Improving Collaborative Filtering's Rating Prediction Coverage in Sparse Datasets through the Introduction of Virtual Near Neighbors

Dionisis Margaris
Department of Informatics and Telecommunications
University of Athens
Athens, Greece
margaris@di.uoa.gr

Dionysios Vasilopoulos, Costas Vassilakis and Dimitris Spiliotopoulos
Department of Informatics and Telecommunications
University of the Peloponnese
Tripoli, Greece
dvasilop@uop.gr, costas@uop.gr, dspiliot@uop.gr

***Abstract*—Collaborative filtering creates personalized recommendations by considering ratings entered by users. Collaborative filtering algorithms initially detect users whose likings are alike, by exploring the similarity between ratings that have insofar been submitted. Users having a high degree of similarity regarding their ratings are termed near neighbors, and in order to formulate a recommendation for a user, her near neighbors' ratings are extracted and form the basis for the recommendation. Collaborative filtering algorithms however exhibit the problem commonly referred to as "gray sheep": this pertains to the case where for some users no near neighbors can be identified, and hence no personalized recommendations can be computed. The "gray sheep" problem is more severe in sparse datasets, i.e. datasets where the number of ratings is small, compared to the number of items and users. In this paper, we address the "gray sheep" problem by introducing the concept of virtual near neighbors and a related algorithm for their creation on the basis of the existing ones. We evaluate the proposed algorithm, which is termed as CF_{VNN}, using eight widely used datasets and considering two correlation metrics which are widely used in Collaborative Filtering research, namely the Pearson Correlation Coefficient and the Cosine Similarity. The results show that the proposed algorithm considerably leverages the capability of a Collaborative Filtering system to compute personalized recommendations in the context of sparse datasets, tackling thus efficiently the "gray sheep" problem. In parallel, the CF_{VNN} algorithm achieves improvements in rating prediction quality, as this is expressed through the Mean Absolute Error and the Root Mean Square Error metrics.**

Keywords—Collaborative Filtering, Sparse Datasets, Virtual Near Neighbors, Pearson Correlation Coefficient, Cosine Similarity, Evaluation.

I. Introduction

Collaborative filtering (CF) creates personalized recommendations, by considering users' likings and tastes, which are expressed as ratings that have been entered by users and are recorded in a ratings database. CF algorithms are distinguished in *user-user* (or *user-based*) and *item-item* (or *item-based*), while some algorithms combine characteristics of both these categories yielding *hybrid* algorithms. In user-user CF, initially users having similar tastes are detected, by comparing the likeness of ratings that are recorded in the ratings database: for each user u, other users whose tastes are very close to those of u are labeled as *nearest neighbors of (NNs) u*. Subsequently, when the algorithm needs to predict the rating $r_{u,i}$ that user u would assign to an item i that has not been rated insofar by u, the algorithm extracts and combines the ratings assigned to item i by u's NNs [1]. This is based on the rationale that users that had similar likings in the past are highly likely to exhibit similar likings in the future, too [2,3]. Item-item CF algorithms follow similar approaches: in this case, ratings are grouped by item, and items that have received similar ratings are located to create the *nearest neighborhood* of each item. Under both approaches, the similarity of ratings related to an entity (user or item) has to be computed; this is calculated using a *correlation coefficient*. A correlation coefficient maps pairs of entities (users or items) to a *similarity metric*, typically falling in the range of [0, 1] or in the range [-1, 1]; in both cases, the lowest value in the range denotes entirely dissimilar entities, while the highest value denoting totally alike ones.

Under this scheme, when computing entity similarities or using an entity's NNs to predict ratings related to that entity, only "real" entities recorded in the user or item database are considered. In this work, we introduce the concept of *virtual NNs*, which are artificially crafted user entities combining the ratings of real users that surpass a similarity threshold and we argue that these artificial user profiles can be used to leverage the recommendation formulation process coverage, i.e. lead to an increased percentage of the users for whom personalized predictions can be computed [2,6,32]. In parallel, the introduction on virtual NNs maintains or slightly improves rating prediction quality.

To illustrate the concept of virtual NNs, let us consider the case where we want to predict the rating $r_{U1,i4}$ that user U_1 would give to item i_4, in order to decide whether it should be recommended to her or not. At that time, the relevant part of the CF ratings database is as shown in Table I.

TABLE I. CF RATING DATABASE EXCERPT

User/Item	i_1	i_2	i_3	i_4	i_5	i_6	i_7
U_1	5	1	3				
U_2				4	5	1	3
U_3	5	1	4		5	1	2

Under the usual practice for computing rating predictions, it is not possible to calculate a correlation (likeness) between users

U_1 and U_2 (since they have rated no items in common), hence the rating $r_{U2,i4}$ cannot be used to formulate a prediction for rating $r_{U1,i4}$. Additionally, while U_1 can be correlated as being "highly similar" with U_3, no rating has been provided by U_3 for item i_4 and, conclusively, no personalized prediction can be computed for rating $r_{U1,i4}$. At this point, some CF-based recommender system implementations will not consider at all item i_4 for recommendation to user U_1, while some other implementations will default to a non-personalized prediction (e.g. the mean of all ratings entered for item i_4), nevertheless such non-personalized predictions are bound to be inaccurate.

However, we can notice in Table 1 that users U_2 and U_3 are highly similar, since, for the items that they have rated in common, all their ratings are either identical or very close. Consequently, it would be possible to merge the ratings of users U_2 and U_3 into a virtual NN profile $VNN_{U2\text{-}U3}$. Now, the virtual profile $VNN_{U2\text{-}U3}$ will include ratings for all items, hence it will be possible to calculate a correlation between U_1 and $VNN_{U2\text{-}U3}$, and ultimately exploit the rating $r_{U2,i4}$ for formulate a prediction for rating $r_{U1,i4}$.

Taking the above into account, in this paper we (1) introduce the concept of VNNs i.e. virtual users, which are created from the combination of real ones and (2) investigate how we can incorporate the VNN user profiles into the rating prediction computation process, so as to leverage the prediction coverage of CF recommender systems (RSs), ensuring in parallel that prediction accuracy is not adversely affected. The incorporation of the VNN aspect into CF leads to a novel algorithm, termed as CF_{VNN}.

