Enhancing Collaborative Filtering Using Implicit Relations in Data

Manuel Pozo, Raja Chiky, Elisabeth Métais

► To cite this version:

Manuel Pozo, Raja Chiky, Elisabeth Métais. Enhancing Collaborative Filtering Using Implicit Relations in Data. Lecture Notes in Computer Science, 2016, 10.1007/978-3-662-49619-0_7. hal-01314918

HAL Id: hal-01314918

<https://hal.science/hal-01314918>

Submitted on 12 May 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Copyright

Enhancing Collaborative Filtering using Implicit Relations in Data

Manuel Pozo¹, Raja Chiky¹ and Elisabeth Métais²

¹ Institut Supérieur d'Électronique de Paris, LISITE Lab
28, rue Notre-Dame-des-Champs. 75006 Paris, France.

<http://lisite.isep.fr>

manuel.pozo, raja.chiky@isep.fr,

² Conservatoire National des Arts et Métiers, CEDRIC Lab

<http://cedric.cnam.fr>

elisabeth.metais@cnam.fr

Abstract. This work presents a Recommender System (RS) that relies on distributed recommendation techniques and implicit relations in data. In order to simplify the experience of users, recommender systems pre-select and filter information in which they may be interested in. Users express their interests in items by giving their opinion (explicit data) and navigating through the web-page (implicit data). The Matrix Factorization (MF) recommendation technique analyze this feedback, but it does not take more heterogeneous data into account. In order to improve recommendations, the description of items can be used to increase the relations among data. Our proposal extends MF techniques by adding implicit relations in an independent layer. Indeed, using past preferences, we deeply analyze the implicit interest of users in the attributes of items. By using this, we transform ratings and predictions into "semantic values", where the term semantic indicates the expansion in the meaning of ratings. The experimentation phase uses MovieLens and IMDb database. We compare our work against a simple Matrix Factorization technique. Results show accurate personalized recommendations. At least but not at last, both recommendation analysis and semantic analysis can be parallelized, alleviating time processing in large amount of data.

Keywords: collaborative filtering, distributed systems, recommender system, implicit interest.

1 Introduction

The amount of information in the web has greatly increased in the past decade, and it is continuously growing. This makes tough the task of seeking information, and thus users of the Internet may feel overwhelmed when they do not find what they are looking for. These phenomenons has encouraged the development of Recommender Systems (RS). The aim of these systems is to pre-select and to filter information in webs in order to present first those in which users may be more interested. This field has specially raised the attention of the e-commerce to

offer personalized products (a.k.a. items) to users. Thus, one may observe these systems in movie platforms and online-shops, such as video media in Netflix or products in Amazon, but also in article researching and social networks, as Mendely, Google, Facebook or Twitter.

Typically, users express their interest in items by giving opinions (i.e. explicit data) and navigating through the web-pages (i.e. implicit data). For instance, users may rate items (e.g. movies) using a 0-5 stars scale, or they might just click on items links. This data is the interaction between users and items, and for the recommender it represents a feedback of users interest. Hence, recommender systems exploit this available information to predict future interests of users.

In literature, recommendation techniques may be classified in Content-Based (CB), Collaborative Filtering (CF) and Hybrid methods [1]. Content-based takes into account the domain of the recommendation (e.g. movies or books) and it recommends similar items to those the user liked in the past. This carries out overspecialization in recommendations and an item-domain dependency, e.g. always the same genre of movies. Collaborative filtering groups users according to their preferences or tastes, then it recommends items that people from the same group have already liked in the past [2]. Yet, it suffers from cold-start: the system have not yet information about new users/items in order to correctly group them [3]. Among these techniques, Matrix Factorization (MF) has demonstrated high accuracy and easy implementation [4]. In addition, it alleviates time processing in large amount of data by using a parallelizable algorithm. Hybrid methods combine different techniques to alleviate disadvantages and improve the general performance of the global system. In order to increase the quality of the recommendation, trend hybrid techniques seek more relations between users and items by implementing Semantic Technologies [5]. This enhances data representation and help to find out the reasons for which users may or may not be interested in a particular item. However, hybrid systems add complexity and item-domain dependency. In addition, the parallelization of the recommender becomes more difficult.

In this paper, we want to highlight a lack of knowledge in feedback: the interest of users in the attributes of items is hardly captured. Indeed, items contain many attributes (a.k.a. features, such as a movie genre or a movie actor), and moreover they may take several values (such a comedy genre or a concrete actor). This quantity of information makes very difficult to find out the interest of users in these aspects. In fact, (1) users are not willing to give too much explicit information about the features of items, and (2) the large amount of features makes explicit feedback in features inappropriate. For instance, users hardly would rate every actor in a movie.

We claim that the interest of users in these features may render predicted items more acceptable by users. We present a flexible and generic collaborative filtering system that relies on matrix factorization and implicit relations in data. We exploit the description of items and attributes to allow making implicit relations among data. This may help to discover the implicit interest of the user in the attributes of items. The framework scores-up recommendations regarding

not only the preference of users in items, but also their implicit preference in the attributes of the items. Thus, users might be more likely to click on recommendations offered if these recommendations contain features they know and they are interested in.

Indeed, by using this new knowledge we transform ratings into "semantic values", which better represent the interests of users. Thus, the concept of semantic used in this paper to indicate the expansion in the meaning of ratings. That is, this semantic concept does not lead to inferences or reasonings. A similar idea was used in [6], where authors create a matrix of items-attributes.

