject ?subject }
```

Our approach was developed in Python version 3.6, and the source-code is freely available in Case Recommender Framework version 1.0.13[6]. The Framework also contains the used baseline competitors and evaluation metrics.

D. Results

Table I presents the results for the MovieLens and R1-Yahoo Music datasets, using three different values of neighbors k, in order to evaluate the behavior of this parameter to choose the best one of them to compare with the other baselines. For the MovieLens dataset the best value of k was 10, while for the R1-Yahoo Music was 20. In Table II we can notice a considerable reduction in the error of the proposed Item-MSMF algorithm when compared to the baselines, using the best values of k, presented in the Table I. The improvement in prediction is very clear for the new items in both data sets. In MovieLens, Item-MSMF achieved the largest error reduction with a reduction of up to 9.95% compared to traditional MF.

Table III presents a comparison of best performance of the algorithms in terms of computational time using both datasets. Such results were obtained using a personal computer with an Intel Core i7 6800K processor running at 3.40GHz and with 32 GB of DDR4 RAM, with Microsoft Windows 10 OS. From Table III, we note that both Item-MSMF and MF have equivalent times for training and prediction, showing that no additional cost is imposed for new items recommendations. However, the Item-MSMF needs a precomputed similarity matrix, which requires additional time before training. In addition, we can notice a significant difference between the Item-MSMF and the gSVD++, this is because the proposed model does not use metadata in the training step. Finally, comparing the computational times of both Item-MSMF and ItemAttr, we note that: i) the similarity using COO is faster than VSM Cosine to be computed, although for the proposed model COO achieved better accuracy; ii) the prediction time for the proposed Item-MSMF is faster than ItemAttr, so both training and similarity calculations can be accomplished offline for the sake of better prediction accuracy.

[3]http://wiki.dbpedia.org/

[4]https://github.com/RDFLib/sparqlwrapper

[5]http://dbpedia.org/sparql

[6]https://github.com/caserec/CaseRecommender

TABLE I
COMPARISON OF RMSE OF THE ITEM-MSMF ON TWO DATASETS, USING THREE DIFFERENT NUMBER OF NEIGHBORS.

Recommender	k = 10	k = 20	k = 40
Movielens 100k			
Item-MSMF-COO	**0,97892***	0,98355	0,99402
Item-MSMF-VSM Cosine	1,01031	1,01228	1,01981
Item-MSMF-VSM Pearson	1,01130	1,01359	1,02104
R1-Yahoo Music			
Item-MSMF-COO	0,94273	**0,94247***	0,94482
Item-MSMF-VSM Cosine	0,95730	0,95476	0,95464
Item-MSMF-VSM Pearson	0,95741	0,95498	0,95477

Bold typeset indicates the best performance. * indicates statistical significance at p 0.01 pairwise compared to the other results.

TABLE II
COMPARISON OF RELATED MODELS USING RMSE.

Recommender	Movielens 100k	R1-Yahoo Music
MF	1,08714	1,02921
GSVD++	1,04975	1,02958
ItemAttr-COO	1,08500	1,21269
ItemAttr-VSM Cosine	1,07851	1,20877
ItemAttr-VSM Pearson	1,08011	1,20943
Item-MSMF-COO	**0,97892***	**0,94273***
Item-MSMF-VSM Cosine	1,01031	0,95730
Item-MSMF-VSM Pearson	1,01130	0,95741

Bold typeset indicates the best performance. * indicates statistical significance at p 0.01 pairwise compared to the other results.

TABLE III
COMPARISON OF THE COMPUTATIONAL TIME IN SECONDS

Recommender	Similarity	Training	Prediction	Total time
Movilens 100k				
MF	-	42,2271	0,0241	42,2512
gSVD++	-	4099,1318	0,5072	4099,6390
ItemAttr-VSM Cosine	6,3709	2,6709	2,0335	11,0753
Item-MSMF-COO	14,5957	42,6888	0,1705	57,4550
R1 Yahoo - Music				
MF	-	87,8744	0,0558	87,9302
gSVD++	-	4732,9034	0,7278	4733,6312
ItemAttr-VSM Cosine	30,7133	29,1085	2,5234	62,3452
Item-MSMF-COO	72,4976	87,6181	0,8311	160,9468

E. Discussion

The experiments show that Item-MSMF achieved better results than using the trained recommenders with the traditional matrix factorization model in both data sets. This is because we can infer the representativeness of new items by similar items, adding more predictive power to the recommendation system. Our approach consistently improves the performance of prediction for items that have not received any user interaction across all data sets. This is because the vectors of latent factors of the new items are not well trained as they do not receive interactions, that is, the latent factor vectors of the new items are not adjusted to generate the prediction correctly. Thus, when we replace the vector of the new item by the weighted average of the vectors of the most similar items, we obtain better accuracy in prediction.

In addition to comparing our approach against the traditional MF approach, we also compared it against gSVD++, which incorporates items' metadata into its recommendation model. We can observe that our approach obtains better results in both data sets. Because gSVD++ incorporates metadata in its model, it can soften the cold-start problem in some cases, but there are situations in which some metadata may have few interactions resulting in an inaccurate training.

Regaring the content-based ItemAttr algorithm, although we have used the same item similarity matrix for both baseline and proposal, we can see that our approach achieves better accuracy across all metrics and data sets. This shows the effectiveness of using a modified matrix factorization model to address cold-start.

