Feature Frequency Inverse User Frequency for Dependant Attribute to Enhance Recommendations

Sonia Ben Ticha*†, Azim Roussanaly* , Anne Boyer* and Khaled Bsaïes†
* Lorraine University, KIWI team, Loria Laboratory, Nancy, France.
† Tunis El Manar University, LRPAH Laboratory, Tunis, Tunisia.
Email: sonia.benticha@loria.fr, azim.roussanaly@loria.fr, anne.boyer@loria.fr, khaled.bsaies@fst.rnu.tn

***Abstract*—Recommender system provides relevant items to users from huge catalogue. Collaborative filtering and content-based filtering are the most widely used techniques in personalized recommender systems. Collaborative filtering uses only the user-ratings data to make predictions, while content-based filtering relies on semantic information of items for recommendation. The aim of this work is to introduce the semantic aspect of items in a collaborative filtering process in order to enhance recommendations. Many works have addressed this problem by proposing hybrid solutions. In this paper, we present another hybridization technique that predicts users preferences for items based on their inferred preferences for semantic information of items. For this, we propose a new approach to build user semantic model by using TF-IDF measure and we provide solution to reduce the dimension of data. Applying our approach to real data, the MoviesLens 1M dataset, significant improvement can be noticed compared to usage only approach, Content only approach and hybrid algorithm.**

***Keywords*—hybrid recommender system; collaborative filtering; TF-IDF.**

I. INTRODUCTION

Recommender Systems (RS) provide relevant items to users from a large number of choices. Several recommendations techniques exist in the literature. Among these techniques, there are those that provide personalized recommendations by defining a profile for each user. In this work, we are interested in personalized recommender systems where the user model is based on an analysis of usage. This model is usually described by a user-item ratings matrix, which is extremely sparse ($\geq 90\%$ of missing data).

Collaborative Filtering (CF) and Content-Based (CB) filtering are the most widely used techniques in RS. The fundamental assumption of CF is that if users X and Y rate n items similarly and hence will rate or act on other items similarly [1]. CB filtering assumes that each user operates independently and user will be recommended items similar to the ones he preferred in the past [2]. The major difference between CF and CB recommender systems is that CF uses only the user-item ratings data to make predictions and recommendations, while CB relies on item content (semantic information) for recommendations. However, CF and CB techniques must face many challenges like the data sparsity problem, the scalability problem for large datasets with the increasing numbers of users.

To overcome the disadvantages of both techniques and benefit from their strengths, hybrid solutions have emerged. In this paper, we present a new approach taking into account the semantic information of items in a CF system. In our approach, we design a new hybridization technique, which predicts user preferences for items based on their inferred preferences for item content; and presents a solution to the sparsity and scalability problems. Our system consists of two components: the first builds a new user model, *the user semantic model*, by inferring user preferences for item content; the second computes predictions and provides recommendations by using the user semantic model in a user-based CF algorithm to calculate the similarity between users. The originality of this work is in the building of the user semantic model. Indeed, assuming that items are represented by structured data in which each item is described by a same set of attributes, we build a *user semantic attribute model* for each relevant attribute. With this aim, we define two classes of attributes: *dependent* and *non dependent* and we propose a suited algorithm for each class. User semantic model is then deducted from the horizontal concatenation of all user semantic attribute model. In previous works [3], [4] we have presented solutions based on machine learning algorithm to build a user semantic attribute model for non dependent attribute. In this work, we present a new approach for building a user semantic attribute model for dependent attribute based on TF/IDF measure. Due to the high number of attribute values, and to reduce the expensiveness of user similarity computing, we propose also a solution to reduce the size of the user semantic attribute model. We compare our results to the standards user-based CF, item-based CF, CB and hybrid algorithms. Our approach results in an overall improvement in prediction accuracy.

The rest of paper is organized as follows: Section II summarizes the related work. User semantic model is described in Section III. Section IV describes our approach to build user semantic attribute model for non dependent attribute. Section V describes the recommendation component of our system. Experimental results are presented and discussed in Section VI. Finally, we conclude with a summary of our findings and some directions for future work.