Experimentations are done in the domain of movies: we use the large set of ratings in MovieLens and attributes of IMDb database. The results achieved show the good performance of our approach compared to a semantic-less matrix factorization approach.

This article is structured as follows: In section 2 related work is presented. Section 3 explains our approach. In section 4 and section 5 the experimentations and evaluations done are shown. Finally, in section 6 conclusions and possible future work are discussed.

2 Related Work

In general, Recommender Systems (RS) use the feedback of users in items in order to predict their interest in other items. In this state of the art we would like to focus on three aspects of typical RS: (1) the scalability of the system, (2) the capacity of the system to incorporate heterogeneous information, and (3) the domain dependency of the system. Looking for a RS that achieves these goals is not trivial. In [1] presents and explains the paradigms of each recommendation technique. Typical CB techniques can incorporate external heterogeneous information from different resources, but they are domain dependent. CF has demonstrated high accuracy and item domain generality, yet difficult to deal with more heterogeneous data. Other hybrid methods usually combines CB and CF in order to improve recommendations. However, the system increases its domain dependency and complexity, and it becomes more difficult to distribute.

CF techniques based on Matrix Factorization (MF) are specially interesting, since they suit with large amount of data. The Matrix Factorization (MF) technique decomposes a matrix R into two random matrices, P and Q , in such a way that the multiplication of both gives approximately the original one. This concept is used in RS to predict the missing rating values of users using the knowns ones [7]. Indeed, this problem can be resolved by using optimization algorithms. The two most known optimization techniques that may find out accurate predictions are based on alternating minimization and gradient descent [8]. On the one hand, alternating minimization techniques have demonstrated to have a simple algebraical problem resolution. It was popularized by the Alternating Least Square (ALS) method [9, 10], and other modifications have been suggested [11, 12]. This technique decomposes the problem into two simple optimization problems represented in P and Q . Then, by fixing one matrix, they

have to guess the other one. Iterating the fixed matrix in order to guess the other one yields in an approximated result for R . In [12], authors use the ALS concept to optimize the overall ranking prediction in top-K recommendations. Recently, [13] expose a detailed theoretical discussion about the optimal usability and the accuracy of ALS methods. On the other hand, the gradient descent optimization technique includes learning-parameters that study the ratings patterns to improve the results of the algorithm. It was popularized by [14] and many improvements and variations have been proposed [8, 15, 16]. In order to minimize the error, this technique iterates among each single entry in R looking for a global minimum. After each iteration, the parameters are updated taking the negative gradient of the function into account. This technique is also known as Singular Value Decomposition (SVD) for recommender systems.

However, these techniques above do not simplify the incorporation of external heterogeneous data. In [8] it is argued that some aspects as the time can be taken into consideration. Yet, still more heterogeneous data can be used to improve the system (e.g. the features of users/items).

In contrast, some authors focus on hybridizations. For instance, [17] suggested a CF and Knowledge-based system to generate multi-type recommendations. A multi-type recommendation suggests not only the goal item, but also some other interesting facts related to the recommended item, e.g. recommending restaurants and the best route to get there. To do that, they use a memory-based CF to compute cosine similarity between experienced cases, and a Case-based Reasoning that adjusts the cases proposed by the CF. Other authors propose Multi-Criteria recommendations [18–22]. Briefly, they consider the ratings from users as a solution for an equation, where the variables are some item attributes. Thus, in order to explain an overall rating in items, they independently analyze explicit ratings given for these attributes and also execute predictions for them. However, these approaches assume the existence of explicit ratings for the attributes of items, but indeed these ratings are hard to get in real-life.

In [23], authors also want to study the interest of users in detail. Their approach uses a three-layer representation, user-interest-item. For a user, an interest is a characteristic that an item must have. For an item, an interest is one of its attributes. Then, they apply a Latent Dirichlet Allocation (LDA) algorithm based on "topic models" from text domains in order to tackle the similar multiple "theme" problem [24]. Hence, authors interpret that the text documents are users, the words are items, and the topics are the latent interests. This extracts hidden interests by establishing a correlation matrix graph about items and interests. This approach shows good performance, although the complexity is not acceptable for large-scale applications.

Other approaches focus on improving the disadvantages of the used recommendation technique. For instance, in [25, 26] authors suggest a CF-CB hybrid system to improve recommendations in an item-based collaborative filtering technique. Authors propose a framework to control the similarity/diversity factor in a top-K recommended items. The approach is based on clustering techniques. The most relevant items are hierarchically ordered and forms trees of interest in

recommendations, what allows creating a zoom-in technique to see more items of the same tree, which tend to be similar.

The usage of Semantic Technologies may facilitate incorporating heterogeneous data to the system, although it also difficulties its domain independency and its scalability. In [5] author propose a state of art for this topic.

On the one hand, some authors propose creating a profile of the interest of users, like [27, 28]. In [27] a hybrid recommender system for TV Programs called Avatar is proposed. It creates structures for both items and users in ontologies and it aims to do inference similarity. Authors use ontologies to implement (1) a content-based technique that computes item similarities, and (2) a collaborative filtering that computes user-profile similarities based on positive and negatives preferences. The system first filters the N most similar user profiles and focus on their positive preferences. After a pre-selection of items that could match to this requirements, they filter out items with negative preferences matches. Finally, they take the top-K items with highest matching values. In the same way, in [28] authors propose a hybrid approach to overcome shortcomings in CB, Knowledge-based and CF. The architecture uses three different agents: Semantic Association Discovery Agent, Data Mining Agent and Random Selection Agent. The former exploits ontologies using knowledge-based techniques to overcome new item problem. The second addresses new user problem. The latter utilizes a random item selection alleviating overspecialization.

