Rate it Again: Increasing Recommendation Accuracy by User re-Rating

Xavier Amatriain
Telefonica Research
xar@tid.es

Josep M. Pujol
Telefonica Research
jmps@tid.es

Nava Tintarev
Telefonica Research
nava@tid.es

Nuria Oliver
Telefonica Research
nuriao@tid.es

ABSTRACT

A common approach to designing Recommender Systems (RS) consists of asking users to explicitly rate items in order to collect feedback about their preferences. However, users have been shown to be inconsistent and to introduce a non-negligible amount of *natural noise* in their ratings that affects the accuracy of the predictions. In this paper, we present a novel approach to improve RS accuracy by reducing the natural noise in the input data via a preprocessing step. In order to quantitatively understand the impact of natural noise, we first analyze the response of common recommendation algorithms to this noise. Next, we propose a novel algorithm to denoise existing datasets by means of re-rating: *i.e.* by asking users to rate previously rated items again. This denoising step yields very significant accuracy improvements. However, re-rating all items in the original dataset is unpractical. Therefore, we study the accuracy gains obtained when re-rating only some of the ratings. In particular, we propose two partial denoising strategies: data and user-dependent denoising. Finally, we compare the value of adding a rating of an unseen item *vs.* re-rating an item. We conclude with a proposal for RS to improve the quality of their user data and hence their accuracy: asking users to re-rate items might, in some circumstances, be more beneficial than asking users to rate unseen items.

1. INTRODUCTION AND MOTIVATION

Recommender Systems (RS) [1] are becoming so ubiquitous in our lives that popular authors have even ventured to say that "we are leaving the age of information and entering the age of recommendation" [3]. The goal of RS is to model the users' tastes (preferences) in order to suggest (recommend) unseen content that the users would find of interest.

User preferences can be captured *implicitly*, by observing user actions or consumption patterns [12]; or *explicitly*, by asking users to rate certain items. Although explicit feedback adds a burden on the users, and different users might respond differently to incentives [9], it is commonly used because it respects user integrity, allows for greater transparency, and is more reliable than implicit data in many cases. RS based on explicit user feedback are built under the assumption that user ratings constitute the ground truth about the users' likes and dislikes. However, little attention has been paid to how consistent users are in giving these ratings. Great efforts are being invested in finding better algorithms that can reduce the Root Mean Square Error (RMSE) in predicting user ratings by 10% [5]. However, it might well be that similar improvements could be obtained by improving the quality of the input data and applying off-the-shelf algorithms.

In previous work [2], we reported on a user study designed to capture user inconsistencies when giving explicit feedback in the form of movie ratings. Our goal was to characterize the user input (*natural*) noise and to analyze the impact that this noise has on the so-called *magic barrier* in RS [10] – *i.e.* the asymptotic limit on accuracy improvements for all recommendation techniques, regardless of how fine-tuned they are. The present work takes these previous results as the starting point, so we summarize them in the first part of Section 2.

The focus of this paper, though, is on novel approaches to improve the accuracy of standard Collaborative Filtering (CF) algorithms by removing some of the natural noise present in the original rating data. We first analyze the behavior of common CF algorithms under different natural noise conditions in the second part of Section 2. Next, we propose a novel algorithm to remove natural noise by means of *re-rating* – *i.e.* by asking users to rate again previously rated items (Section 3). However, in a practical situation it might not be feasible to have all users re-rate all items. Therefore, we discuss strategies to select the optimal ratings to be re-rated, in order to improve the algorithm's accuracy with minimal intervention (Section 4). This analysis raises the question of the relative value of re-rating versus adding ratings for new items, which we discuss in Section 5. Finally, we describe related work in Section 6, and present our conclusions and highlight our lines of future research in Section 7.

In sum, the main contributions of this paper are: (1) A study on how common CF algorithms behave under a variety of natural noise conditions; (2) A novel algorithm to denoise explicit rating datasets based on re-rating; (3) The proposal of strategies to identify the best items to re-rate in order to accomplish effective partial; denoising and (4) A discussion on the impact that selective re-rating *vs.* rating of new items would have on the performance of RS

2. NATURAL NOISE IN USER RATINGS

The results presented in this paper build upon previous work in analyzing user inconsistencies and characterizing user input (natural) noise in RS via explicit ratings [2]. Therefore, we shall summarize next the findings of that work that are relevant to our current discussion.