To validate our approach, we have conducted an extensive evaluation, assessing (a) the coverage increase gains achieved by the CF_{VNN} algorithm and (b) the impact of the CF_{VNN} algorithm on rating prediction quality. In this evaluation we use eight contemporary data sets from various domains, and we consider two widely used similarity metrics, namely the Pearson Correlation Coefficient (PCC) as well as the Cosine Similarity (CS) [5]. Additionally, we compare the performance of the presented algorithm against the performance of the *negNNs* algorithm, presented in [4], which is a state-of-the-art algorithm also targeting the increase of coverage in the context of CF-based rating predictions. Finally, it is worth noting that the proposed approach can be combined with other algorithms that have been proposed for improving performance, rating prediction accuracy and recommendation quality in CF-based systems, including clustering techniques [7], exploitation of social network (SN) data [8,9,10], pruning of old user ratings [11,12,13] or hybrid filtering algorithms [14]. The rest of the paper is structured as follows: section 2 overviews related work, while section 3 presents the proposed algorithm. Section 4 presents the evaluation procedure and the results obtained and, finally, section 5 concludes the paper and outlines future work.

II. Related Work

The proliferation and widespread use of CF-based systems has led to extensive research on the field. One of the main targets of this research is rating prediction accuracy (e.g. [3,15,16,17]), whereas research on CF-based systems' coverage is relatively limited. [14] introduces "Item HyCov", a filtering algorithm combining the merits of two widely used CF approaches, namely item-based CF and user-based CF, into a hybrid system combining features from both approaches. "Item HyCov" aims at addressing low prediction coverage, an issue mainly appearing in sparse datasets; however, its usefulness in sparse datasets cannot be directly assessed, since results regarding this dataset have only been reported for the "MovieLens100K" dataset [18,19], the density index of which (calculated as $\frac{\#ratings}{\#users * \#items}$) is 6.3%; on the other hand, contemporary datasets exhibit significantly lower density index values, e.g. the density index values of the Amazon datasets [20,22] is smaller by 2-4 orders of magnitude. Additionally, the "Item HyCov" algorithm exhibits higher resource needs, requiring extra storage space and CPU time for the execution of a preprocessing step; furthermore, it introduces the need for continuous updates to maintain the up-to-dateness of both the item neighborhood and the user neighborhood [14]. The concept of user neighborhood is also undertaken in [25], where relationships between users sourced from social networks are used to identify their neighborhoods. In more detail, the work in [25] firstly applies a complex network clustering technique on a graph representing the user SN to identify groups of similar users and; after this step, standard CF algorithms may be employed to compute rating predictions and generate recommendations. While this algorithm can deliver higher rating prediction coverage in sparse datasets, it necessitates the ability to source user relationship data from a SN, and this requirement cannot always be satisfied. [23] adopts an alternative approach, increasing the density of the user-item rating matrix through the calculation of *virtual ratings*, which are computed on the basis of textual reviews entered by users; the computation of *virtual ratings* is performed by identifying opinion words that are present in each review text, subsequently mapping opinion words to sentiments and finally aggregating sentiments into a numeric rating value. An analogous methodology is followed in [24], where the probability that a user will like an item is estimated based on the emotions that are extracted from textual reviews. Both these approaches are successful in increasing coverage, however they can only be applied when textual reviews are available, a condition that is not always met; on the contrary, the algorithm presented in this paper does necessitate any additional information, being able to operate using solely the user-item rating matrix.

Finally, the work in [4] introduces the *negNNs* algorithm which exploits user dissimilarity to leverage rating prediction coverage. This algorithm is based on the rationale that users having exhibited opposite likings in the past are highly likely to exhibit opposing likings in the future, too; henceforth, such "furthest neighbor" users are identified and exploited in the rating prediction process, in order to improve coverage in sparse datasets. The *negNNs* algorithm has been validated to increase coverage between 5.1% and 14.8% across the tested datasets, with an average of 11.6%, while attaining at the same time an average MAE improvement of 0.7%.

The present paper advances the state-of-the-art regarding coverage increase in the context of sparse datasets, by introducing an algorithm that significantly leverages CF coverage, while at the same time achieving to increase CF rating prediction accuracy; this behavior is proven to be consistent under both the PCC and the CS similarity measures, and has been validated using eight contemporary and widely used datasets

III. The Proposed Algorithm

In CF systems, the computation of predictions for a user U are based on U's NNs, i.e. the set of users whose ratings bear a high degree of similarity to those that U has entered for the same items. While a multitude of similarity measures exist, the PCC is the one typically employed. Equation (1) depicts the mathematical formula used to compute the PCC similarity measure between two users U and V:

$$sim\_p(U,V) = \frac{\sum_{k\in I(U)\cap I(V)}\left(r_{U,k} - \overline{r_U}\right) * \left(r_{V,k} - \overline{r_V}\right)}{\sqrt{\sum_{k\in I(U)\cap I(V)}\left(r_{U,k} - \overline{r_u}\right)^2 * \sum_{k\in I(U)\cap I(V)}\left(r_{V,k} - \overline{r_V}\right)^2}} \quad (1)$$

where $I(U)$ and $I(V)$ are the sets of items that have been rated by user U and user V, respectively, while $\overline{r_U}$ and $\overline{r_V}$ are the mean values or ratings entered by users U and V, correspondingly. Subsequently, for each user U her nearest neighbors are selected, among the users U' for which $sim\_p(U, U') > 0$.

Analogously, the Cosine Similarity measure is computed via the mathematical formula shown in Equation (2):

$$sim\_cs(U,V) = \frac{\sum_{k\in I(U)\cap I(V)} r_{U,k} * r_{V,k}}{\sqrt{\sum_{k\in I(U)\cap I(V)}\left(r_{U,k}\right)^2} * \sqrt{\sum_{k\in I(U)\cap I(V)}\left(r_{V,k}\right)^2}} \quad (2)$$

As noted in [22], the CS measure lags behind in terms of rating prediction accuracy when the ratings entered by users have only non-negative values, e.g. have a scale of 1-5 or 1-10 (which is a usual setting in rating systems). In this case, similarity computation process must include a transformation of ratings, which will map ratings below the center of the rating scale to negative values, and ratings above the center of the rating scale to positive values. A typical transformation maps the rating range of $[low, high]$ to the range $[\frac{low-high}{2}, \frac{high-low}{2}]$ in a linear fashion. In this work, we adopt this approach.