On the other hand, other existing approaches focus on better describing items to improve the recommendations. In [29], authors propose a Semantic CB method to improve standard CF techniques. They use item-item similarity based on context pages in Wikipedia to compute artificial ratings for an element. These artificial ratings are used instead of classic rating when we have a very sparse user-item matrix.

In contrast, [30] propose to integrate to RS the social network system tags, where users provide keywords. Tags are mapped in concepts within the ontology, bypassing clustering. The approach creates a matrix of items and concepts and then, it matches the tags of users to concepts in matrix in order to know adequate items for users. Another approach using keywords and ontologies is [31]. They first characterize items using attributes as keywords. Then, they compute item-similarity regarding their keywords. Besides, they reduce the number of keywords by using WordNet as a concept ontology to establish synonyms or similar meanings among keywords.

In this paper we try to achieve the three goals: a capacity for incorporating heterogeneous information, a high level domain genericity and a scalable system. We suggest a flexible and generic collaborative filtering system that relies on matrix factorization and implicit relations in data. The matrix factorization warrants the scalability and domain genericity of the system. The implicit relations in data allows scoring-up items regarding the implicit interest of users in the attributes of items.

3 Suggested architecture

In general, recommender systems still can better exploit the feedback of users. This fact may be achieved by improving current recommendation techniques and incorporating external heterogeneous information of users or items [5]. Matrix Factorization techniques have already demonstrated a highly accurate prediction. In addition, it suits with large sets of data and it is domain independent. However, this technique makes difficult the incorporation of heterogeneous data.

In this work we take advantage of the domain independency and the scalability of Matrix Factorization and we try to improve its heterogeneity constraint. We propose to add an external layer, which will be in charge of the external heterogeneous data. In this layer, items, the attributes of items, users and ratings are analyzed together to find out new implicit relations in data. By exploiting this we transform ratings into "semantic values". Note that the term "semantic" indicates an expansion in the meaning of ratings. Indeed, this new value represents the interests of the users in items and the attributes of the items. Despite this usage, the approach aims to keep a high level of domain independency. As a consequence, in order to achieve presented goals (genericity, scalability and accuracy), we suggest a three-layer recommender architecture: a pre-analysis layer, a semantic layer and a recommender layer. This architecture is shown in the figure 1.

Since the number of attributes and the number of values for the attributes might be huge (e.g. all the actors in a movie, or all movie tags), the pre-analysis module implements a feature selection module and a counting module. The former reduces the number of attributes to focus on. The latter speeds up the system while deeply studying the user interests: we count the implicit number of times that a concrete value for an attribute appears among the rated items of users. The semantic module uses the information deduced in the previous layer in order to transform the ratings of users: we expand the meaning of ratings by adding the implicit relation in data. At last, the recommender module uses an existing collaborative filtering technique based on Matrix Factorization technique to generate accurate recommendations and to keep the high scalability and genericity.

3.1 Pre-analysis module

This layer gathers information from the dataset and the domain description of items (e.g. a database or an ontology), making it abstract and quickly available for next modules. First, we study the relevancy of the attributes in the domain by using Principle Component Analysis (PCA). Then, we analyze the interest of users in these selected attributes and store the deduced information in a fast and low space counting module called Counting Bloom Filter (CBF).

Feature Selection using PCA As long as the number of item attributes might be huge, we apply a reduction technique based on PCA to select the

Fig. 1: Global architecture of the recommender system

most representative attributes. Besides, this technique provides weights for such attributes in order to balance their importance in the recommendation. This weight extraction is achieved by deeper studying the results of PCA. One may see how ratings may be explained by the attributes of items.

Counting Module Ratings reflect the interest of users in items. It is important to understand their item rating-reasons in order to better serve the users. However, an item is composed of several attributes and getting feedback for all of them is complicated. Indeed, users are not willing to rate every single attribute of a movie, e.g. a user may not want to rate every actor in a movie. As a consequence, suggest to implicitly gather this information using the past rated items. For instance, movies with a certain actor might be preferred by users who have rated and liked a movie with this actor. This implicit knowledge should be computed and stored in order to have it quickly available.

On the one hand, databases or semantic technologies as ontologies, describe items environment and they can easily return their unique properties. This fact gives free access to navigate through items features. On the other hand, the implicit information should be stored to have this information quickly accessible. We use using Counting Bloom Filter to this fact.

A Bloom Filter (BF) is a bit structure that allows to represent a set of elements in a lower space size [32]. It uses hash functions in order to efficiently distribute elements among the structure. This filter allows doing fast membership queries, and hence, one may check whether an element is in the structure or not (presence or absence). However, it can not say how many times an element appears in the filter. Counter Bloom Filter (CBF) addresses this constraint by adding counter bits to the filter [33]. These filters have a limitation while doing membership queries: filters assure the absence of an element, but they do not

assure the presence of them. This uncertainty generates a false positive ratio to deal with. However, this error can be estimated and reduced.