Our user experiment consisted on 3 trials of the same task: rating 100 movies selected from the Netflix Prize database [5] via a Web

	$\#T_i$	$\#T_j$	$\#$		$RMSE$	
			$\cap$	$\cup$	$\cap$	$\cup$
T_1, T_2	2185	1961	1838	2308	0.573	0.707
T_1, T_3	2185	1909	1774	2320	0.637	0.765
T_2, T_3	1969	1909	1730	2140	0.557	0.694

Table 1: Summary of results on the pairwise comparison between trials. Columns 1 and 2 contain the number of ratings in trials T_i and T_j. Columns 3-4 depict the number of elements in the intersection and the union for trials T_i and T_j. The last two columns report the noise measured as the RMSE of the intersection and the union sets respectively.

interface, and included 118 participants. The three trials took place at different points in time in order to assess the reliability of the user rating paradigm and to measure the variability of users. The minimum time difference between the first and second trial was 24 hours while the minimum time difference between the second and third trial was 15 days. The first and third trial presented items in the same random order while the second trial presented items in order of popularity. User ratings were provided on a 1 to 5 star scale, with the additional option (an icon) of specifying that they had not seen or were unable to rate the movie. We compared the Netflix dataset with ours in terms of the (a) rating distribution and (b) number of ratings per user, and concluded that our experimental dataset is comparable to the Netflix dataset with respect to both variables.

We define the aggregated rating of user u's ratings of movie m as a tuple $\langle r_k \rangle_{um}$, where r_k corresponds to the rating in trial T_k. Therefore, for a given user u and movie m we have vector of three ratings $\langle r_{um1}, r_{um2}, r_{um3} \rangle$. Note that there are user $\times$ movies tuples (*i.e.* $118 \times 100 = 11800$ in our case). A rating is considered to be *consistent* across trials when all values of r_k are the same. Note that we exclude form our analysis the tuples where all r_k are zeros, which is the value used to represent a *not-seen*.

In order to analyze the effect that the not seen value has in our study, we considered two different subsets: a) the *intersection*, or only tuples where all ratings are *seen* (> 0); and b) the *union*, where not seen values are included. For instance, the tuple for two trials with ratings $\langle 4, 5 \rangle_{um}$ would be considered inconsistent, because user u changed the evaluation of movie m from 4 to 5 in the last trial. This tuple would be included both in the intersection and the union set. However, tuple $\langle 4, 0 \rangle_{um}$ would not be included in the intersection set, because one of the ratings is a *not-seen*.

Table 1 summarizes the results of the experiment when grouping the trials by pairs, where T_k corresponds to trial k, $k = 1, ..., 3$. For example, in T_1, users provided 2185 out of the potential 11800 ratings. Thus, 9615 positions in the rating matrix of T_1 are *not-seen* values. The differences in the number of ratings illustrate how users were not even able to consistently determine whether they have seen a movie or not. Out of the 2185 items that users rated in the first trial only 1682 (77%) were consistently rated again in trials $T2$ and $T3$.

The right side of the Table contains the RMSE for the intersection and union sets across all trials. Note that this RMSE is not the error by a particular RS algorithm but rather the direct measure of the error between any two trials [1]. The RMSE ranges between 0.56 and 0.64 for the intersection sets and 0.69 and 0.77 for the union, depending on the trials. RMSE is lower for any pair in which T_2 is involved. Therefore, we concluded that T_2, where items were presented to the user in order of popularity, is the trial with the least amount of natural noise [2].

[1]Computed as the square root of the square differences between ratings in the two trials. In the union set, "not seen" values are replaced by the user average.