When user similarities have been calculated, formula (3) is applied to compute a prediction $p_{U,i}$ for the rating that user U would assign to item i:

$$p_{U,i} = \overline{r_u} + \frac{\sum_{V\in NN_u} sim(U,V) * \left(r_{V,i} - \overline{r_V}\right)}{\sum_{V\in NN_u} sim(U,V)} \quad (3)$$

The proposed algorithm introduces the concept of *VNNs*, according to which a VNN user profile can be created out of two (real) users V and W, who are NNs with each other. This VNN user profile will be denoted as $VNN_{V\text{-}W}$. The $VNN_{V\text{-}W}$ profile will be populated with ratings, which will result from merging the ratings of users V and W: in that respect, the $VNN_{V\text{-}W}$ profile will be more comprehensive than each individual profile of its constituent user profiles V and W, and therefore there is a probability that it can be used to formulate predictions for other users in some cases that the constituent user profiles V and W cannot: indeed, as shown in the example listed in the introduction section, there exist cases where in the context of computing a prediction for the rating $r_{X,k}$ that some user X would assign to item k, the profile of user V does include a rating for item k but cannot be correlated to the profile of X because V and X have not rated any items in common, while the profile of user W can be correlated with that of user X, but does not include a rating for item k. In such cases, the $VNN_{V\text{-}W}$ profile can contribute to the computation of $r_{X,k}$ since it *both* includes a rating for item k *and* it can be correlated with that of user X.

Regarding the computation of the rating values within the profile of VNNs, the following formula applied:

$$r_{VNN_{V-W},k} = \begin{cases} null, & if\ r_{V,k} \notin M \wedge r_{W,k} \notin M \\ \frac{\overline{r_V} + \overline{r_W}}{2} + \left(r_{V,k} - \overline{r_V}\right), & if\ r_{V,k} \in M \wedge r_{W,k} \notin M \\ \frac{\overline{r_V} + \overline{r_W}}{2} + \left(r_{W,k} - \overline{r_W}\right), & if\ r_{V,k} \notin M \wedge r_{W,k} \in M \\ \frac{\overline{r_V} + \overline{r_W}}{2} + \left(r_{V,k} - \overline{r_V}\right) + \left(r_{W,k} - \overline{r_W}\right), & if\ r_{V,k} \in M \wedge r_{W,k} \in M \end{cases} \quad (4)$$

where M is the ratings matrix. This computation formula arranges so that ratings within the $VNN_{V\text{-}W}$ profile are centered around the mean ratings of the two constituent (real) database users, thus being in line with the rationale followed by both the PCC and the CS measures.

When two (real) users V and W are found to have a positive correlation, $VNN_{V\text{-}W}$ is marked as a candidate for instantiation; however, the CF_{VNN} may accommodate additional criteria that are applied to candidate VNN user instantiation, in order to limit the creation of VNN users only to cases where the confidence to the similarity of the constituent users is deemed adequate, so as to avoid adding VNNs that would adversely affect the accuracy of rating predictions. This can be realized by considering the following two aspects:

1. a similarity threshold $Th(sim)$ between users V and W. Under this condition, a virtual user $VNN_{V\text{-}W}$ can be produced out of (real) users V and W only if $sim(V, W) \geq Th(sim)$.
2. a number of common ratings threshold $Th(cr)$, which corresponds to the minimum number of items that users V and W have rated in common [3]. Under this condition, a virtual user $VNN_{V\text{-}W}$ can be produced out of (real) users V and W only if $|I(V) \cap I(W)| \geq Th(cr)$, where $I(V)$ and $I(W)$ denote the set of items rated by users V and W, respectively.

In the next section we explore different options for the values of thresholds $Th(sim)$ and $Th(cr)$, in order to identify the optimal settings for the proposed algorithm's parameters.

IV. Experimental Evaluation

In this section, we report on our experiments aiming to:

1. determine the optimal values for the parameters that are used in the algorithm; these parameters are the thresholds $Th(sim)$and $Th(cr)$; and
2. evaluate the proposed algorithm's performance in terms of coverage and prediction accuracy. This performance is evaluated both against (i) the performance of the plain CF algorithm, which is considered as a baseline, and (ii) the performance of the *negNNs* algorithm introduced in [4].

The *negNNs* algorithm is a state-of-the-art algorithm achieving considerable improvements in terms of coverage, while maintaining (and slightly improving) the quality of rating predictions. Furthermore, the *negNNs* algorithm operates using only the information in the CF ratings database, without necessitating any additional information (e.g. user relationships sourced from SNs).

TABLE II. DATASETS SUMMARY

Dataset name	#Users	#Items	#Ratings	Avg. #Ratings / User	Density	DB size (in text format)
Amazon "Videogames" [20]	8.1K	157K	50K	19.6	0.0039%	3.8MB
Amazon "CDs and Vinyl" [20]	41.2K	1.3M	486K	31.5	0.0065%	32MB
Amazon "Movies and TV" [20]	46.4K	1.3M	134K	29.0	0.0209%	31MB
Amazon "Books" [20]	295K	8.7M	2.33M	29.4	0.0001%	227MB
Amazon "Digital Music" [20]	6.2K	86K	35K	13.9	0.0040%	1.9MB
Amazon "Office Supplies" [20]	3.7K	66K	25K	17.8	0.0714%	1.4MB
Amazon "Grocery and Gourmet Food" [20]	9K	184K	65K	20.4	0.0314%	4.2MB
MovieLens "Latest 100K – Recommended for education and development" [18]	700	100K	9K	143	1.5873%	2.19MB

In order to determine the optimal parameter values, we experimentally explored the parameter value solution space, by iteratively selecting parameter value assignments and examining the effect that the particular parameter value assignments have on the coverage and rating prediction quality. To quantify rating prediction quality, we employed two widely used error metrics, namely the Mean Absolute Error (MAE), and the Root Mean Squared Error (RMSE). The use of two different metrics allows us to gain more detailed insight on the prediction accuracy achieved by each parameter setting, since the MAE metric handles all error scales in a uniform fashion, whereas the RMSE metric penalizes more severely larger errors. To compute the algorithm's coverage, the MAE and the RMSE, we exercised the standard "hide one" technique [3]: each time one rating in the database was hidden and then its value was predicted on the basis of the values of other, non-hidden ratings. This procedure was repeated for every rating in the database. We also performed a second experiment where, for each user, we hid only her last rating, and again predicted its value on the basis of the values of other, non-hidden ratings. The rationale behind the second experiment is that this setting follows closely the operation of a recommender system, where rating values are predicted by considering only ratings that are present at that time point in the database, and not ratings that will be entered in the future (as is the case of the first experiment, where e.g. when hiding the first r_U^1 rating that a user U has entered and predicting its value, the recommender system has access to all ratings that U has entered *after* having submitted r_U^1). The results obtained from the two experiments were in close agreement (the differences observed were less than 2.5% in all cases), therefore for conciseness purposes we report only on the results of the first experiment.