II. RELATED WORK

RS have become an independent research area in the middle 1990s. CF is the most widespread used technique in RS, it was the subject of several researches [5], [6]. In CF, user will be recommended items that people with similar tastes and preferences liked in the past [7]. CB is another important technique; it uses techniques developed in information filtering research [8]. CB assumes that each user operates independently and recommends items similar to the

 ISBN: 978-1-61208-312-4

ones he preferred in the past. To overcome the disadvantages of both techniques and benefit from their strengths, several RS use a hybrid approach by combining CF and CB techniques. The Fab System [9] counts among the first hybrid RS. Many systems have been developed since [10]. Most of these hybrid systems do not distinguish between attributes and treat their values in a same way. Moreover, because of the huge number of items and users, calculating the similarity between users in CF algorithm became very expensive in time computing. Dimension reduction of data is one of the solution to reduce the expensiveness of users similarity computing. Mobasher et al. [11] combine values of all attributes and then apply a Latent Semantic Analysis (LSA) to reduce dimension of data. Sen et al. [12] are inferring user preferences for only one attribute, the item' tags, without reducing dimension. Manzato [13] computes a user semantic model for only the movie genre attribute and applies a Singular Value Decomposition (SVD) to reduce the dimension of data. In our approach, we compute a user semantic attribute model for each relevant attribute and we propose a low cost solution to reduce the data dimension.

III. USER SEMANTIC MODEL

Pazzani et al. [8] have identified three alternative item representations. Item can be represented by *structured data* in which there is a small number of attributes, each item is described by the same set of attributes, and there is a known set of values that each attribute may have, for instance, the attributes of a movie can be *title*, *genres*, *actors* and *director*; *unstructured data* such as *news articles*, *abstract of movie*, *description of restaurant*, is an unrestricted text in which there are no attribute names with well-defined values; or *semi-structured data* in which there are some attributes with a set of restricted values and some free-text. In this paper, we are interested only to items described by structured data. The others representations will be addressed in future work. In the following, we will use the term *feature* to refer to an attribute value, for instance *Documentary*, *Musical* and *Thriller* are features of *movie genre* attribute.

A. Dependent and non Dependent attribute

In structured representation, each attribute has a set of restricted features. However, the number of features can be related or not to the number of items. That is why we have defined two classes of attributes:

- **Dependent attribute**: attribute, which having very variable number of features. This number is closely related to the number of items. So, when the number of items is increasing, the number of features is increasing also. For example: *directors* and *actors of movies*, *user tags*.
- **Non dependent attribute**: attribute, which having a very few variable number of features, and this number is not related to the number of items. Thus, the increasing number of items has no effect on the number of features. For example: *movie genres*, *movie origin* and *cuisine of restaurants*.

In addition, all attributes do not have the same degrees of importance to users. There are attributes more relevant than others. For instance, the *movie genre* can be more significant, in the evaluation criteria of user, than the *origin*. Experiments that we have conducted (see Section VI) confirmed this hypothesis. In this paper, we assume that relevant attributes will be provided by a human expert. Therefore, for each relevant attribute A, we build a *user semantic attribute model* that predicts the users preferences for its features. This model is described by a matrix Q_A (users in lines and features of A in columns). In our approach, we design a suited algorithm for building the *user semantic attribute model* for each class of attribute. For non dependent attribute, due to the low number of features, we have used a clustering algorithm. Section III-B briefly described the operating principle of our solution that have been addressed in previous works [3], [4]. For dependent attribute, we have explored techniques issues from retrieval and information filtering research, Section IV presents our solution for building the *user semantic attribute model* for dependent attribute that is the aim of this paper. The user semantic model for all relevant attributes, described by the matrix Q, is the result of the horizontal concatenation of all user semantic attribute models Q_A.

B. User semantic model for non dependent attribute

Let us denote by S the set of items, U the set of users, s a given item $\in S$, u a given user $\in U$ and a rating value $r \in \{1, 2, ..., 5\} \equiv R$. U_s the set of users that rating the item s, then we define the rating function for item s by $\delta_s : u \in U_s \longmapsto \delta_s(u) \in R$. We denote also by F_A the set of features of attribute A, f a given feature $\in F_A$ and S_f the set of items associated to feature f. For instance if we consider the *movie genre* attribute, S_{action} is the set of all action movies.