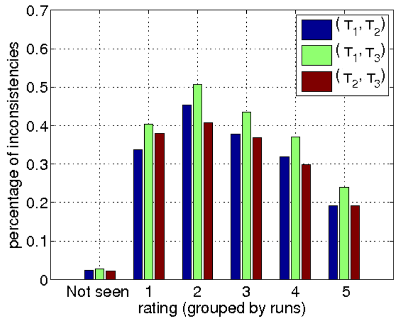


Figure 1: Percentage of user inconsistencies by rating value

Figure 1 shows the probability of inconsistency by the value of the rating between pairwise trials (T_1,T_2), (T_2,T_3) and (T_1,T_3). In other words, the probability that a rating with a value of X in trial T_i will be different in trial T_j. Note how ratings with extreme opinions (*i.e.* the lowest and highest ratings in the scale) are *more consistent* (*i.e.,* less noisy) across different trials: the probability of inconsistencies is highest for 2 and 3 star ratings. Also note that the distribution is not symmetric since the probability of inconsistencies is higher in the lower end of the scale (1 and 2 ratings) than in the positive ratings (4 and 5). The average ratings in our study are 2.73, 2.79 and 2.79 for T_1, T_2 and T_3, respectively. Finally, note that the probability of inconsistency with *not-seen* is significantly lower than any of the rating values.

2.1 Algorithm Robustness to Natural Noise

We shall now analyze the robustness of three common RS algorithms to natural noise and compare their ability to predict datasets with varying levels of natural noise. For comparison purposes, we also include two baselines – user mean and item (movie) mean, although such strategies would not be used in a practical scenario. The three algorithms are amongst the most commonly used in collaborative filtering settings: *k*-Nearest Neighbors or *k*NN (item and user based) [4] and Singular Value Decomposition (SVD) [14].

In order to test the algorithms, we carry out a 100-fold repeated random sampling, selecting 90% of our data as *training* and 10% as *testing* – *i.e.* 10% of unknown ratings are predicted by each of the algorithms, given the other 90%. Our dataset consists of few users (118) and items (100), with an average of 18.5 ratings per user. Therefore, each testing subset might yield significant differences in the final error measure. Hence, we use a large number of cross-validation runs and a 90/10 split – instead of the common 80/20, in order to ensure consistent and fair results for all algorithms. This same validation procedure will be used throughout the paper.

2.1.1 Trial RMSE

First, we analyze the performance and robustness of the algorithms to natural noise by applying each of the algorithms to each of trial datasets (T_1, T_2, and T_3).

Table 2 summarizes the RMSE of each of the five algorithms with the datasets for each trial. Rather than focusing on the RMSE values for each algorithm – which depend on the parameter settings and nature of the dataset, we are interested in comparing their robustness to natural noise. As shown in the table, all algorithms perform best on the second trial (T_2). This is the trial that was found to contain the least amount of natural noise, as previously explained. However, the relative performance ranking of the algorithms remains the same regardless of the dataset –*i.e.* user-based *k*NN first, SVD second, and item-based *k*NN third. There are no

Dataset	T_1	T_2	T_3	$T_{best}\Delta T_{worst}$
User Average	1.2011	**1.1469**	1.1945	4.7%
Item Average	1.0555	**1.0361**	1.0776	4%
User-based *k*NN	0.9990	**0.9640**	1.0171	5.5%
Item-based *k*NN	1.0429	**1.0031**	1.0417	4%
SVD	1.0244	**0.9861**	1.0285	4.3%

Table 2: RMSE of the algorithms in predicting each trial dataset. The best predicted dataset appears in bold. The last column reports improvement in % for less noisy trial (T_2) as compared to noisiest (either T_1 or T_3, depending on the algorithm)

Training/Testing Dataset	$T_1 \blacktriangleleft T_2$	$T_1 \blacktriangleleft T_3$	$T_2 \blacktriangleleft T_3$
User Average	**1.1585**	1.2095	1.2036
Movie Average	**1.0305**	1.0648	1.0637
User-based *k*NN	**0.9693**	1.0143	1.0184
Item-based *k*NN	**1.0009**	1.0406	1.0590
SVD	**0.9741**	1.0491	1.0118

Table 3: RMSE of the algorithms when predicting one dataset in terms of another one ($T_i \blacktriangleleft T_j$ means predicting T_j using T_i for training). Best result for each algorithm in bold.

significant differences in improvement between the best and worst predicted dataset: column 5 ranges from 4% to 5.5%. Note that the larger this difference, the more affected the algorithm is by natural noise.

2.1.2 *Prediction RMSE*

In our previous analysis, we predicted ratings within a single session – *i.e.* the training and test datasets have the same time stamp. A more realistic setup consists of predicting future ratings from past ones. With this rationale in mind, we now measure the RMSE when predicting a future trial by using a past one. We use a similar validation procedure as in our previous experiment. However, the 10% test set is now taken from the target trial, which is different – and later in time – than the training set.