All our experiments were run on eight datasets. Seven of these datasets are relatively sparse and have been obtained from Amazon [20], while the eighth dataset is a relatively dense one and has been sourced from MovieLens [18]. While the CF_{VNN} algorithm mainly targets sparse datasets (since typically dense datasets exhibit adequate coverage and therefore there is little or no need to increase it), we conducted experiments with the (more) dense dataset and present the relevant results, in order to gain insight on the behavior of the CF_{VNN} algorithm in the context of higher density datasets, and confirm that it does neither deteriorate the coverage, nor it adversely affects accuracy metrics. The eight datasets used in our experiments are summarized in Table 1, while they also exhibit the following properties:

1. they are up to date (published between 1996 and 2016),
2. they are widely used as benchmarking datasets in CF research and
3. they differ in regard to the type of item domain of the dataset (videogames, movies, music, books, office products and grocery and gourmet food) and size (ranging from 1.4MB to 227MB in plain text format).

Each dataset was initially preprocessed, and users found to have less than 10 ratings were dropped, since predictions formulated for users with few ratings are known to demonstrate high error levels [2]. This procedure did not have any effect on the MovieLens dataset, since it includes only users that have submitted at least 20 ratings. For our experiments we used a machine equipped with six Intel Xeon E7 - 4830 @ 2.13GHz CPUs, 256GB of RAM and one 900GB HDD with a transfer rate of 200MBps, which hosted the datasets and ran the rating prediction algorithms.

A. Determining the VNN threshold parameters

The goal of the first experiment is to determine the optimal settings regarding the criteria that two NN users, *Y* and *W*, must fulfill in order to produce a VNN user. Note that by virtue of their property of being NNs, *Y* and *W* are known to have at least one rating in common (otherwise no similarity metric could be calculated for them, hence it would not be possible for them to be NNs). Recall from section 3 that the relevant parameters of the CF_{VNN} algorithm explored in this paper are:

1. *Th(sim),* corresponding the similarity threshold that two NNs *Y* and *W* must meet, in order to allow the creation of the $VNN_{Y\text{-}W}$ virtual user ($sim(Y, W) \geq Th(sim)$).
2. *Th(cr)*, indicating the minimum number of commonly rated items that two NNs *Y* and *W* must have, in order to proceed with the creation of the $VNN_{Y\text{-}W}$ virtual user ($|I(V) \cap I(W)| \geq Th(cr)$, where $I(V)$ and $I(W)$ denote the set of items rated by users *V* and *W*, respectively).

Regarding the values of *Th(sim)* we consider only values that are greater than zero, under the rationale that it is only meaningful to include NNs that are positively correlated under the employed similarity metric. Moreover, since it always holds that $|I(V) \cap I(W)| \geq 1$, setting *Th(cr)* to values less than or equal to 1 effectively voids the criterion related to *Th(cr)*.

In order to find the optimal setting for parameters *Th(sim)* and *Th(cr)*, in our first experiment, we explored different combinations of values for these parameters.

In total, more than 25 value combinations were examined, however, in the rest of this paper we report only on the most indicative ones, for conciseness purposes. For each of them, we report the coverage increase achieved and the impact on rating prediction accuracy incurred (rating prediction accuracy is

measured in terms of the MAE and the RMSE metrics, as described in the previous section). Furthermore, the experiments were run for all the datasets listed in Table 1; the results were consistent across all datasets, in the sense that the ranking of parameter value combinations was the same for all datasets, hence in this section we only present the mean values of the respective metrics for all datasets. The results obtained for each individual dataset are discussed in more detail in the next subsection.

Fig. 1 illustrates the coverage increase (bottom half of the chart) and the rating prediction error reduction (upper half of the chart) under different threshold parameter value combinations, when similarity is measured using the PCC similarity metric.

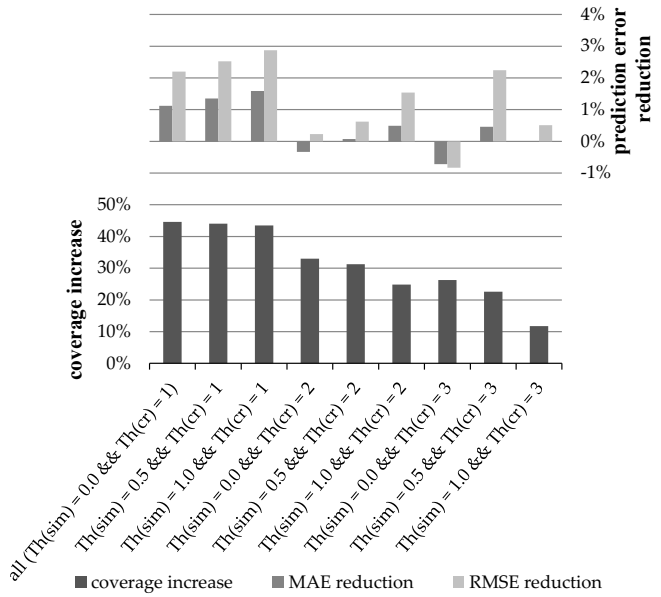

Fig. 1. Coverage increase and prediction error reduction under different threshold parameter value combinations, using the PCC similarity metric

In Fig. 1 we can observe that the three settings in which the common ratings threshold *Th(cr)* is assigned the minimum value (1) are those that deliver the biggest increases in coverage and the highest reductions in prediction errors, for both error quantification metrics (MAE and RMSE). In more detail:

- The setting *Th(sim)*=0.0 &&*Th(cr)*=1 achieves a coverage increase of 44.62%, while the MAE drops by 1.12% and the RMSE is reduced by 2.20%
- The setting *Th(sim)*=0.5 &&*Th(cr)=1* yields a coverage increase of 44.06%, coupled with a decrement of the MAE by 1.35% and an RMSE reduction by 2.52%.
- The setting *Th(sim)*=1.0 &&*Th(cr)=1* leads to an increase of coverage by 43.51%, while the achieved MAE and RMSE decrements are 1.59% and 2.87%, respectively.

