University of the Peloponnese
Department of Informatics and Telecommunications
Software and Database Systems Laboratory
https://soda.dit.uop.gr

Experimental results for considering Virtual Near Neighbors in Collaborative Filtering's Rating Prediction

Technical Report TR-19001

Dionisis Margaris, Dionysios Vasilopoulos, Costas Vassilakis, Dimitris Spiliotopoulos
margaris@di.uoa.gr, dvasilop@uop.gr, costas@uop.gr, dspiliot@uop.gr

March, 2019
Tripoli, Greece

1. Introduction

In this technical report, we present the experimental findings from applying an algorithm that considers virtual near neighbors (VNNs) in the rating prediction formulation process, in order to increase coverage in the context of sparse datasets.

To this end, the algorithm is applied to seven sparse datasets, which are widely used in recommender system research. Additionally, the algorithm is applied to one dense dataset, in order to gain insight on the performance of the proposed algorithm in this class of datasets, as well.

In short, the algorithm introduces the concept of VNNs i.e. virtual users, which are created from the combination of real ones, in order to be used as candidate NNs in the rating prediction computation process.

In these experiments, the optimal values for the parameters that are used in the algorithm are investigated and more specifically, the thresholds that two individual users can constitute a VNN.

2. Experiment results

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).

To compute the algorithm's coverage, the Mean Absolute Error (MAE) and the Root Mean Squared Error (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.

The datasets used in the experiment are summarized in Table I, and the results obtained are listed in the following subsections. In the results presentation subsections, cells with a gray background indicate cases where the rating prediction accuracy of the proposed algorithm surpasses that of the plain CF algorithm, while cells with bold typeface indicate that the respective cell corresponds to the optimal performance (rating prediction accuracy or coverage) achieved.

TABLE I. DATASETS SUMMARY

Dataset name	#Users	#Items	#Ratings	Avg. #Ratings / User	Density	DB size (in text format)
Amazon "Videogames" [1]	8.1K	157K	50K	19.6	0.0039%	3.8MB
Amazon "CDs and Vinyl" [1]	41.2K	1.3M	486K	31.5	0.0065%	32MB
Amazon "Movies and TV" [1]	46.4K	1.3M	134K	29.0	0.0209%	31MB
Amazon "Books" [1]	295K	8.7M	2.33M	29.4	0.0001%	227MB
Amazon "Digital Music" [1]	6.2K	86K	35K	13.9	0.0040%	1.9MB
Amazon "Office Supplies" [1]	3.7K	66K	25K	17.8	0.0714%	1.4MB
Amazon "Grocery and Gourmet Food" [1]	9K	184K	65K	20.4	0.0314%	4.2MB
MovieLens "Latest 100K – Recommended for education and development" [1]	700	100K	9K	143	1.5873%	2.19MB

2.1 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). The relevant parameters of the CF_{VNN} algorithm explored in this paper are:

- *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)*).
- *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.

Figure 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.