Thus, instead of asking the explicit opinion of users in each single value of attributes, we implicitly gather this information by using the description of items and the past rated items. Then we store this implicit value in CBFs. The steps of this module are as follows: (1) for each user we create an empty counting bloom filter, (2) for each rated item by this user, we extract its attributes and (3) finally we insert these attributes in the filter. Thus, the filter contains all the attributes of items which have some relation with the user. Highlight that each user has his own CBF, and these filters are used by the semantic module in order to improve recommendations.

3.2 Semantic module

This module aims to expand the meaning of a rating by incorporating the implicit interest of users in the attributes of items. As said above, an item is composed of several attributes and getting feedbacks for all of them is complicated. The CBF of a user contains the implicit interest of the user in the attributes of an item. We aim to exploit this information in order to add a new sense to users feedback. This expands the meaning of ratings, what we dubbed "semantic values".

The semantic module transforms the initial rating given by users into a new "semantic rating". Indeed, this new value takes into account not only the user preference in the item but also the preference in the attributes of the item. For instance, an item rated as 4 out of 5 may transform its rating value into 4.5. This fact reflects that this item has several attributes in common with the rest of items rated by the user. As a consequence, this boosts the recommendation of items which contain similar attributes to the ones the user liked in the past. Hence, recommended items are more suitable and acceptable by users because they may recognize relevant features for them.

The transformation of the ratings follows the equation presented in (1). Equally, we call it "semantic equation" because it aims to expand the sense of a rating.

$$sv_{u,i} = r_{u,i} + E[r_{u,*}] * \frac{\left| \sum_{j=1}^F C_j * W_j \right|}{N_u} \quad (1)$$

Here, $r_{u,i}$ is the real rating for item "i" given by user "u". N_u is the total number of items rated by user "u". $E[r_{u,*}]$ is the average of the ratings given by user "u". F is the number of selected attributes. W_j are the weights for these attributes computed by PCA. C_j are the number of times that the value of an attribute has appeared for a user, easily got using the computed CBF. Besides, since parameters are pre-calculated, the number of attributes does not have a relevant impact on the execution time of the module. In addition, the process of this equation is easy to parallelize.

Moreover, we use this equation in two different levels of the recommendation. On the one hand, we apply it to all the ratings available in the original training

Fig. 2: Semantic Dataset: input approach

dataset, which is the input approach. On the other hand, we apply the semantic equation to the output of the recommendation. These approaches are explained in the next subsections 3.3 and 3.4.

3.3 Semantic Dataset (input approach)

This approach implements the semantic module at the input of the recommender module. Briefly, it transforms feedback in the training dataset into a semantic feedback, according to the semantic equation (1). That is, for each rating a new "semantic rating" is computed. Hence, a "semantic dataset" is built from the original one. The figure 2 shows this approach. The semantic module takes a training dataset, which contains the "original dataset", and generates a new "semantic dataset", which contains the new "semantic ratings". This latter is used to train the recommender module and create a prediction model to exploit. As the incoming dataset has changed, the recommendation module can return different items.

Remember that collaborative filtering analyzes the ratings of users in order to find out patterns to group similar users. In this approach the recommendation technique still looks for similarities among users, by involving not only items but also attributes. In fact, by increasing the ratings of items in which users are interested (according to their interest in the attributes of items), one helps the recommendation technique to focus on such accuracy and predictions. As a result, the latent space model created by Matrix Factorization learn the importance of these items.

Example Imagine a reduced dataset as shown in the set of tables 1. It contains information about ratings of users in items and the attributes of items, in this case in the domain of movies (genres and actors). This approach takes and

modifies every rating in the dataset according to the implicit interest of users in the attributes of items.

Focus on the rating of the item 1 given by the user 1 ($r_{u,i} = r_{1,1} = 4$). Our goal is to obtain a new "semantic rating" for this value. We first calculate the average of ratings for this user, who has rated $N_u = 4$ movies:

$$E[r_{1,*}] = \frac{4.0 + 3.0 + 1.0 + 2.0}{4} = 2.50 \quad (2)$$

Secondly, we get the weight for attributes computed by PCA (e.g. $W_1 = 0.4$ and $W_2 = 0.6$ for genres and actors respectively). The third step is to get the implicit occurrences stored in CBF:

- The user 1 has rated the items 1, 2, 3 and 4, and these items have actors and genres.
- Focus on the item 1 and its genres: comedy and fantasy. Already rated movies 2 and 4 are comedies, besides the movie 4 is also a fantasy movie. Hence, the occurrences count $C_1 = 3$.
- Focus on the item 1 and its actors: actor 1 and actor 3. The actor 1 also appears on movies 2 and 4. Thus, the occurrences count in this attribute $C_2 = 2$, since the actor 3 does not appear on any other movie.