When the value of the *Th(cr)* threshold increases to 2 or more, we can observe that the gains reaped for both coverage and prediction error reduction decline, hence these configurations will not be considered further.

Among the three settings where *Th(cr)*=1, we select as optimal the setting where *Th(sim)*=1.0, since the it achieves the highest improvement in terms of prediction accuracy, while also increasing coverage to levels that are comparable with those of the other two settings.

Similarly, when the CS similarity metric is used, the three settings, in which the common ratings threshold *Th(cr)* is assigned the minimum value (1), are those that deliver the biggest increases in coverage and the highest reductions in prediction errors, for both error quantification metrics (MAE and RMSE). Among the three settings, where *Th(cr)*=1, the one having *Th(sim)*=1 achieves the biggest improvements in prediction accuracy in this case as well (the MAE and the RMSE drop by 2.22% and 3.30%, respectively), while attaining a coverage increase equal to 37.88%, which is comparable to that achieved by the other two settings where *Th(cr)*=1. The results of this experiment are listed in detail in [21].

Taking the above results into account, in the rest of this paper we adopt the setting where *Th(sim)*=1.0 and *Th(cr)*=1 and present in more detail the individual dataset results obtained in the experiments where the CF$_{VNN}$ parameters were set to those exact values. We note at this point that the results obtained from the experiments with the two other settings where *Th(cr)*=1 (i.e. the settings in which *Th(sim)*=0.0 and *Th(sim)*=0.5) follow the same pattern as the one exhibited in Fig. 1, i.e. both settings achieve a slightly superior coverage increase compared to the setting where *Th(sim)*=1.0, while the improvements in terms of prediction accuracy observed for these settings are inferior to those of the setting where *Th(sim)*=1.0.

B. Performance evaluation

After having determined the optimal parameters for the operation of the CF$_{VNN}$ algorithm (i.e. the values for the Th(sim) and Th(cr) thresholds), we proceed in presenting in detail the algorithm's performance metrics for each of the datasets listed in Table 1. In the following, we initially report on the results obtained from our experiments on the seven sparse datasets listed in Table 1, since the proposed algorithm targets this dataset category. The results obtained from the dense dataset (MovieLens "Latest 100K") are discussed separately, so as to gain insight on the effect of the algorithm mainly on the prediction accuracy, since for dense datasets the coverage is already at high levels.

Besides presenting absolute performance metrics regarding improvements in coverage and accuracy achieved by the CF$_{VNN}$ algorithm, we compare its performance with the performance of the *negNNs* algorithm introduced in [4]. The *negNNs* algorithm is a recently published state-of-the-art algorithm targeting the increase of CF coverage, necessitating no additional information (e.g. user relationships sourced from SNs) and achieving considerable improvements in coverage while maintaining (and slightly improving) the quality of rating predictions.

Fig. 2 depicts the measurements obtained regarding the increase in coverage, when user similarity is quantified using the PCC measure. We can notice that the average coverage increase attained by the CF$_{VNN}$ algorithm over all datasets is equal to 43.51%, exceeding by 3.38 times the performance of the *negNNs* algorithm, which achieves an average increment equal to 12.86%.

At the level of individual datasets, the performance of the CF$_{VNN}$ algorithm exceeds that of the *negNNs* one [4] by a factor ranging from 2.6 for the "Amazon Movies & TV" dataset to 4.7 times higher, observed for the "Amazon Grocery and Gourmet

Food" dataset. Interestingly, the "Amazon movies and TV" dataset, where the proposed algorithm achieves its lowest increase, has the highest (#ratings / #items) ratio among the seven sparse datasets. Although this behavior is not consistent across all datasets, i.e. it is not concurred that lower (#ratings / #items) ratios in a dataset lead necessarily to lower increases in coverage achieved by the CF_{VNN} algorithm, this observation is notable and will be further studied in our future work.

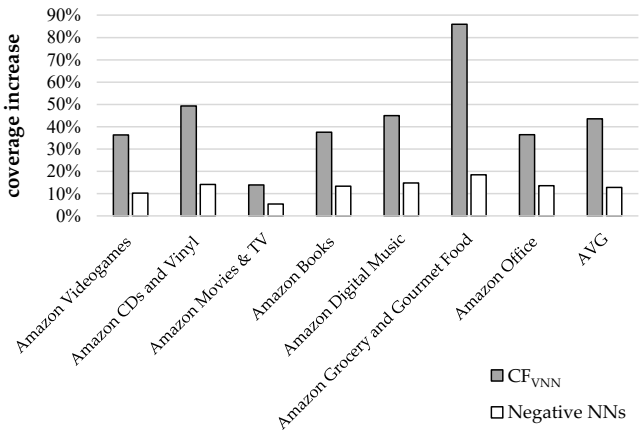

Fig. 2. Coverage increase for the different datasets, under the PCC user similarity metric

Interestingly, the "Amazon movies and TV" dataset, where the proposed algorithm achieves its lowest increase, has the highest (#ratings / #items) ratio among the seven sparse datasets. Although this behavior is not consistent across all datasets, i.e. it is not concurred that lower (#ratings / #items) ratios in a dataset lead necessarily to lower increases in coverage achieved by the CF_{VNN} algorithm, this observation is notable and will be further studied in our future work.

Although this behavior is not consistent across all datasets, i.e. it is not concurred that lower (#ratings / #items) ratios in a dataset lead necessarily to lower increases in coverage achieved by the CF_{VNN} algorithm, this observation is notable and will be further studied in our future work.