The results are summarized in Table 3. A first observation is that the error values are similar to the errors obtained when doing intra-trial prediction for all algorithms. The worst results are obtained when predicting T_3 (last two columns). Note that T_3 is the dataset with the largest time gap from the rest (at least 15 days from T_2 and 16 days from T_1). However, the error obtained when predicting T_3 from T_1 or T_2 is comparable to the error obtained when trying to predict T_3 from its own data (see third column in Tables 2 and 3). As in the previous experiment, the differences in behavior between algorithms are not significant.

The best results are obtained when predicting T_2 (least noisy dataset) from T_1. However as summarized in Table 1, the number of user inconsistencies between T_1 and T_2 was not the minimum among all trials – which corresponded to T_1 and T_3 (last column in the Table). Therefore, we shall posit that the algorithms are *more sensitive* to the amount of noise in the target (test) dataset than the amount of noise in the training dataset. This hypothesis is confirmed when predicting T_1 from T_2, where T_1 is noisier than T_2[2]. In this case, the RMSE values are notably higher than when predicting T_2 from T_1: 1.02019 (*vs.* 0.9693), 1.05338 (*vs.* 1.0009), and 1.02778 (*vs.* 0.9741) for user-based *k*NN, item-based *k*NN and SVD respectively.

This finding suggests that RMSE should be applied with care when used as a success measure for RS: the performance of the algorithms will be affected by errors and noise in the test dataset, such that their true performance might be higher than measured, due to the noisy "ground truth". Our findings also suggest that the performance of all algorithms is affected by the noisiness in the rating dataset: a change in the order with which items were presented to the users to be rated (*e.g.*, in T_1 *vs.* T_2) might improve the algorithm's performance up to a 5.5%, as illustrated in the last column of Table 2. These findings serve as motivation for the work presented in the rest of this paper, where we propose an approach to reduce existing natural noise in order to increase the prediction accuracy of the algorithm. Since we have already seen that there is no significant difference between algorithms, results in the next sections will be reported only for two representative algorithms: user-based *k*NN and SVD.

[2]Note that this experiment is done for illustration purposes, since in a practical situation it does not make sense to predict a past rating from a future one.

3. DENOISING VIA RE-RATING

In the previous section, we have shown the impact that natural input noise has on the RMSE of RS algorithms and hypothesized that denoising the input dataset might improve performance. We explore now whether having multiple repetitions of a user rating (re-rating) can be exploited to remove some of the natural noise in the original dataset and hence obtain better predictions.

In particular, we propose a novel denoising algorithm aimed at removing input noise while retaining maximum information in the dataset. Hence, we impose two *fairness* conditions to be met by the algorithm:

- The algorithm shall produce a denoised version of the input dataset that has as many ratings as possible (*i.e.* it should delete as few ratings as possible from the original dataset)
- The denoised dataset should always include one of the ratings provided by the user in one of the trials (*i.e.* the algorithm should not create new values, but rather decide on which of the existing input ratings to trust)

Algorithm 1 Denoising ($\mathcal{D}$) a t-source re-Rated item

```
Input: $R_{um} = \{r_0, ..., r_t\}$ – set of ratings of user $u$ on movie $m$ (original rating plus $t$ re-ratings)
Input: $\gamma$ – threshold of noise
Output: $r$ – de-noised rating
function $\mathcal{D} : (R_{um} = \{r_0, ..., r_t\}, \gamma) \rightarrow r$
$S = \{\mathcal{M}(\{a, b\})\} \mid a \in R_{um} \wedge b \in R_{um} \wedge a \neq b \wedge |a-b| \leq \gamma$
– Create a set $S$ composed of the mildest value of all possible pairs of ratings in $R_{um}$ s.t. $|a-b| \leq \gamma$
if $\exists a, b \in S$ s.t. $|a-b| > \gamma$ then
    return $\mathcal{D}(S, \gamma)$ – apply recursively if there is at least one SD
else
    return $\mathcal{M}(S)$
end if
```

Given these two conditions, we formalize the Algorithm 1. The algorithm receives the set of $t+1$ ratings that a user has given to a given item – *i.e.* original rating plus t re-ratings – and returns a single denoised rating r. The denoised rating is obtained by recursively removing all ratings that belong to a rating pair (a, b) whose distance is larger than γ, and by replacing the pairs whose distance is less or equal than γ by the mildest one.