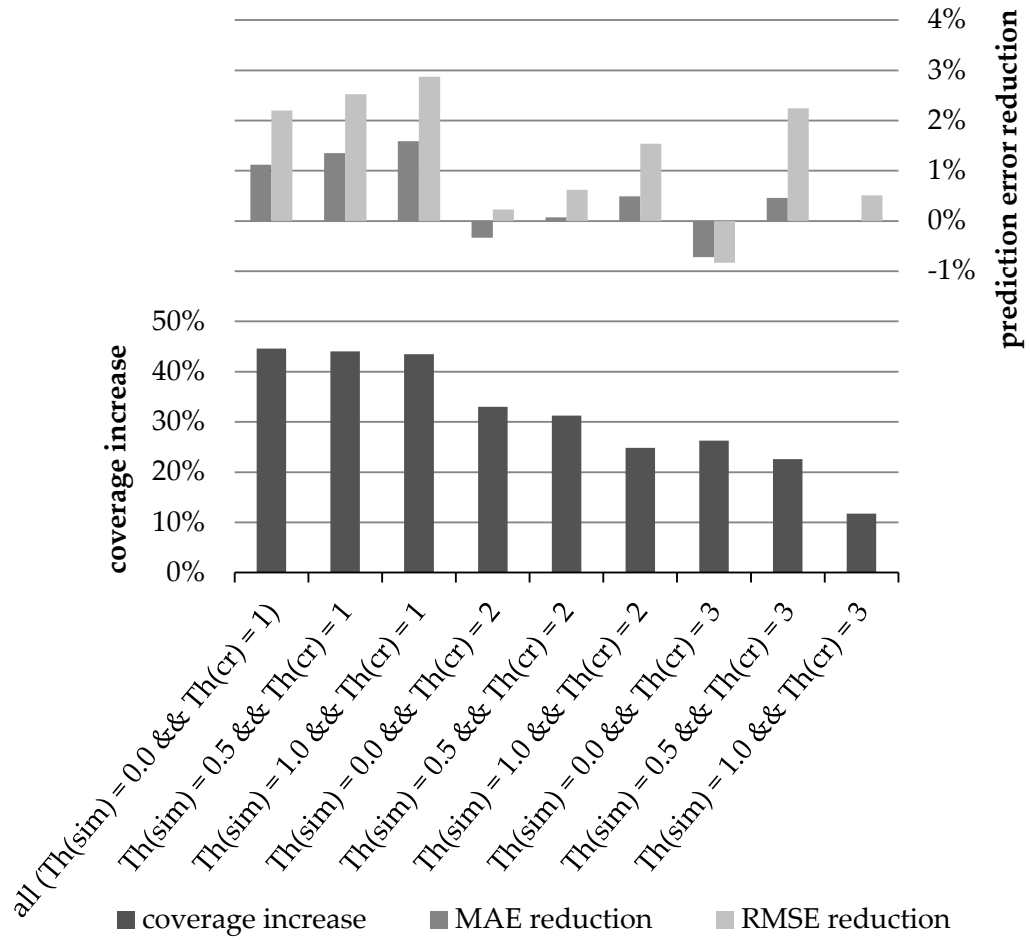

Figure 1. Coverage increase and prediction error reduction under different threshold parameter value combinations, using the PCC similarity metric

In Figure 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, Figure 2 depicts the coverage increase (bottom half of the chart) and the rating prediction error reduction (upper half of the chart) under different threshold parameters values, when using the CS similarity metric.