In the case that the user similarity is computed using the CS measure, the gains introduced by the CF_{VNN} algorithm regarding coverage increase are 36.62% on average over all datasets, ranging from 7.41% (observed for the "Amazon movies and TV" dataset) to 77.02% (observed for the "Amazon Grocery and Gourmet Food" dataset). These improvements surpass the corresponding ones achieved by the negNNs algorithm [4] (8.64%) by approximately 4.23 times. The results of this experiment are listed in detail in [21].

Fig. 3 depicts the measurements obtained regarding the reduction of the MAE when user similarity is quantified using the PCC measure. In Fig. 3 we can observe that the average MAE reduction achieved over all datasets is equal to 1.59%, which is approximately 2.3 times higher than the corresponding MAE drop attained by the *negNNs* algorithm [4] (0.69%). When considering the algorithms' performance on individual datasets, the MAE reductions achieved by the CF_{VNN} algorithm surpass those attained by the *negNNs* one [4], by a factor that ranges from 1.66(for the "Amazon Digital Music" dataset) to 5.13(observed for the "Amazon Books" dataset).

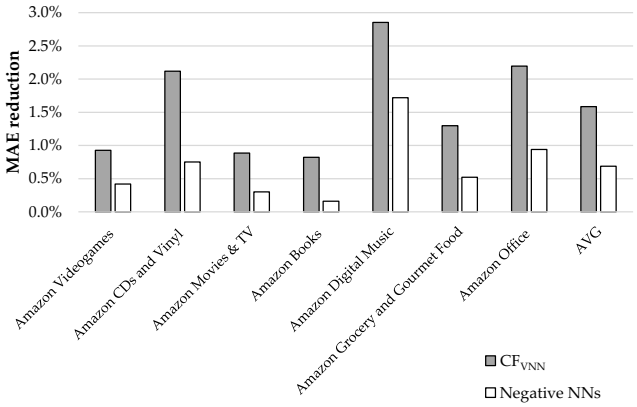

Fig. 3. MAE reduction for the different datasets, under the PCC user similarity metric

When using the CS metric, the CF_{VNN} algorithm attains an average MAE reduction over all datasets equal to 2.22%; the smallest improvement is 0.47% (achieved for the "Amazon Videogames" dataset), while the largest one is 4.74% (for the "Amazon Digital Music"). On average, the improvements achieved by the CF_{VNN} algorithm regarding the MAE exceed those of the negNNs algorithm (1.29%) by 1.72 times. The MAE reduction achieved by the CF_{VNN} algorithm, in this case, is 39.6% higher than the corresponding reduction attained by the same algorithm under the PCC similarity metric. The results of this experiment are listed in detail in [21].

Finally, Fig. 4 demonstrates the measurements obtained regarding the reduction of the RMSE, when user similarity is quantified using the PCC measure.

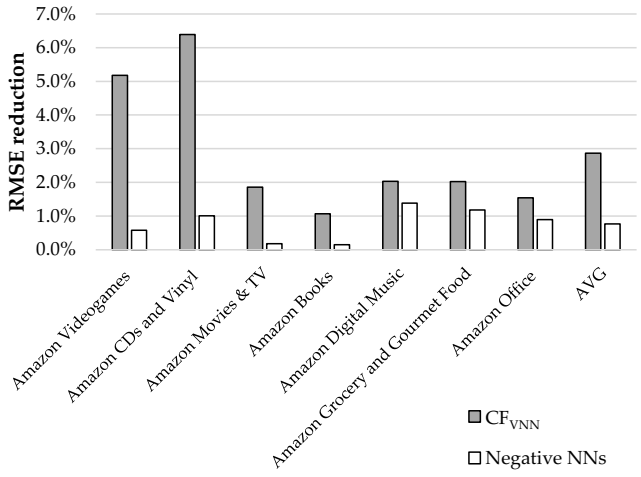

Fig. 4. RMSE reduction for the different datasets, under the PCC user similarity metric

We can observe that the CF_{VNN} algorithm reduces on average across all datasets the RMSE by 2.87%, exceeding the respective improvement attained by the negNNs algorithm [4] (which is equal to 0.77%) by approximately 3.7 times. When the performance at individual dataset level is considered, the performance edge of the CF_{VNN} over the negNNs one [4] ranges from 1.5 times higher for the "Amazon Digital Music" dataset to 10.3 times higher, observed for the "Amazon Movies & TV" dataset.

In the case that the user similarity is computed using the CS measure, the average RMSE reduction over all datasets achieved by the CF$_{VNN}$ algorithm is equal to 3.30%, ranging from 1.44% (for the "Amazon office" dataset) to 5.57% (for the "Amazon Digital Music"). The average RMSE reduction attained by the CF$_{VNN}$ algorithm exceeds that of the *negNNs* algorithm [4] (1.20%) by 2.75 times. As in the case when PCC is employed as a similarity measure, the RMSE reduction achieved by the CF$_{VNN}$ is higher than the MAE reduction, indicating that the algorithm remedies some errors with high absolute magnitudes. The results of this experiment are listed in detail in [21].

In all datasets and under both similarity metrics, we can notice that the improvement achieved for the RMSE metric surpasses the corresponding improvement in the MAE. This shows that the CF$_{VNN}$ algorithm manages to remedy some prediction errors with high absolute magnitudes, since the RMSE metric is known to "punish" high errors more severely, contrary to the MAE metric where all errors are taken into account with equal weight, regardless of their magnitude.

C. The MovieLens "Latest 100K – Recommended for education and development" Dataset

In this subsection we present the results regarding the application of the CF$_{VNN}$ algorithm on the MovieLens "Latest 100K" dataset. This dataset has a considerably higher density than the other seven datasets used in this paper: more specifically, its density index (calculated as $\frac{\#ratings}{\#users* \#items}$) is 1.59%, surpassing the corresponding index of the other seven datasets from 22 to 16,000 times. Due the high density of this dataset, the coverage attained by the plain CF algorithm is 94.04%, when user similarity is computed using the PCC similarity measure, hence the coverage improvement margins are severely restricted. According to the results presented in [4], the negNNs algorithm achieves a coverage increase equal to 0.45%; on the other hand, the CF$_{VNN}$ algorithm introduced in this paper increases coverage by 1.9%, exceeding the performance of the negNNs algorithm by approximately 4.2 times.