An item s is represented by its usage profile vector $s_{up} = (\delta_s(u) - \overline{\delta_u})_{(u=1..|U|)}$, where $\overline{\delta_u}$ is the average rating of all rated items by user u. The idea is to partition all items described by their usage profile in K clusters, each cluster is labeled by a feature $f \in F_A$ (or a set of features).

The number K of clusters and the initial center of each cluster is computed by the initialization step of the clustering algorithm. In initial step, each cluster C_k consists of items in $\bigcup_{f\ labeling\ C_k} S_f$ and labeled by the set of corresponding features; so its center is the mean of its items described by their usage profile vector s_{up}. Moreover, an attribute can be mono valued or multivalued depending on the number of features that can be associated to a given item s. For example, the attribute *movie genre* is multivalued because a movie can have several genres while *movie origin* is a mono valued attribute because a movie has only one origin. Thus, if an attribute is multivalued, s can belong to several clusters C_k, while for mono valued attribute, an item should belong only to one cluster. Therefore, for multivalued attribute, the clustering algorithm should provide non disjointed clusters (a fuzzy clustering), whereas, for mono valued attribute, the clustering algorithm should provide disjointed clusters.

After running the clustering algorithm, we obtain K cluster centers; each center k is described by a vector $c_k = (q_{k,u})_{(u=1..|U|)}$. Thus, the user semantic attribute model is described by the matrix $Q_A = (q_{u,k})_{(u=1..|U|,\ k=1..K)}$.

With non dependent attribute, the number of associated features is low, this is why the clustering is suitable. More-

 ISBN: 978-1-61208-312-4

over, the user semantic attribute model allows an important reduction of dimension and so reduce the expensiveness of user similarity computing. In [4], we have used the Fuzzy CMean Algorithm on the movie genre attribute, we have obtained good performance because the user semantic attribute model has no missing values and all similarities between users were able to be computed. In [3], we have used the KMean clustering algorithm on the movie origin attribute. Because of the missing values in the user item rating matrix, we have proposed an algorithm for the initialization step of the KMean clustering using a movie origin ontology. We obtained good results compared to user-based CF but not as good as the genre attribute.

IV. User semantic model for dependent attribute

For a dependent attribute A, the set F_A of its features can be important and it augments with the increasing of the set of items S. In this paper, we present our solution to compute a user semantic attribute model for dependent attribute. Due to the high number of attribute features in this case, and with the aim to reduce the expensiveness of user similarity computing, we propose also a solution to reduce the dimension based on features selection.

In addition to the formalism used in Section III-B, we denote by F_{A_s} the set of features $f \in F_A$ associated to item s and by S_u the set of items $s \in S$ rated by user u. We define the feature function for item $s \in S$ as $\beta_s : f \in F_A \mapsto 1 if\ f \in F_{A_s}(f\ associated\ to\ item\ s),\ 0\ otherwise$, the item-feature matrix is so equal to $(\beta_s(f))_{s \in S\ and\ f \in F_A}$. We denote also, the rating function of user u as $\delta_u : s \in S_u \mapsto \delta_u(s) \in R$; and the user frequency function as $freq_u : f \in F_A \mapsto freq_u(f) \in \mathbb{N}$. The frequency matrix $F = (freq_u(f))_{u \in U\ and\ f \in F_A}$ is provided by computing $freq_u(f)$ for all users $u \in U$ and all features $f \in F_A$. In the following sections, we propose 4 methods for computing the frequency function $freq_u$.

A. Computing the frequency function for all items

In this case, the frequency function $freq_u(f)$ consists of counting the number of times, the user u rated an item associated to feature f. The formula is given by (1).

$$freq_u(f) = \sum_{s \in S_u} \beta_s(f) \tag{1}$$

B. Computing the weighted frequency function for all items

In (1), all items are treated in a identical way. In this case, we introduce the evaluation of item by user in the frequency function. Thus, the feature function is weighted by the user rating function δ_u. For instance, if $\delta_u(s1) = 4$, $\beta_{s1}(f) = 1$ and $\delta_u(s2) = 2$, $\beta_{s2}(f) = 1$ the frequency of feature f for user u is 5 in this case and 2 in (1).