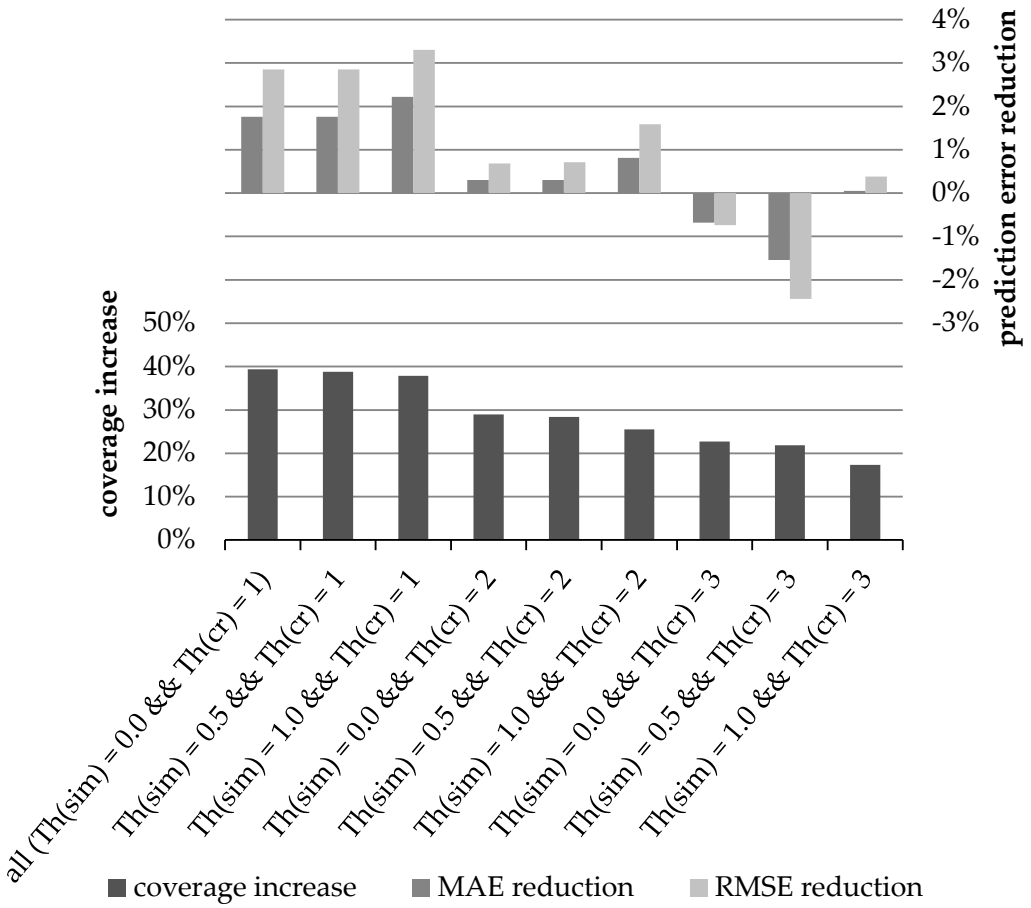

Figure 2. Coverage increase and prediction error reduction under different threshold parameters values, using the CS similarity metric

Again, 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). Among the three settings where *Th(cr)*=1, the one having *Th(sim)*=1 achieves the biggest improvements in prediction accuracy in this case too (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.

Taking the above results into account, in the rest of this paper we adopt the settingwhere *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 Figures 1 and 2, 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.

2.2 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.

2.1.1 Experiments using the Pearson Correlation Coefficient as a similarity metric

Figure 3 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%.

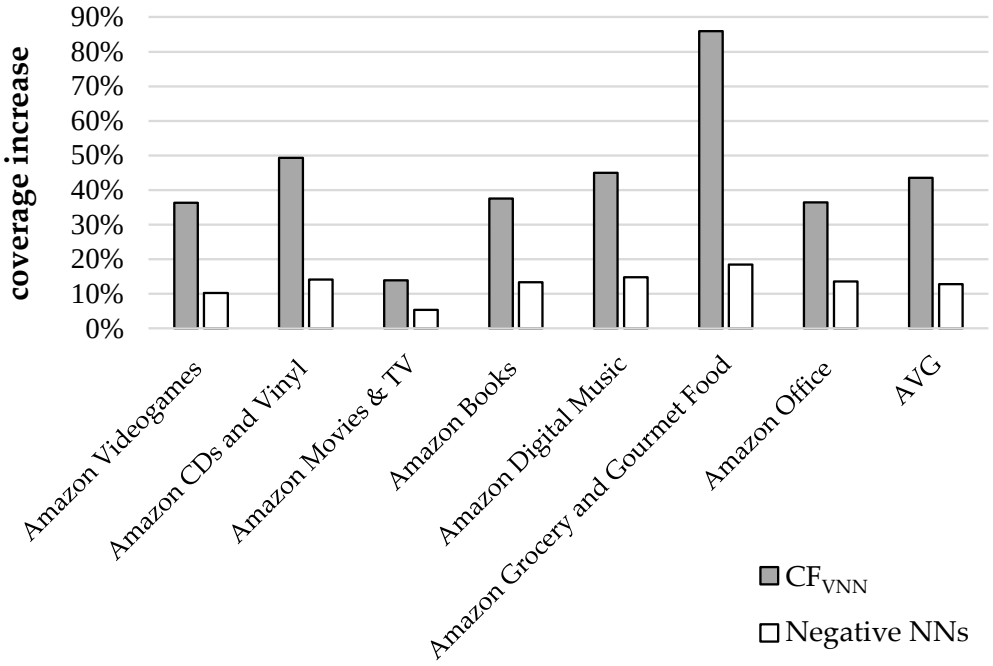

Figure 3. Coverage increase for the different datasets, under the PCC user similarity metric

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.

Figure 4 depicts the measurements obtained regarding the reduction of the MAE when user similarity is quantified using the PCC measure.