Regarding the accuracy of rating prediction, the negNNs algorithm achieves a MAE drop equal to 0.27%, while the corresponding RMSE drop is equal to 0.35%; the respective MAE and RMSE drops achieved by the CF$_{VNN}$ algorithm are equal to 0.47% and 0.56%, respectively, surpassing the performance of the negNNs algorithm.

In the case that user similarity is computed using the CS measure, the coverage of the plain CF algorithm for this dataset is equal to 95.68%. Under this setting, both the CF$_{VNN}$ and the negNNs algorithms exhibit equivalent performance regarding coverage increase, achieving to leverage coverage by 1.2%. As far as rating prediction accuracy is concerned, the CF$_{VNN}$ algorithm reduces the MAE and the RMSE by 0.4% and 1.7%, respectively, surpassing the performance of the negNNs algorithm, which does not improve or deteriorate the MAE (i.e. its MAE is equal to that of the plain CF algorithm), while reducing the RMSE by 1.3%.

We can notice that in the context of dense datasets, the CF$_{VNN}$ algorithm achieves a small coverage increase, while it also delivers a considerable improvement in rating prediction accuracy. In comparison to the negNNs algorithm [4], the CF$_{VNN}$ algorithm is found again to have a performance edge.

V. Conclusion And Future Work

In this paper we have introduced a novel CF algorithm, namely CF$_{VNN}$, which targets the improvement of rating prediction coverage in sparse datasets. The CF$_{VNN}$ algorithm is based on the concept of VNNs, which are artificial user profiles created by merging pairs of near neighbor profiles corresponding to real users; under this scheme the VNN profiles are more comprehensive and thus can be correlated with a larger number of real user profiles in comparison to their constituent user profiles, aiding thus to the alleviation of the "gray sheep" problem.

The proposed algorithm has been validated through experiments in which two broadly used similarity measures (PCC and CS) and eight contemporary datasets have been employed. The experiment results have shown that in the context of sparse datasets, the CF$_{VNN}$ algorithm delivers a significant increase in coverage, ranging from 13.93% to 85.96%, with an average of 43.5%, under the PCC similarity measure, while under the CS similarity measure coverage gains range from 7.41% to 77.02%, with an average of 36.62%. In parallel, the CF$_{VNN}$ algorithm delivers notable improvements regarding rating prediction accuracy: under the PCC similarity measure, the CF$_{VNN}$ algorithm reduces the MAE by 1.59% and the RMSE by 2.87% on average across all datasets, while under the CS similarity measure the respective drops are 2.22% and 3.30%.

We have also compared the performance of the proposed algorithm against that of the negNNs algorithm [4], which is a state-of-the-art algorithm targeting coverage increase in sparse datasets and being able to operate using only the information within the CF ratings database. The CF$_{VNN}$ algorithm, presented in this paper, has been found to consistently outperform the negNNs algorithm, both regarding coverage increase and rating prediction accuracy improvement. Finally, we have evaluated the performance of the proposed algorithm in the context of dense datasets; in this context, the CF$_{VNN}$ algorithm has been found to deliver small improvements in coverage and rating prediction accuracy.

These results indicate that the CF$_{VNN}$ algorithm can be used in any dataset, regardless of its density, and therefore CF system administrators/practitioners may integrate the proposed algorithm into running systems without needing to examine the characteristics of the used dataset, given that the CF$_{VNN}$ algorithm will increase both the coverage and rating prediction accuracy of the CF system.

The encouraging results from the CF$_{VNN}$ algorithm experiments create further research opportunities including (a) studying the behavior of the algorithm under more similarity measures, such as the Euclidian distance, the Spearman coefficient and the Manhattan distance [5], (b) analyzing in more depth how different dataset characteristics affect the performance of the algorithm, (c) adapting the CF$_{VNN}$ algorithm for use in combination with matrix factorization [27], since matrix factorization techniques exhibit high potential and (d) investigating the combination of the CF$_{VNN}$ algorithm with other techniques, such as concept drift detection [30,31,33], exploitation

of social network [8,29,34,35,36] or Internet of Things data [28,26], in order to further improve recommendation quality.

Our future work will include the above directions, while we will also investigate the formulation and validation of new techniques for increasing coverage and/or leveraging rating prediction accuracy in sparse CF datasets.

VI. References

[1] M Balabanovic and Y. Shoham, "Fab: content-based, collaborative recommendation," Communications of the ACM, vol 40(3), pp. 66-72, 1997.

[2] M. Ekstrand,. R. Riedl J. Konstan, "Collaborative Filtering Recommender Systems," Foundations and Trends in Human-Computer Interaction, vol. 4(2), pp. 81-173, 2011.

[3] K.Yu, A.Schwaighofer, V.Tresp, X. Xu and H.P. Kriegel, "Probabilistic Memory-Based Collaborative Filtering," IEEE Transactions on Knowledge Data Engineering, vol. 16(1), 56-69, 2004.

[4] D. Margaris and C. Vassilakis, "Improving Collaborative Filtering's Rating Prediction Coverage in Sparse Datasets by Exploiting User Dissimilarity," Proceedings of the 4th IEEE International Conference on Big Data Intelligence and Computing, pp. 1054-1059, 2018.

[5] J.Herlocker, J. Konstan, L. Terveen and J. Riedl, "Evaluating collaborative filtering recommender systems," ACM Transactions in Information Systems, vol. 22(1), pp. 5-53, 2004.

[6] D. Margaris, C. Vassilakis and P. Georgiadis, "An integrated frameworkfor QoS-based adaptation and exception resolution in WS-BPEL scenarios", Proceedings of the 28th ACM Symposium on Applied Computing, pp. 1900-1906, 2013

[7] D.Margaris, P. Georgiadis and C. Vassilakis, "A Collaborative Filtering Algorithm with Clustering for Personalized Web Service Selection in Business Processes," Proceedings of the 9th IEEE International Conference on Research Challenges in Information Science, pp. 169-180, 2015.

[8] E. Bakshy, D. Eckles, R. Yan and I. Rosenn, "Social Influence in Social Advertising: Evidence from Field Experiments," Proceedings of the 13th ACM Conference on Electronic Commerce, pp. 146-161, 2012.