Putting everything into the equation, we obtain the new "semantic rating":

$$sv_{1,1} = 4.0 + 2.50 * \frac{|3 * 0.4 + 2 * 0.6|}{4} = 5.5 \quad (3)$$

Table 1: Example. (a)Ratings table, (b)Actors table and (c)Genres table

User "u"	Movie "i"	Rating " $r_{u,i}$ "	Movie "i"	Actor	Movie "i"	Genre
1	1	4.0	1	Actor 1	1	Comedy
1	2	3.0	1	Actor 3	1	Fantasy
1	3	1.0	2	Actor 1	2	Comedy
1	4	2.0	3	Actor 2	2	Drama
2	9	4.0	4	Actor 1	3	Thriller
			10	Actor 3	3	Drama
					4	Comedy
					4	Fantasy
					10	Comedy

3.4 Semantic Top-K (output approach)

Recommendations given by this collaborative filtering are pertinent due to its collaborative nature: it analyzes the ratings of users in order to find out patterns to group similar users. However, users may prefer some features in movies

Fig. 3: Semantic top-K: output approach

rather than others, e.g. a movie has a high rating because the user like the actor independently of the genre of the movie. This approach implements the semantic module at the output of the recommender system. It modifies the recommendations done by an already built collaborative filtering recommender in order to insert the interest of users in the attribute of items. This adapts the recommendations to users based on his implicit feedback in the features of items.

For the user to whom recommendations are required, it takes the top-K' ($K' \gg K$) best predictions from the recommender system and transforms the rating predictions of these recommendations. This prediction modification aims to better adapt the recommendations to the users. Indeed, it takes into account the singular preferences of the user in the attributes of the items.

Each item in the top-K' contains a predicted rating which reflects the interest in the item. In fact, this top-K' is usually ordered by this predicted value, hence items in the top are likely more interesting for users. We aim to change this prediction into a "semantic prediction". For this purpose, two sets are required: (1) items in a top-K', and (2) the preferences of the user, i.e. the set of rated items by the user. The former is the recommended item which has attributes and a predicted interest value. The latter implicitly contains attributes in which the user is interested in. In this conditions, we apply the semantic module to change the prediction value. Doing this process among the whole top-K' results in a new resorted top-K', which contains the same items in different positions. Thus, we score up items with similar attributes to the ones the user is interested in. The fact of taking $K' \gg K$ helps the system to put in the top-K new relevant items which initially were out of it. Finally, the system returns the smaller top-K new best items of the re-ordered top-K'. The figure 3 represents this process.

This approach is much faster than the semantic dataset because it requires to transform many less ratings. In addition, since collaborative filtering uses to

return a certain grade of diversity in their predictions [1], we adjust the top-K items according to the interest in items and attributes.

Example In this case, in addition to the tables in 1, we need also a recommended top-K items to modify, like in table 2, where we have an example of recommendations for the user "1". Now, we aim to modify the predicted ratings in the top-K by using the semantic equation in (1). We already know that $E[r_{1,*}] = 2.50$, $W_1 = 0.4$ and $W_2 = 0.6$. The current value to modify is one of the predicted recommendations, for instance the prediction of movie 10 ($r_{u,i} = r_{1,10} = 4.5$). Now we get the implicit occurrences stored in CBF:

- The user 1 has rated the items 1, 2, 3 and 4, and these items contain actors and genres.
- Focus on the item 10 and its genres: comedy. Already rated movies 1 2 and 4 are comedies. Hence, the occurrences count $C_1 = 3$.
- Focus on the item 1 and its actors: actor 3. The actor 3 appears on movie 3. Thus, the occurrences count in this attribute $C_2 = 1$.

Putting everything into the equation, we obtain the new "semantic rating":

$$sv_{1,10} = 4.5 + 2.50 * \frac{|3 * 0.4 + 1 * 0.6|}{4} = 5.625 \quad (4)$$

Applied to the whole top-K', this process provokes a new order in the top-K. This new recommendations are more personalized to the user according to the interest in the attributes of items.

Table 2: Example. Top-3 recommendations for the user "1"

Top-3	Movie 21	Movie 10	Movie 64
Predicted Rating	5	4.5	4

4 Experimentation

Dataset

We suggest using the ratings in MovieLens dataset³ and domain attributes from IMDb⁴ database. This merged dataset is provided by GroupLens [34]. It is composed of 2113 users and 855598 ratings over 10197 movies. It also offers six attributes: genre, directors, actors, countries, locations and tags. The total number of distinct values for these attributes is 112881: 20 movie genres, 95321 actors,

³ <http://grouplens.org/datasets/movielens>

⁴ <http://www.imdb.com/>

Table 3: Experimentation: Weights % for variables in dimensions. Approximate values.

Variables	actor	country	director	genre	location	total
Dimension D_1	19.537	12.719	19.896	0.000	5.064	57,25
Dimension D_2	4.785	8.3732	5.303	6.459	17.823	42.75
total (%)	24	21	25	6	23	100

72 countries, 4266 locations and 13222 tags. Under the authors knowledge, there is not any public and available ontology that perfectly fits in this dataset. Thus, for experimentation purposes, the ontology relations are modeled within a sql database, as done in [6].

Principle Component Analysis

Due to the high number of ratings in the MovieLens dataset, and in order to apply the feature selection, we extract the 100 users who have rated the highest number of movies. Thus, we obtain 169155 ratings, which represent almost the 19.77% of the total ratings in MovieLens dataset. The PCA method analyzes the relevancy of items attributes over this data and returns the most representative features. In addition, it returns relevancy ceiled-weights for these attributes. As is shown in table 3, this module takes out the attribute "tags" since it seems to be, for the PCA, the less relevant over the presented ones.

Counting Bloom Filter

CBF are built in off-line in order to speed up the semantic equation. The dataset contains 2113 users and 112881 different values for the attributes. Regarding the CBF structure, we accept a very low false-positive ratio of 0.01%. In addition, we consider that each value for each user will not appear more than 64 times. That is, we set 6 bits for counting tasks. As a result, the size of one filter corresponding to one single user is around 1.3 Mb. Hence, for the 2113 users the total size of all filters is around 2.7 Gb.