$$freq_u(f) = \sum_{s \in S_u} \beta_s(f) \times \delta_u(s) \tag{2}$$

C. Computing the frequency function for relevant items

In this case, we consider only relevant items for user u. Let us denote $S_{u_{relevant}} = \left\{ s \in S_u / \ \delta_u(s) \geq \overline{\delta_u} \right\}$ the set of relevant items for user u and $\overline{\delta_u}$ is average rating function of user u over all items in S_u. The use of the average rating function instead of a threshold offers two advantages: first, it avoids adding a new parameter, the threshold value. Secondly, it allows a personalized frequency, because it takes into account the variation between users ratings.

$$freq_u(f) = \sum_{s \in S_{u_{relevant}}} \beta_s(f) \tag{3}$$

D. Computing the weighted frequency function for relevant items

In this case, the feature function is weighted by the rating of only relevant items $s \in S_{u_{relevant}}$.

$$freq_u(f) = \sum_{s \in S_{u_{relevant}}} \beta_s(f) \times \delta_u(s) \tag{4}$$

E. Reduction of dimension of data

For dependent attribute, the number of feature is correlated to the number of items, and so it can be very elevated and even higher than the number of items. Thus, the semantic user attribute model can have dimension greater than the user rating matrix thereby aggravating the scalability problem. In order to reduce the dimension of the user semantic attribute model, we propose a reduction method based on selecting a subset of relevant features from the original set F_A. We choose to reduce the number of features without using an expensive algorithm like LSA or SVD, but by selecting features having a number of ratings greater than a given threshold μ.

So, we define the number of feature ratings function η as:

$$\eta : f \in F_A \mapsto \sum_{u \in U} \sum_{s \in S_u} \beta_s(f)$$

$\eta(f)$ provides the number of ratings associated to f. $F_{A_\mu} = \{ f \in F_A \ / \ \eta(f) \geq \mu \}$ is the set of selected features for attribute A having a number of ratings greater than μ. Thus, only features in F_{A_μ} are used to compute the frequency matrix F.

F. User semantic attribute model

One of the best-known measures for specifying keyword weights in Information Retrieval is the TF-IDF (Term Frequency/Inverse Document Frequency) [14]. It is a numerical statistic, which reflects how important a word is to a document in a corpus. In our case, we replace document by user and term by feature, so we obtain a Feature Frequency Inverse User Frequency(FF-IUF) that is defined as:

$$FF(f,u) = \frac{freq_u(f)}{max_j freq_u(j)} \tag{5}$$

where the maximum is computed over the $freq_u(j)$ of all feature $j \in F_{A_\mu}$ assigned to user u.

The measure of Inverse User Frequency IUF is usually defined as:

$$IUF(f) = \log \frac{|U|}{U_f} \tag{6}$$

where U_f is the number of users assigned to feature f (ie $freq_u(f) \neq 0$). Thus, the FF-IUF weight of feature f for user u is defined as:

$$\omega(u,f) = FF(f,u) \times IUF(f) \tag{7}$$

 ISBN: 978-1-61208-312-4

In summary, for building the user semantic attribute matrix Q_A for a dependent attribute A; first, we define the selected set of features F_{A_μ} to reduce the dimension; second, we compute the frequency matrix F over features $\in F_{A_\mu}$ by using one of the formula from (1) to (4); third, we compute the FF-IUF on this matrix to obtain the matrix Q_A. In Section VI, we will study the performance of each solution. For reasons of clarity, we have called the algorithm using the formula in (1) as FFIUF-NW-NR (no weighted no relevant), in (2) as FFIUF-W-NR (weighted and no relevant), in (3) as FFIUF-NW-R and in (4) as FFIUF-W-R.