VI. Final Remarks

This paper presented Item-MSMF, an approach based on matrix factorization that explores the similarities among items to replace the latent factor vectors of new items, which in traditional MF approaches are usually disregarded. We compared the proposed method against different baselines, using two datasets of different domains. The results showed the efficiency of our approach in the cold-start scenario.

The main advantage of our approach is the possibility of providing a more representative item factor vector based on its similar neighbors and, consequently, reducing the effects of cold-start. In addition, our approach computes items' similarities based on semantic descriptions, which are gathered from available knowledge bases. Indeed, with little effort our approach can be extended to any matrix factorization algorithms, and use different types of similarity metrics based on metadata to compute items' correlations.

In future work, we intend to apply this technique to different types of metadata and similarity metrics. In addition, we intend to apply this technique of replacing the vectors in more robust matrix factorization algorithms, since it is a flexible technique and applicable in several approaches. We also plan to extend our model to address the user cold-start problem.

Acknowledgment

We would like to acknowledge FAPESP (2016/20280-6) CAPES and CNPq for the financial support.

References

[1] J. Bobadilla, F. Ortega, A. Hernando, and A. GutiéRrez, "Recommender systems survey," *Know.-Based Syst.*, vol. 46, pp. 109–132, Jul. 2013. [Online]. Available: http://dx.doi.org/10.1016/j.knosys.2013.03.012

[2] C. C. Aggarwal, *Recommender Systems: The Textbook*, 1st ed. Springer Publishing Company, Incorporated, 2016.

[3] P. Lops, M. de Gemmis, and G. Semeraro, "Content-based recommender systems: State of the art and trends," in *Recommender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. Springer US, 2011, pp. 73–105.

[4] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, Aug. 2009. [Online]. Available: http://dx.doi.org/10.1109/MC.2009.263

[5] T. Di Noia, R. Mirizzi, V. C. Ostuni, D. Romito, and M. Zanker, "Linked open data to support content-based recommender systems," in *Proceedings of the 8th International Conference on Semantic Systems*, ser. I-SEMANTICS '12. New York, NY, USA: ACM, 2012, pp. 1–8. [Online]. Available: http://doi.acm.org/10.1145/2362499.2362501

[6] G. Piao and J. G. Breslin, "Measuring semantic distance for linked open data-enabled recommender systems," in *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, ser. SAC '16. New York, NY, USA: ACM, 2016, pp. 315–320. [Online]. Available: http://doi.acm.org/10.1145/2851613.2851839

[7] L. Peska and P. Vojtas, "Using linked open data in recommender systems," in *Proceedings of the 5th International Conference on Web Intelligence, Mining and Semantics*, ser. WIMS '15. New York, NY, USA: ACM, 2015, pp. 17:1–17:6. [Online]. Available: http://doi.acm.org/10.1145/2797115.2797128

[8] M. Saveski and A. Mantrach, "Item cold-start recommendations: Learning local collective embeddings," in *Proceedings of the 8th ACM Conference on Recommender Systems*, ser. RecSys '14. New York, NY, USA: ACM, 2014, pp. 89–96. [Online]. Available: http://doi.acm.org/10.1145/2645710.2645751

[9] L. Iaquinta and G. Semeraro, "Lightweight approach to the cold start problem in the video lecture recommendation," vol. 770, 01 2011.

[10] P. Forbes and M. Zhu, "Content-boosted matrix factorization for recommender systems: Experiments with recipe recommendation," in *Proceedings of the Fifth ACM Conference on Recommender Systems*, ser. RecSys '11. New York, NY, USA: ACM, 2011, pp. 261–264. [Online]. Available: http://doi.acm.org/10.1145/2043932.2043979

[11] M. G. Manzato, "gsvd++: Supporting implicit feedback on recommender systems with metadata awareness," in *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, ser. SAC '13. New York, NY, USA: ACM, 2013, pp. 908–913. [Online]. Available: http://doi.acm.org/10.1145/2480362.2480536

[12] M. Rowe, "Semanticsvd++: Incorporating semantic taste evolution for predicting ratings," in *Proceedings of the 2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT) - Volume 01*, ser. WI-IAT '14. Washington, DC, USA: IEEE Computer Society, 2014, pp. 213–220. [Online]. Available: http://dx.doi.org/10.1109/WI-IAT.2014.36

[13] Y. Koren, "Factorization meets the neighborhood: A multifaceted collaborative filtering model," in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '08. New York, NY, USA: ACM, 2008, pp. 426–434. [Online]. Available: http://doi.acm.org/10.1145/1401890.1401944

[14] D. Liang, J. Altosaar, L. Charlin, and D. M. Blei, "Factorization meets the item embedding: Regularizing matrix factorization with item co-occurrence," in *Proceedings of the 10th ACM Conference on Recommender Systems*, ser. RecSys '16. New York, NY, USA: ACM, 2016, pp. 59–66. [Online]. Available: http://doi.acm.org/10.1145/2959100.2959182

[15] Y. Koren and R. Bell, "Advances in collaborative filtering," in *Recommender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. Springer US, 2011, pp. 145–186.

[16] D. Jannach, M. Zanker, A. Felfernig, and G. Friedrich, *Recommender Systems: An Introduction*, 1st ed. New York, NY, USA: Cambridge University Press, 2010.

[17] M. P. Fay and M. A. Proschan, "Wilcoxon-mann-whitney or t-test? on assumptions for hypothesis tests and multiple interpretations of decision rules," *Statistics surveys*, vol. 4, p. 1, 2010.