Recommender

We use the SVD algorithm in Apache Mahout⁵ to build the recommender core. This algorithm will iterate a maximum of 30 times to find out the best 30 latent-features that explain the ratings. However, the semantic module uses this recommender as a black box. The experimentations are done in both explained configurations: semantic dataset and semantic top-K approaches.

⁵ <https://mahout.apache.org>

5 Evaluation and results

Our approach uses the features of items and the past preferences of users in order to get a new hidden implicit information about the interest of users in these features. This fact does not directly affect the recommendation. In fact, item similarity measures or items comparisons are not considered, and hence there is not any content-based technique used. Indeed, we exploit these implicit analysis to enhance collaborative filtering recommendations. Due to this assumption, we do not consider our approach a hybrid method: the core of recommendations remains a pure collaborative filtering technique. Because of that, we would like to compare the behavior of our "semantic" recommender system approaches against a non-semantic system.

We aim to study the behavior of the system regarding the ratings in training data. The more training data, the better one can profile a user, and thus, the better one study the implicit interest of users in the attributes of items. In fact, this dependency on the training data corresponds to a study of different sparsity levels. Therefore, for the evaluation of the systems we use the full MovieLens dataset containing 855598 ratings over 10197 movies. To represent the different sparsity levels, we randomly split the dataset into 90%, 80%, 70%, 60% and 50% training sets. The remaining percentage in each level is the test set⁶. As a consequence, we can train systems and compare the predictions in the model with the real-observed values in the test set.

In order to demonstrate the properties of the approaches, we use three different evaluations: a prediction accuracy based one, a ranking accuracy one and an item similarity evaluation.

Finally, note that the graphs show the results of three approaches: svd, semantic dataset and semantic top-K. The former is the semantic-less recommender system. The second implements the semantic at the input of the system. The last uses the semantic at the output of the recommender module.

Outline

This outline aims to give a deployed example of what the recommender systems return. It visually compares top-K returned items from the different approaches. The interest of this outline is to compare the items that different recommenders may show to the same user.

We focus on the user 6757, who is the user with more ratings and hence the best profiled user. (1) In the training set, we subtract 60 out of 119 ratings with the maximum rating score (movies rated with a 5). (2) Then, we train the three different systems in this context: the SVD approach creates a model using this training set, the Semantic Dataset approach first apply the semantic equation to the training set and then creates a model, and the Semantic Top-K approach

⁶ Denote that, since the convergence of the collaborative filtering has been already proved and the semantic approaches do not modify this convergence capability, we do not need a cross-validation set.

Table 4: Experimentation: Top-10 recommendation for user 6757. Items ID and predicted values

SVD		Sem. Dataset		Sem.top-K	
ID	Value	ID	Value	ID	Value
6669	4.34	6669	4.21	6669	4.40
26350	4.20	858	4.14	912	4.31
858	4.189	912	4.13	858	4.239
912	4.186	26350	4.09	8492	4.237
8492	4.16	7749	4.08	26350	4.23
7762	4.128	1221	4.07	3462	4.226
3077	4.1241	3462	4.05	2624	4.224
7749	4.1240	7762	4.03	4806	4.219
4806	4.12	1213	4.027	1221	4.218
2624	4.11	8492	4.026	7256	4.210

modifies the recommendations done by the simple SVD approach. (3) Finally, we ask the systems for a top-60 items for user 6757, expecting to find those ratings deleted from the training set. Table 4 shows the top-10 items (over these 60 movies).

The semantic-less recommendations returns 2 items (858 and 912) which belong to the extracted items. However, the semantic approaches improve this fact: the semantic dataset returns 4 items (858, 912, 1213 and 1221) and the semantic top-K returns 3 items (858, 912 and 1221). This fact is due to the accuracy of the SVD and the extra-knowledge added by the implicit interest in features of items. In addition, we notice the appearance of different items in the semantic approaches (such as item 3462). Specially, we highlight new order in items of the semantic top-K (items 912 or 2624). In fact, we have scored up items which contain interesting attributes for the user, and thus, less interesting items regarding attributes get down in the list. These results show our assumptions: by adding the implicit interest of users in items, recommendations are more suitable and acceptable to users, i.e. more items out of the extracted high scored items set are predicted.

Root Mean Square Error (RMSE)

The RMSE measure evaluates the system in terms of accuracy of the ratings prediction. It represents the standard deviation in the error of the prediction. This error is the difference between predicted values and real-observed values in the test set. Thus, the lower is this error, the better is this metric.

Since our frameworks modify the ratings, they do not overcome the accuracy of the SVD. The reason is that the semantic module scores up items due to the presence of attributes, yet it does not penalize the absence of them. Thus, the semantic rating is always higher than the explicit ratings. These results are shown in figure 4. A further study is being doing to improve this fact.

Fig. 4: RMSE metric comparisons.

Precision, Recall and F-Measure

Precision and Recall techniques measure the relevancy of items in a previously selected top-K. This relevancy is a binary value associated to the item: an item is relevant or not regardless its predicted rating value. Precision represents the percentage of relevant items (items that should be recommended first) over the recommended top-K items. Recall represents the percentage of relevant items over the whole set of items. Figures in 5 and 6 show the results in precision and recall measures.