The algorithm can be understood by analyzing its output in each of the three possible conditions. If there is *agreement* (A) between all ratings belonging to a given user and movie, the algorithm returns that rating. If all ratings differ by more than a given threshold

Denoised⊚Denoising	$T_1 \circledcirc T_2$	ΔT_1	$T_1 \circledcirc T_3$	ΔT_1	$T2 \circledcirc T3$	ΔT_2
User-based *k*NN	0.8861	**11.3%**	0.8960	10.3%	0.8984	6.8%
SVD	0.9121	**11.0%**	0.9274	9.5%	0.9159	7.1%

Table 4: RMSE of the algorithms when predicting a dataset that has been denoised using rating values from another dataset ($T_i \circledcirc T_j$ means T_i has been denoised using T_j). Also included relative improvement over RMSE in the original (noisy) dataset

γ, there is *strong disagreement* (*SD*) and the rating is removed from the original dataset. Note that in this particular case we are removing existing ratings since there is too much inconsistency. We must ensure that this situation happens only rarely not to conflict with the first fairness condition that we imposed.

Finally, if at least two ratings differ by less than γ, there is a *mild disagreement* condition (*MD*). The strategy to follow in this case is not obvious. In the proposed algorithm, the $\mathcal{M}$ function in Equation 1 is used to select the *mildest* rating. After empirical tests, this was the strategy that yielded the best performance with our dataset. The rationale behind the choice of the *mildest rating* is the following: Given a user profile, milder ratings are the least informative and therefore they will have a small influence when deciding whether to recommend an item or not. In addition, they introduce the least amount of risk in terms of possible errors. Hence, whenever a disagreement is detected, the rating that minimizes the risk of error is chosen. Although this particular way of treating mild disagreements should generalize well to other datasets, other strategies, such as using the average of the conflicting ratings, might prove better in datasets with different conditions. However, they yielded worse accuracy results in our case.

In order to obtain the mildest rating of a set, we define the function $\mathcal{M}$ applied to a set of ratings R_{um} that returns the rating r that is closest in value to the average of the rating scale. Formally,

$$\mathcal{M}:\ R_{um} = \{r_1, ..., r_t\} \rightarrow r$$
$$r \in R_{um}\ s.t.\ |r-\sigma| \leq |r'-\sigma|\ \ \forall r' \in R_{um} \qquad (1)$$

where $\sigma = \frac{1}{M}\sum_j r_j^*$ and R^* is the numerical rating scale of choice (*e.g.* $R^* = \{1,2,3,4,5\}$). Note that the mildest rating function applied to an empty set returns a non-determined value, $\mathcal{M}(\emptyset) = IND$.

Evaluation procedure.

Again, we use a similar experimental procedure as before with denoised ratings for both the test and training sets as our aim is to predict a clear (denoised) set of preferences rather than to predict the noise inherent in the original ratings. It is clear that in applying the denoising algorithm, the mean user rating may move closer to the middle of the scale, reducing the RMSE. We argue however, that it does so rightfully: recall that we have not invented any new ratings, and only deleted ratings – less than 10% in any case – when there was severe disagreement (i.e. user feedback could not be trusted).

One-source re-rating.

Although Algorithm 1 can be applied with t re-rates, it is not realistic to expect having items that are re-rated many times by the same user. Therefore, a case of particular interest is that in which users re-rate items only once (one-source re-rating). Algorithm 2 includes the pseudocode to instantiate Algorithm 1 to the case of one-source re-rating.

The previous one-source re-rating algorithm is applied to denoise the data in T_1 and T_2 with datasets collected at a later time (T_2 and/or T_3). The performance of the CF algorithms when predicting each of the denoised datasets (T_1 and T_2) is summarized in Table 4,