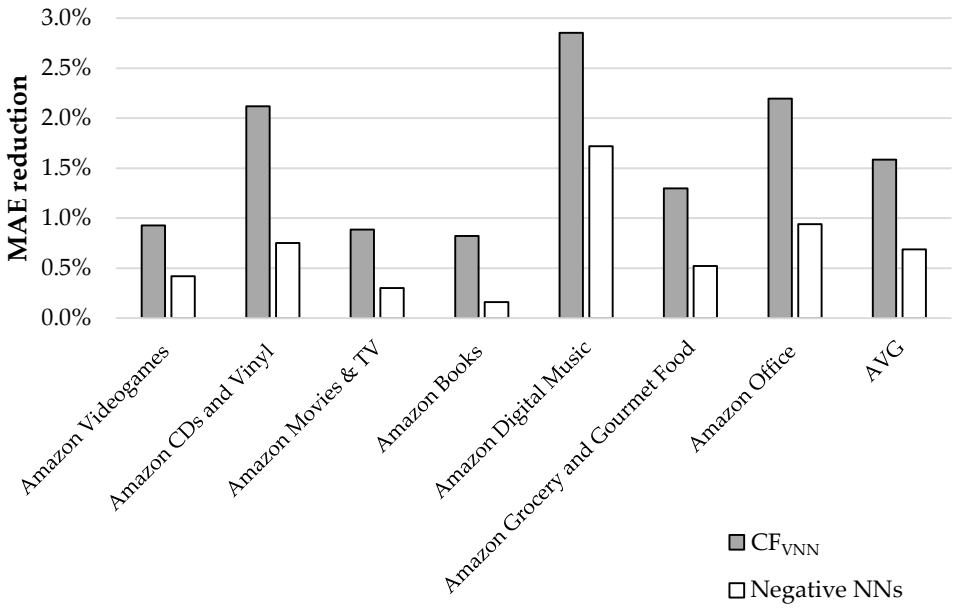

Figure 4. MAE reduction for the different datasets, under the PCC user similarity metric

In Figure 4 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).

Finally, Figure 5 demonstrates the measurements obtained regarding the reduction of the RMSE, when user similarity is quantified using the PCC measure.

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.

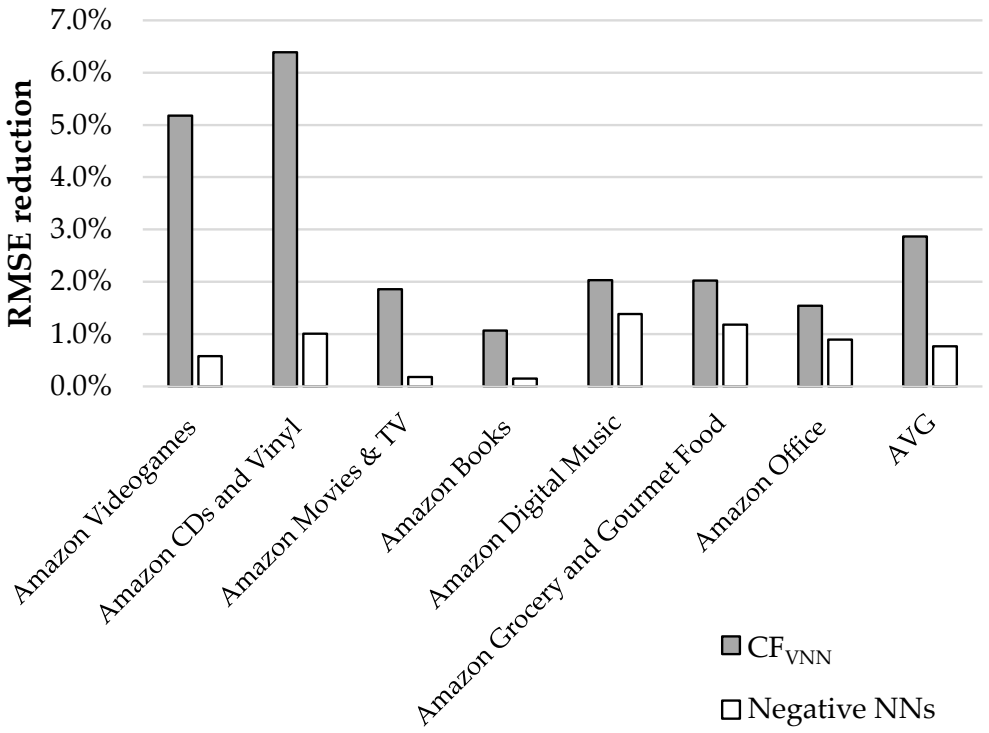

Figure 5. RMSE reduction for the different datasets, when using the PCC as a similarity metric

In all datasets we can notice that the improvement achieved for the RMSE metric surpasses the corresponding improvement in the MAE. This shows that the CFVNN 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.