On the one hand, due to the prediction accuracy of the SVD, the non-semantic system puts easily relevant items in the ranking, and thus precision is high. However, the semantic top-K approach slightly overcomes this precision, since it scores-up items and thus relevant items are likely to appear. On the contrary, in the semantic dataset approach, the ratings modification affects to this accuracy and thus precision is fewer. On the other hand, since we score up items which contain interest attributes for the users, our semantic approaches identify more relevant items among the whole dataset, and hence recall metric are higher, specially in the case of "semantic dataset".

Fig. 5: Precision metric comparisons regarding a top-20 items.

Fig. 6: Recall metric comparisons regarding a top-20 items.

The F-Measure and the F2-Measure are figure of merits for Precision and Recall. The former equally balance the importance of precision and recall. The latter gives the double importance to precision than to recall. Figures 7 and 8 show these metrics. One may observe that by adding a semantic layer improves top-K recommendations, enhancing the overall performance of the system as well. Summing up, scoring up items with common attributes indeed increases the probability of taking relevant items.

Fig. 7: F-Measure metric comparisons regarding a top-20 items.

Fig. 8: F2-Measure metric comparisons regarding a top-20 items.

Intra-List-Similarity (ILS) and Intra-List-Diversity (ILD)

Recommending always too similar items may bore users, and too different items might generate confusion. The ILS (Intra-List-Similarity) metric, also called ILD (Intra-List-Diversity), balances items similarity/diversity among a recommended top-K. In a scale 0-1, the closer is the value to 1, the more similar items in the top-K are between them. On the contrary, the closer to "0", the more diversity exists among recommendations. Typically, collaborative filtering technique tends to show some diversity among its recommendations. Adding a semantic layer

either at the input or at the output of the system, one increases the similarity among recommendations. This added similarity is indeed based on the interest of users in the attributes of items. These facts are shown in figures 9 and 10, where we represent the similarity/diversity measure regarding the genre attribute and the actor attribute of items.

Fig. 9: ILS metric comparisons regarding the attribute "genre".

Fig. 10: ILS metric comparisons regarding the attribute "actor".

6 Conclusion

Recommender systems select, among a huge amount of data, the information in which users might be more interested. In order to do that, these systems exploit the known interest of users in items, which is in an explicit or implicit feedback. In this paper, we highlighted a lack of feedback regarding the attributes of items, which may be really useful for improving recommendations. However this information is hard to retrieve: users are not willing to rate all aspects of items (e.g. all the actors in a movie).

We proposed an approach which relies on collaborative filtering techniques and implicit relations in data. On the one hand, using CF techniques based on MF, one can generate very accurate recommendation in a parallelizable algorithm. Besides, this fact alleviates the analysis of large datasets. On the other hand, the description of items allows making more relations among data. Thus, one can easily extract the implicit interest of users in the attributes of items. Using this information, we suggest modifying the explicit ratings given by users in order to represent also the implicit interest in the attribute of items. We called this new interest representation "semantic value", because it expands the meaning of ratings.

The process is as follows: first we use PCA in order to reduce the number of attributes to focus on. Second, we count how many times a user has liked an item with certain attributes. This fact needs a high processing time which is reduced by using a counting module based on Counting Bloom Filters (CBF). Third, we use this new stored data to modify the ratings of user in items. Finally, new updated recommendations are done using a collaborative filtering matrix factorization technique

The presented architecture is divided in independent layers and allows a flexible usage. Two approaches are presented regarding this architecture: Semantic Dataset and Semantic Top-K. The former acts in the input of the recommender system by analyzing the whole train dataset and modifies the input ratings. Enhancing the relevancy of attributes in the feedback of users, we help the system to focus on such kind of items. The latter aims to apply the semantic layer in the output of the system. Typically, RS provide top-K items ordered by predicted user's preference. In this approach, we score-up the items whose attributes may be of users interest.

Experimentation uses the public and available MovieLens dataset merged with IMDb database. Results show the performance of the approach over different measures. Specially, our approaches enhance the fact of taking relevant items for users. Thus, users might be more likely to click on recommendations because they may contain features they know and they are interested in.

Finally, note that the approaches implement the semantic technologies taking into account process scalability and a high domain independent level. Future work focuses on the penalization in the appearance of non-preferred attributes and on the agility in the counting structure.