V. Recommendation

To compute predictions for the active user u_a, we use the user-based CF algorithm [5]. User-Based CF predicts the rating value of active user u_a on non rated item $s \in S$, it is based on the k-Nearest-Neighbors algorithm. A subset of nearest neighbors of u_a are chosen based on their similarity with him or her, and a weighted aggregate of their ratings is used to generate predictions for u_a. Equation 8 provides formula for computing predictions.

$$p(u_a, s) = \overline{\delta_{u_a}} + L \sum_{v \in V} sim(u_a, v)(\delta_v(s) - \overline{\delta_v}) \qquad (8)$$

where $L = \frac{1}{\sum_{v \in V} |sim(u_a, v)|}$ and V is the set of the nearest neighbors (most similar users) to u_a that have rated item s. V can range anywhere from 1 to the number of all users.

$$sim_(u, v) = \frac{\sum_k (q_{u,k} - \overline{q_u})(q_{v,k} - \overline{q_v})}{\sqrt{\sum_k (q_{u,k} - \overline{q_u})^2} \sqrt{\sum_k (q_{v,k} - \overline{q_v})^2}} \qquad (9)$$

The function $sim(u, v)$ provides the similarity between users u and v and is computed by using the Pearson Correlation (9). In the standard user-based CF algorithm, the users-items rating matrix $(\delta_u(s)_{(u \in U,\, s \in S)})$ is used to compute users' similarities. In our algorithm for computing the similarities between users we use instead the user semantic matrix Q. As we have already mentioned, the matrix Q is the horizontal concatenation of user semantic attribute model of each relevant attribute.

Although we apply a user-based CF for recommendation, our approach is also a model-based method because it is based on a new user model to provide ratings of active user on non rated items. Our approach resolves the scalability problem for several reasons. First, the building process of user semantic model is fully parallelizable (because the computing of user semantic attribute model is done in independent way for each other) and can be done off line. Second, this model allows a dimension reduction since the number of columns in the user semantic model is much lower than those of user item rating matrix, so, the computing of similarities between user is less expensive than in the standard user-based CF. In addition, our approach allows inferring similarity between two users even when they have any co-rated items because the users-semantic matrix has less missing values than user item ratings matrix. Thus, our approach provides solution to the neighbor transitivity problem emanates from the sparse nature of the underlying data sets. In this problem, users with similar preferences may not be identifies as such if they haven't any items rated in common.

VI. Performance Study

In this section, we study the performance of our algorithm, User Semantic Collaborative Filtering (USCF in plots), against the standards CF algorithms: User-Based CF(UBCF), and Item-Based CF(IBCF); standard CB algorithm and an hybrid algorithm. We evaluate these algorithms in terms of predictions accuracy by using the Mean Absolute Error (MAE) [15], which is the most widely used metric in CF research literature. It computes the average of the absolute difference between the predictions and true ratings in the test data set, lower the MAE is, better is the prediction. When comparing results to CB algorithm, we use instead the F1 metric $(2 \times Recall \times Precion/Recall + Precision)$ [15] because CB doesn't provide prediction.

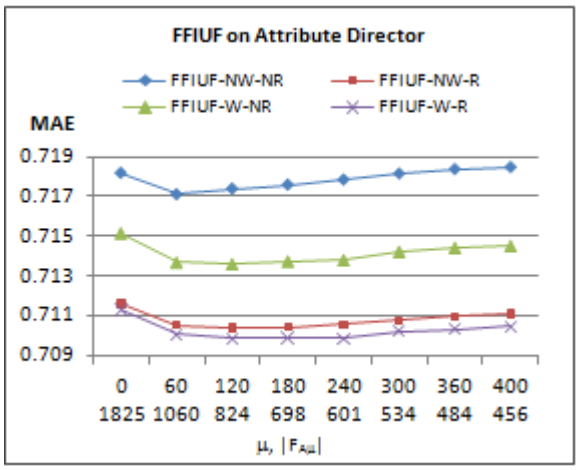
We have experimented our approach on real data from the MovieLens1M dataset of the MovieLens recommender system [16]. The MovieLens1M provides the usage data set and contains 1,000,209 explicit ratings of approximately 3,900 movies made by 6,040 users. For the semantic information of items, we use the HetRec 2011 dataset [17] that links the movies of MovieLens dataset with their corresponding web pages at Internet Movie Database (IMDb) and Rotten Tomatoes movie review systems. We use *movie genre* and *movie origin* as non dependent attributes, *movie director* and *movie actor* as dependent attributes.