Algorithm 2 Denoising based on one-source re-rating

```
Given user u, movie m, trial $t = 1,2,3$, and $\gamma > 0$
$R_{ut} = \{r_1, ..., r_k\}$ (ratings $\neq 0$ for user u in one trial)
$R_{ut'} = \{r_1, ..., r_l\}$ (ratings $\neq 0$ for user u in a different trial)
being 0 the value of the non-rated item
for $r_i \in R_{ut}$ do
    if $0 < |R_{ut}(m) - R_{ut'}(m)| > \gamma$ then
        $R_{ut}(m) = 0$ (SD: we delete rating)
    else if $0 < |R_{ut}(m) - R_{ut'}(m)| < \gamma$ (MD: ratings in both trials differ by less than a given threshold $\gamma$) then
        $R_{ut}(m) = \mathcal{M}(R_{ut}(m), R_{ut'}(m))$ (where $\mathcal{M}$ is defined in Eq. 1)
    else
        $R_{ut}(m) = R_{ut'}(m)$
        (A: we do nothing, we keep rating as valid)
    end if
end for
```

Datasets	$T_1 \circledcirc (T_2, T_3)$	ΔT_1
User-based *k*NN	0.8647	13.4%
SVD	0.8800	14.1%

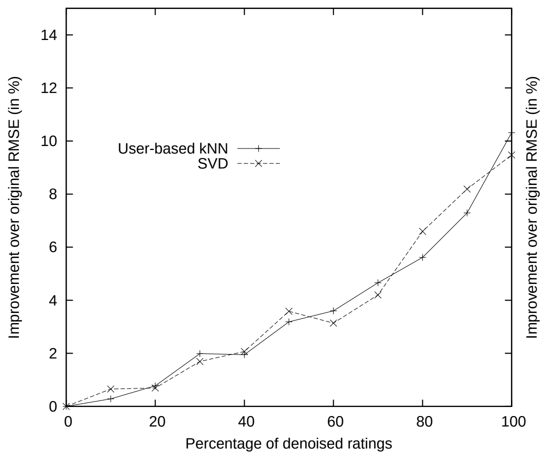
Table 5: RMSE of the algorithms when denoising T_1 with rating values from T_2 and T_3.

where each of the Δ columns contains the performance gain with respect to predicting the original – noisy – dataset, as summarized in Table 2. As shown in Table 4, the accuracy increases above 11% when denoising the original dataset with the information from another dataset collected at a later time. The best results are obtained when denoising T_1, whereas denoising T_2 (originally less noisy than T_1) yields lower relative improvement (last column in Table 4). It is important to note that the accuracy always increases after denoising all datasets. Of particular interest is the case of denoising a noisy dataset with another equally noisy dataset: *e.g.,* denoising T_1 with data from T_3. Note that in this case the performance is significantly better than when predicting the originally less noisy dataset T_2. Our findings illustrate how the proposed algorithm is able to significantly denoise datasets with varying amounts of natural noise.

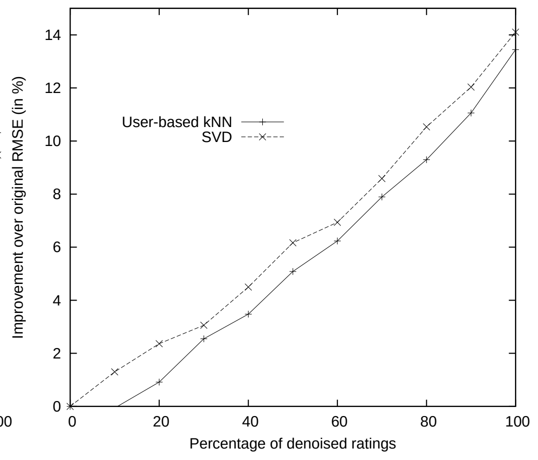
Two-source re-rating.

In order to analyze the value of multiple re-ratings *vs.* one re-rating, we evaluate the denoising algorithm based on two-source re-rating (*i.e.* where users provide three ratings per item: the original rating plus two re-ratings).

Table 5 reports the results where T_1 is denoised using the data from T_2 and T_3, according to Algorithm 1. Note that the *RSME decreases up to* 14.1%, which is a significant improvement with respect to the original results reported in the second column of Table 2 and to the single re-rating denoising results reported in Table 4. For instance, the RMSE for the SVD is now measured to be 0.88, compared to the original 1.0244 for T_1 (noisiest dataset) or 0.9861 for T_2 (less noisy). It is also interesting to note that both algorithms experience very similar improvements. From our results, we conclude that having two re-ratings allows for better denois-