References

1. Kantor, P.B., Ricci, F., Rokach, L., Shapira, B.: Recommender systems handbook. In: Recommender systems handbook, Springer, Springer US (2011) 848
2. Breese, J.S., Heckerman, D., Kadie, C.: Empirical analysis of predictive algorithms for collaborative filtering. In: Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence, Morgan Kaufmann Publishers Inc. (1998) 43–52
3. Su, X., Khoshgoftaar, T.M.: A survey of collaborative filtering techniques. *Advances in artificial intelligence* (2009) 4
4. Koren, Y.: The bellkor solution to the netflix grand prize. Netflix prize documentation (2009)
5. Peis, E., del Castillo, J.M., Delgado-López, J.: Semantic recommender systems. analysis of the state of the topic. *Hipertext. net* **6** (2008) 1–5
6. Mobasher, B., Jin, X., Zhou, Y.: Semantically enhanced collaborative filtering on the web. In: Web Mining: From Web to Semantic Web. Springer (2004) 57–76
7. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. *Computer* **42**(8) (2009) 30–37
8. Koren, Y., Bell, R.: Advances in collaborative filtering. In: Recommender Systems Handbook. Springer (2011) 145–186
9. Schafer, J.B., Frankowski, D., Herlocker, J., Sen, S.: Collaborative filtering recommender systems. In: The adaptive web. Springer (2007) 291–324
10. Zhou, Y., Wilkinson, D., Schreiber, R., Pan, R.: Large-scale parallel collaborative filtering for the netflix prize. In: Algorithmic Aspects in Information and Management. Springer (2008) 337–348
11. Pilászy, I., Zibriczky, D., Tikk, D.: Fast als-based matrix factorization for explicit and implicit feedback datasets. In: Proceedings of the fourth ACM conference on Recommender systems, ACM (2010) 71–78
12. Takács, G., Tikk, D.: Alternating least squares for personalized ranking. In: Proceedings of the sixth ACM conference on Recommender systems, ACM (2012) 83–90
13. Jain, P., Netrapalli, P., Sanghavi, S.: Low-rank matrix completion using alternating minimization. In: Proceedings of the 45th annual ACM symposium on Symposium on theory of computing, ACM (2013) 665–674
14. Funk, S.: Netflix update: Try this at home (3rd place) (2006)
15. Takács, G., Pilászy, I., Németh, B., Tikk, D.: Major components of the gravity recommendation system. *ACM SIGKDD Explorations Newsletter* **9**(2) (2007) 80–83
16. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM (2008) 426–434
17. Zhuo, G., Sun, J., Yu, X.: A framework for multi-type recommendations. In: Fuzzy Systems and Knowledge Discovery (FSKD), 2011 Eighth International Conference on. Volume 3., IEEE (2011) 1884–1887
18. Adomavicius, G., Kwon, Y.: New recommendation techniques for multicriteria rating systems. *Intelligent Systems, IEEE* **22**(3) (2007) 48–55
19. Li, Q., Wang, C., Geng, G.: Improving personalized services in mobile commerce by a novel multicriteria rating approach. In: Proceedings of the 17th International Conference on World Wide Web. WWW '08, New York, NY, USA, ACM (2008) 1235–1236

20. Lakiotaki, K., Tsafarakis, S., Matsatsinis, N.: Uta-rec: a recommender system based on multiple criteria analysis. In: *Proceedings of the 2008 ACM conference on Recommender systems*, ACM (2008) 219–226
21. Mikeli, A., Apostolou, D., Despotis, D.: A multi-criteria recommendation method for interval scaled ratings. In: *Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, 2013 IEEE/WIC/ACM International Joint Conferences on. Volume 3. (2013) 9–12
22. Mikeli, A., Sotiros, D., Apostolou, D., Despotis, D.: A multi-criteria recommender system incorporating intensity of preferences. In: *Information, Intelligence, Systems and Applications (IISA)*, 2013 Fourth International Conference on. (2013) 1–6
23. Liu, Q., Chen, E., Xiong, H., Ding, C.H., Chen, J.: Enhancing collaborative filtering by user interest expansion via personalized ranking. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* **42**(1) (2012) 218–233
24. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. *the Journal of machine Learning research* **3** (2003) 993–1022
25. Boim, R., Milo, T., Novgorodov, S.: Direc: Diversified recommendations for semantic-less collaborative filtering. In: *Data Engineering (ICDE)*, 2011 IEEE 27th International Conference on, IEEE (2011) 1312–1315
26. Boim, R., Milo, T., Novgorodov, S.: Diversification and refinement in collaborative filtering recommender. In: *Proceedings of the 20th ACM international conference on Information and knowledge management*, ACM (2011) 739–744
27. Fernández, Y.B., Arias, J.J.P., Nores, M.L., Solla, A.G., Cabrer, M.R.: Avatar: An improved solution for personalized tv based on semantic inference. *Consumer Electronics, IEEE Transactions on* **52**(1) (2006) 223–231
28. Pan, P.Y., Wang, C.H., Horng, G.J., Cheng, S.T.: The development of an ontology-based adaptive personalized recommender system. In: *Electronics and Information Engineering (ICEIE)*, 2010 International Conference On. Volume 1., IEEE (2010) V1–76
29. Katz, G., Ofek, N., Shapira, B., Rokach, L., Shani, G.: Using wikipedia to boost collaborative filtering techniques. In: *Proceedings of the fifth ACM conference on Recommender systems*, ACM (2011) 285–288
30. Mabroukeh, N.R., Ezeife, C.I.: Ontology-based web recommendation from tags. In: *Data Engineering Workshops (ICDEW)*, 2011 IEEE 27th International Conference on, IEEE (2011) 206–211
31. Werner, D., Cruz, C., Nicolle, C.: Ontology-based recommender system of economic articles. *arXiv preprint arXiv:1301.4781* (2013)
32. Bloom, B.H.: Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM* **13**(7) (1970) 422–426
33. Broder, A., Mitzenmacher, M.: Network applications of bloom filters: A survey. *Internet mathematics* **1**(4) (2004) 485–509
34. Cantador, I., Brusilovsky, P., Kuflik, T.: 2nd workshop on information heterogeneity and fusion in recommender systems (hetrec 2011). In: *Proceedings of the 5th ACM conference on Recommender systems. RecSys 2011*, New York, NY, USA, ACM (2011)