We have filtered the data by maintaining only users with at least 20 ratings, and available features for all movies. After the filtering process, we obtain a data set with 6020 users, 3552 movies, 19 genres, 44 origins, 1825 directors and 4237 actors. The usage data set has been sorted by the timestamps, in ascending order, and has been divided into a training set (including the first 80% of all ratings) and a test set (the last 20% of all ratings). Thus, ratings of each user in test set have been assigned after those of training set. It should be noted that the building of user semantic attribute model for the non dependent attributes *genre* and *origin* have been addressed respectively in previous works [3], [4]. Therefore, we will not detail the experiments conducted for these attributes in this paper. If it is not specified, the number of nearest neighbors is equal to 60 because it provides the best results.

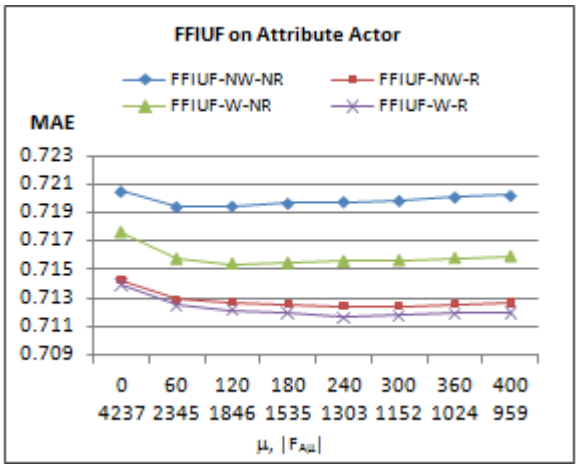
A. Performance Evaluation of the four methods for computing the frequency function

In Figure 1, the MAE has been plotted with respect to the μ threshold parameter (the minimum number of ratings associated to a feature). It compares the 4 algorithms described in Section IV for computing the frequency function, on *director* (Figure 1(a)) and *actor* (Figure 1(b)) attributes. In both case, the plots have the same look, the MAE decreases until a specific value of the number of selected features $|F_{A_\mu}|$ and then grows up; however, FFIUF-W-R algorithm results in an overall improvement in accuracy. We note for both figures, that the reduction of dimension has a slightly effect on improving the accuracy. Indeed, for the director attribute the MAE without reduction (1825 features) is equal to 0.7113 while the best value is equal to 0.7098 obtained by 601 features, so a reduction about 67%. Although the improvement of accuracy isn't elevated, the reduction of dimension is considerable and so allows to reduce the cost of users similarity computing.

 ISBN: 978-1-61208-312-4



(a)



(b)

Fig. 1 Performance evaluation of the four FFIUF algorithms for director (a) and actor (b) attributes.

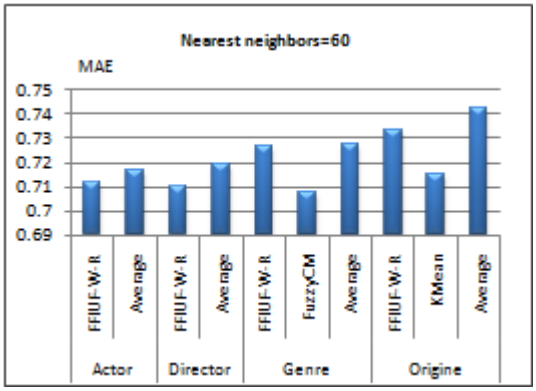


Fig. 2. Impact of user semantic attribute building algorithm on prediction accuracy.