2.1.2 Experiments using the Cosine Similarity as a similarity metric

Figure 6 depicts the measurements obtained regarding the increase in coverage, when user similarity is quantified using the CS measure. We can notice that 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.

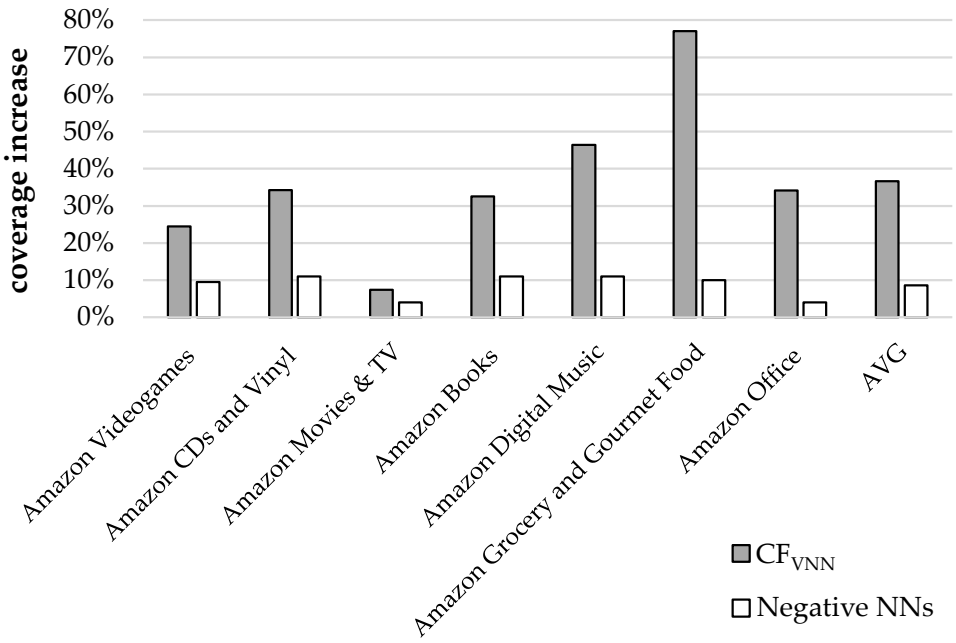

Figure 6. Coverage increase for the different datasets, under the CS user similarity metric

At the level of individual datasets, the CF_{VNN} algorithm outperforms the negNNs one [4]by a factor ranging from 1.85(for the "Amazon Movies & TV" dataset) to 7.7(observed for the "Amazon Grocery and Gourmet Food" dataset). The coverage increase achieved by the CF_{VNN} algorithm under the CS similarity metric is 6.89% lower than the corresponding reduction attained by the same algorithm under the PCC similarity metric.

Figure 7 depicts the measurements obtained regarding the MAE reduction when similarity between users is measured using the CS metric.

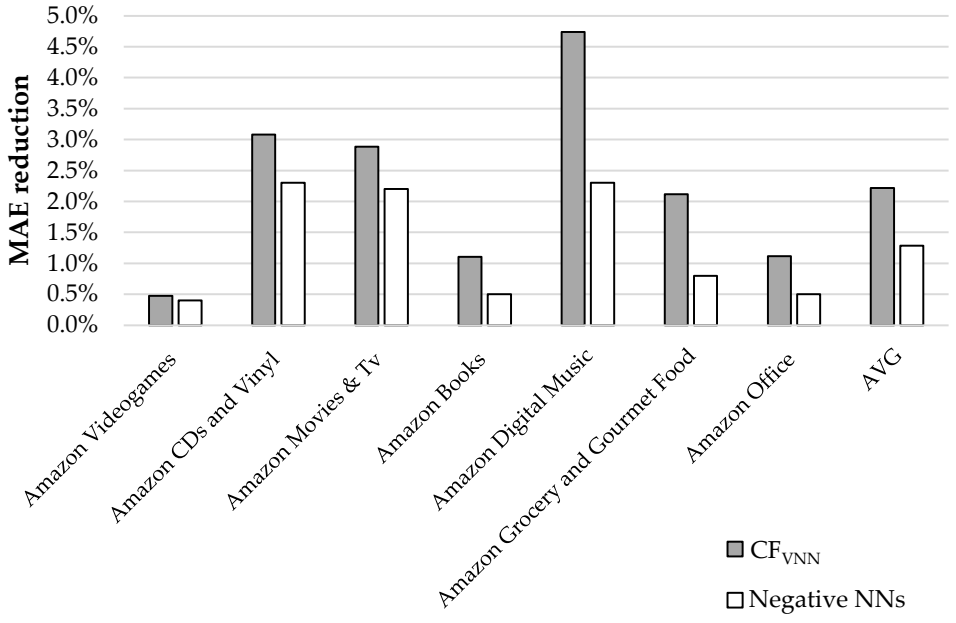

Figure 7. MAE reduction for the different datasets, under the CS user similarity metric