[9] D. Margaris, C. Vassilakis and P. Georgiadis, "Recommendation information diffusion in social networks considering user influence and semantics," Social Network Analysis and Mining, vol. 6(1), 108, pp. 1-22, 2016.

[10] D.Margaris, C. Vassilakis and P. Georgiadis, "Knowledge-Based Leisure Time Recommendations in Social Networks," Current Trends on Knowledge-Based Systems: Theory and Applications, pp. 23-48, 2017.

[11] D. Margaris and C. Vassilakis, "Pruning and aging for user histories in collaborative filtering," Proceedings of the 2016 IEEE Symposium Series on Computational Intelligence, pp. 1-8, 2016.

[12] D. Margaris and C. Vassilakis, "Enhancing User Rating Database Consistency through Pruning," Transactions on Large-Scale Data and Knowledge-Centered Systems, vol. XXXIV, pp. 33–64, 2017.

[13] D. Margaris and C. Vassilakis, "Improving Collaborative Filtering's Rating Prediction Quality in Dense Datasets, by Pruning Old Ratings," Proceedings of the 22nd IEEE Symposium on Computers and Communications, pp. 1168-1174, 2017.

[14] M. Vozalis, A. Markos and K. Margaritis, "A Hybrid Approach for Improving Prediction Coverage of Collaborative Filtering," Artificial Intelligence Applications and Innovations, vol. 296, pp. 491–498, 2009.

[15] R. Dias and M. Fonseca, "Improving Music Recommendation in Session-Based Collaborative Filtering by Using Temporal Context," Proceedings of the 25th IEEE International Conference on Tools with Artificial Intelligence, pp. 783-788, 2013.

[16] D. Margaris, C. Vassilakis and P. Georgiadis, "Query personalization using social network information and collaborative filtering techniques," Future Generation Computer Systems, vol. 78(1), pp. 440-450, 2018.

[17] D. Margaris and C. Vassilakis, "Improving Collaborative Filtering's Rating Prediction Accuracy by Considering Users' Rating Variability," Proceedings of the 4th IEEE International Conference on Big Data Intelligence and Computing, pp. 1022-1027, 2018.

[18] MovieLens datasets. Available online: http://grouplens.org/datasets/movielens/ (accessed on April 4, 2019).

[19] F. Harper and J. Konstan, "The MovieLens Datasets: History and Context," ACM Transactions on Interactive Intelligent Systems, vol. 5(4), Article no. 19, 2016.

[20] Amazon product data. Available online: http://jmcauley.ucsd.edu/data/amazon/links.html (accessed on April 4, 2019).

[21] D. Margaris, D. Vassilopoulos, C. Vassilakis and D. Spiliotopoulos, "Experimental results for considering Virtual Near Neighbors in Collaborative Filtering's Rating Prediction," SDBS Lab Technical report TR-19001, https://soda.dit.uop.gr/?q=TR-19001 (accessed on April 15, 2019).

[22] J.J. McAuley, C. Targett, Q. Shi and A. Van den Hengel, "Image-Based Recommendations on Styles and Substitutes," Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 43-52, 2015.

[23] D. Poirier, F. Fessant and I. Tellier, "Reducing the cold-start problem in content recommendation through opinion classification," Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, pp. 204-207, 2015.

[24] Y. Moshfeghi, B. Piwowarski, and J. Jose, "Handling data sparsity in collaborative filtering using emotion and semantic based features," Proceedings of 34th ACM SIGIR Conference, pp. 625-634, 2011.

[25] M. Pham, Y. Cao, R. Klamma and M. Jarke, "A Clustering Approach for Collaborative Filtering Recommendation Using Social Network Analysis," Journal of Universal Computer Science, vol. 17(4), 583-604, 2011.

[26] M. Munoz-Organero, G. Ramirez, P. Merino, and C. Kloos, "A collaborative recommender system based on space-time similarities for an Internet of Things," IEEE Pervasive Computing, vol. 9(3), pp. 81–87, 2010.

[27] Y. Koren, R. Bell and C. Volinsky, "Matrix factorization techniques for recommender systems," Computer, vol. 42(8), pp. 30-37, 2009.

[28] D. Margaris and C. Vassilakis, "Exploiting Internet of Things Information to Enhance Venues' Recommendation Accuracy," Service Oriented Computing & Applications, vol. 11(4), pp. 393-409, 2017.

[29] D. Margaris and C. Vassilakis, "Exploiting Rating Abstention Intervals for Addressing Concept Drift in Social Network Recommender Systems," Informatics, vol. 5(2), Article no. 21, 2018.

[30] D.Margaris and C. Vassilakis, "Enhancing Rating Prediction Quality through Improving the Accuracy of Detection of Shifts in Rating Practices," Transactions on Large-Scale Data- and Knowledge-Centered Systems, vol. XXXVII, pp. 151-19, 2018.

[31] D. Margaris and C. Vassilakis, "Using Time Clusters for Following Users' Shifts in Rating Practices," Complex Systems Informatics and Modeling Quarterly, vol. 75(13), pp. 22-42, 2017.

[32] D. Margaris, C. Vassilakis and P. Georgiadis, "Adapting WS-BPEL scenario execution using collaborative filtering techniques," Proceedings of the 7th IEEE International Conference on Research Challenges in Information Science, pp. 174-184, 2013.

[33] D. Margaris and C. Vassilakis, "Improving Collaborative Filtering's Rating Prediction Quality by Considering Shifts in Rating Practices," Proceedings of the 19th IEEE International Conference on Business Informatics, pp. 158-166, 2017.

[34] D. Antonakaki, D. Spiliotopoulos, C.V. Samaras, S. Ioannidis and P. Fragopoulou, "Investigating the Complete Corpus of Referendum and Elections Tweets," Proceedings of the IEEE/ACM Conference on Advances in Social Networks Analysis and Mining, pp. 100-105, 2016.

[35] G. Schefbeck, D. Spiliotopoulos and T. Risse, "The Recent Challenge in Web Archiving: Archiving the Social Web," Proceedings of the International Council on Archives Congress, pp. 20-24, 2012

[36] D. Antonakaki, D. Spiliotopoulos, C.V Samaras, P. Pratikakis, S. Ioannidis and P. Fragopoulou, "Social media analysis during political turbulence," PloS one, vol. 12(10), pp. 1-23, 2017.