B. Impact of attributes classes on prediction accuracy

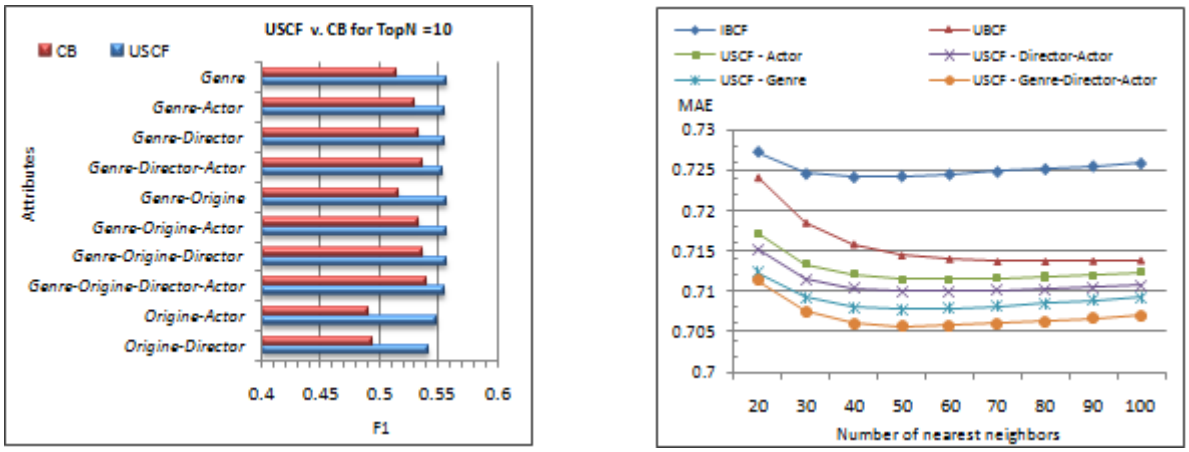
Figure 2 compares algorithms for building user semantic attribute model in term of MAE. The *Average* algorithm (Average in plot) is building user semantic attribute model by computing the average of user ratings by feature ($q_{(u,f)} = AVG\{\delta_u(s)/s \in S_u \; and \; f \in F_{A_s}\}$); *Fuzzy C Mean* algorithm (FuzzyCM in plot) is a fuzzy clustering used for non dependent and multivalued attribute (here *genre*) and *KMean* algorithm (KMean in plot) is used on non dependent and mono valued attribute (here *origin*). When analyzing this figure we note first, that *Average* algorithm provides, for all attributes, the worst performance compared to all other algorithms. Second, if we applied the *FFIUF-W-R* algorithm to non dependent attribute the performance compares unfavorably against the dependent attribute, while the best performance is attained by *FuzzyCM* algorithm on genre attribute and the difference is important (0.7079 for FuzzyCM and 0.7268 for FFIUF-W-R). This allows to deduct that, using a suited algorithm for each attribute class provides best performance than applying the same algorithm for all attributes. Third, the *origin* attribute has the worst performance compared to the other three attributes and this for all algorithms; this is confirm our hypothesis that all attributes don't have the same relevance to users. The attribute *origin* can be less significant in the choice of users than the *genre*, *actor* or *director*, which is intuitively understandable.

C. Comparative results of USCF against CF and CB recommender system

Figure 3 depicts the recommendation accuracy of USCF in contrast to those produced by pure CB (CB in plots) recommender system (Figure 3(a)) using the $F1$ metrics to measure the recommendation accuracy; and standard Item-Based CF (IBCF) and User-Based CF (UBCF) (Figure 3(b)). Pure CB algorithm exploits information derived only from item features. Thus, we create an item-item similarity matrix based on Cosinus similarity applied on item-feature matrix computed on corresponding attribute shown in the plot. In Figure 3(a), recommendations are computed for 60 nearest neighbors. We note that our algorithm USCF results in an overall improvement in accuracy against CB, and this for all combinations of attributes. In Figure 3(b), MAE has been plotted with respect to the number of neighbors (similar users) in the k-nearest-neighbor algorithm. In all cases, the MAE converges between 50 and 60 neighbors, however, USCF results in an overall improvement in accuracy. In addition, the best performance is achieved by the combination *genre-director-actor*. This improvement can be explained by many reasons. First, taking into account the semantic profile of items in a CF recommendation process. Second, for non dependent attribute, user semantic model is built according to a collaborative principle; ratings of all users are used to compute the semantic profile of each user. It is not the case of the Average algorithm; this may explain its results despite taking into account the semantic aspect. Third, the choice of the attribute can have significant influence on improving the accuracy. Lastly, users semantic model Q has few missing values, so, it allows inferring similarity between two given users even when they have any items rated in common.