We can notice that 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 [4] (1.29%) by 1.72 times. The MAE reduction achieved by the CF_{VNN} algorithm under the CS similarity metric is 39.6% higher than the corresponding reduction attained by the same algorithm under the PCC similarity metric.

Finally, Figure 8 illustrates the measurements obtained regarding the reduction of the RMSE, when similarity between users is measured using the CS metric.

We can notice that 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. Similarly to 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.

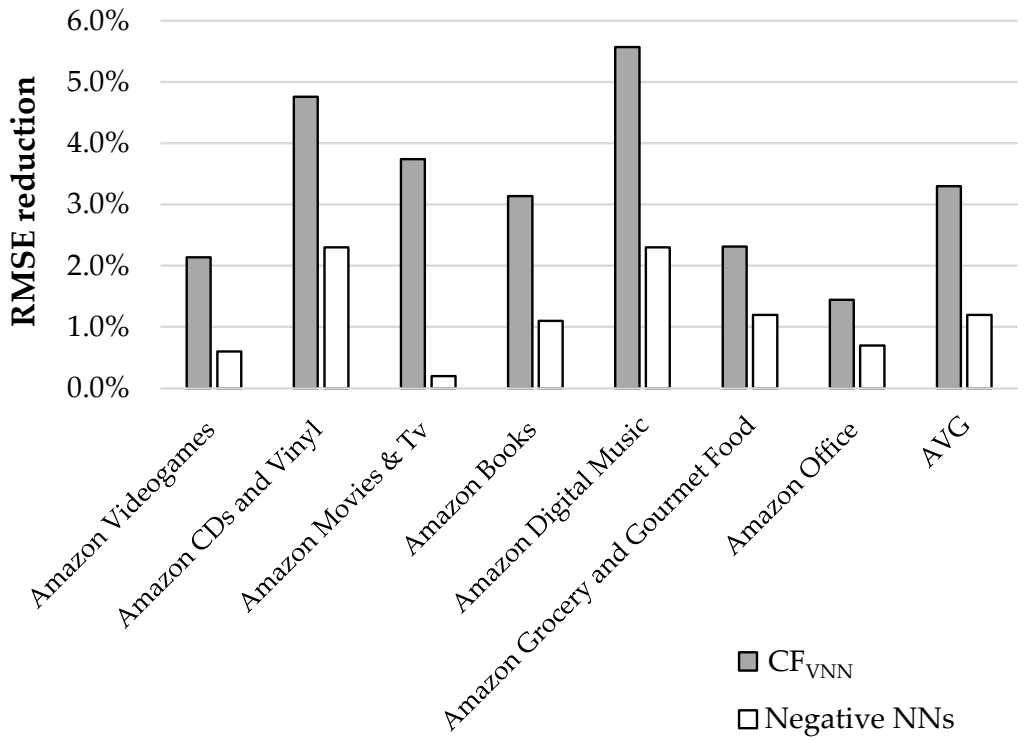

Figure 8. RMSE reduction for the different datasets, under the CS user similarity metric

2.1.3 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 dropis 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.

3. Conclusions

In this report we have presented the experimental findings from applying an algorithm that considers virtual near neighbors in the rating prediction formulation process in order to increase coverage in the context of sparse datasets. The results indicate that the above algorithm achieves to increase coverage, while slightly improving rating prediction accuracy. In the context of dense datasets, coverage increase ranges from nonexistent to very small, while rating prediction quality can be slightly improved.

4. References

[1] Amazon product data. Available online: http://jmcauley.ucsd.edu/data/amazon/links.html (accessed on 11 January 2019).

[2] MovieLens datasets. Available online: http://grouplens.org/datasets_/movielens/ (accessed on 26 January 2019).

[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.