VII. Conclusion and Future Work

The approach presented in this paper is a component of a global work, which the aim, is to introduce the semantic aspect



(a) (b)

Fig. 3 Evaluation of USCF against CB in terms of F1 (a) against standards CF in terms of MAE (b).

of items in a CF process in order to enhance recommendations and to resolve the scalability problem by reducing the dimension. For this purpose, we have designed a new hybridization technique, which predicts users' preferences for items based on their inferred preferences for semantic information. We have defined two classes of attributes: *dependent* and *non dependent* attribute, and presented a suited algorithm for each class for building user semantic attribute model. The aim of this paper is to present our approach for building user semantic attribute model for dependent attribute. We have defined an algorithm based on the TF-IDF measure and have proposed a solution to reduce a dimension by selecting the most relevant features. Our approach provides solutions to the scalability problem, and alleviates the data sparsity problem by reducing the dimensionality of data. The experimental results show that USCF algorithm improves the prediction accuracy compared to usage only approach (UBCF and IBCF), Content only approach (CB) and hybrid algorithm (Average). In addition, we have shown that applying FFIUF on non dependent attribute, decreases significantly the prediction accuracy compared to results obtained with machine learning algorithms. Furthermore, we have experimentally shown that all attributes don't have the same importance to users. Finally, experiments have shown that the combination of relevant attributes enhances the recommendations.

An interesting area of future work is to use machine learning techniques to infer relevant attributes. We will also further study the extension of the user semantic model to non structured data when items are described by free text. Lastly, study how our approach can provide solution to the cold start problem in which new user has few ratings and CF cannot provide recommendation because similarities with others users cannot be computed.

REFERENCES

[1] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," *Adv. Artificial Intellegence*, 2009.

[2] P. Lops, M. de Gemmis, and G. Semeraro, "Content-based recommender systems: State of the art and trends," in *Recommender Systems Handbook*, 2011, pp. 73–105.

[3] S. Ben Ticha, A. Roussanaly, A. Boyer, and K. Bsaïes, "User semantic preferences for collaborative recommendations," in *13th Int. Conf. on E-Commerce and Web Technologies - EC-Web*. Vienna, Austria: Springer, September 2012, pp. 203–211.

[4] S. Ben Ticha, A. Roussanaly, and A. Boyer, "User Semantic Model for Hybrid Recommender Systems," in *The 1st Int. Conf. on Social Eco-Informatics - SOTICS*. Barcelona, Espagne: IARIA, October 2011.

[5] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "Grouplens: an open architecture for collaborative filtering of netnews," in *The 1994 ACM conf. on Computer supported cooperative work*, Chapel Hill, North Carolina, USA, 1994, pp. 175–186.

[6] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *The 10 Int. WWW Conf.*, Hong Kong, China, May 2001, pp. 285–295.

[7] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 6, pp. 734–749, 2005.

[8] M. Pazzani and D. Billsus, "Content-based recommendation systems," in *The Adaptive Web*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2007, vol. 4321, pp. 325–341.

[9] M. Balabanovic and Y. Shoham, "Fab: Content-based, collaborative recommendation," *Commun. ACM*, vol. 40, no. 3, pp. 66–72, 1997.

[10] R. D. Burke, "Hybrid web recommender systems," in *The Adaptive Web*, ser. Lecture Notes in Computer Science, vol. 4321. Springer, 2007, pp. 377–408.

[11] B. Mobasher, X. Jin, and Y. Zhou, "Semantically enhanced collaborative filtering on the web," in *1st European Web Mining Forum*, vol. 3209, Cavtat-Dubrovnik, Croatia, September 2003, pp. 57–76.

[12] S. Sen, J. Vig, and J. Riedl, "Tagommenders: connecting users to items through tags," in *The 18th Int Conf. on WWW*, Madrid, Spain, April 2009, pp. 671–680.

[13] M. G. Manzato, "Discovering latent factors from movies genres for enhanced recommendation," in *The 6th ACM conf. on Recommender systems - RecSys*, Dublin, Ireland, September 2012, pp. 249–252.

[14] G. Salton, *Automatic Text Processing*. Addison-Wesley, 1989.

[15] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. Riedl, "Evaluating collaborative filtering recommender systems," *ACM Trans. Inf. Syst.*, vol. 22, no. 1, pp. 5–53, 2004.

[16] MovieLens, "http://www.movielens.org," [retrieved: september, 2013].

[17] HetRec2011, in *2nd Int Workshop on Information Heterogeneity and Fusion in Recommender Systems*. The 5th ACM Conf. RecSys, 2011.

 ISBN: 978-1-61208-312-4