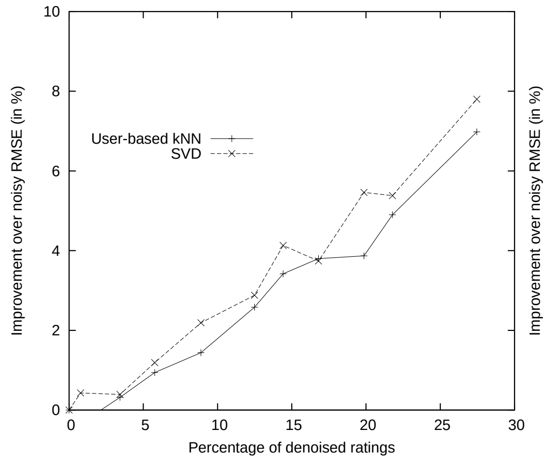


(a) One-source denoising (T_1 via T_3)

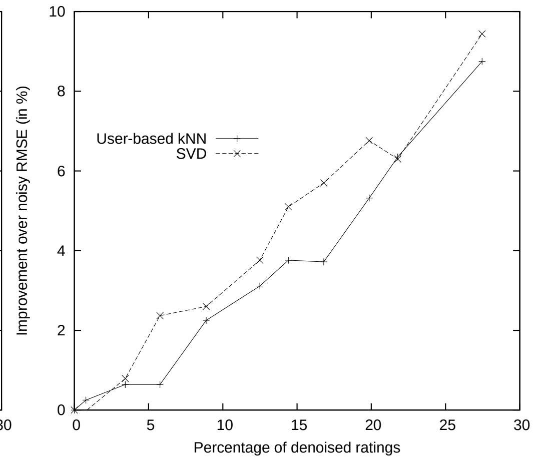


(b) Two-source denoising (T_1 with T_2 and T_3)

Figure 2: Improvement in the prediction error as a function of the percentage of denoised ratings.



(a) One-source denoising (T_1 via T_3)



(b) Two-source denoising (T_1 with T_2 and T_3)

Figure 3: Improvement of the prediction error as a function of the percentage of denoised ratings. Denoising extreme ratings (*i.e.* 1 or 5).

ing than having just one-rating, yielding significant and comparable improvements for all algorithms. We leave for future work the study of the accuracy gains as the number of re-rating sources increases.

4. PARTIAL DENOISING

The denoising analysis presented in Section 3 assumed that there were one or two re-ratings for each of the original ratings. However, it is unrealistic to expect users to provide feedback about all items more than once. A more practical setting would consist on denoising selected ratings (partial denoising). In this section, we analyze the relation between the percentage of denoised ratings and the prediction error. We also propose ways to minimize the number of processed ratings while keeping a significant improvement in RMSE.

In the rest of this section we report our results in $\%$ of improvement over the noisy RMSE measured for T_1 (see first column in Table 2). Note that a particular algorithm having a higher value in the graph does not mean that it performs better than the other in terms of RMSE. It means that its relative improvement in $\%$ of prediction accuracy is larger.

4.1 Random Denoising

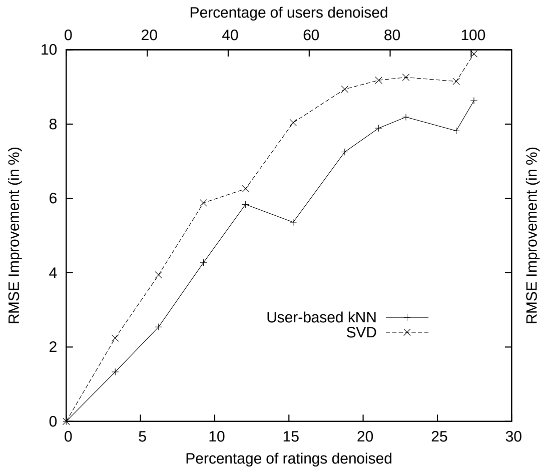
Figure 2 illustrates the evolution in the percentage of improvement of the studied algorithms as the percentage of (randomly chosen) denoised ratings increases. Figure 2a summarizes the results when denoising the data in T_1 with data from T_3 (one-source rerating or denoising), whereas Figure 2b depicts the improvement in RMSE when denoising the data in T_1 via data from T_2 and T_3 (two-source re-rating or denoising).

Both recommendation algorithms follow a similar trend both in the one-source and two-source re-rating experiments. From our experiments, it seems that the relation between the accuracy improvement and the amount of denoising is algorithm independent.

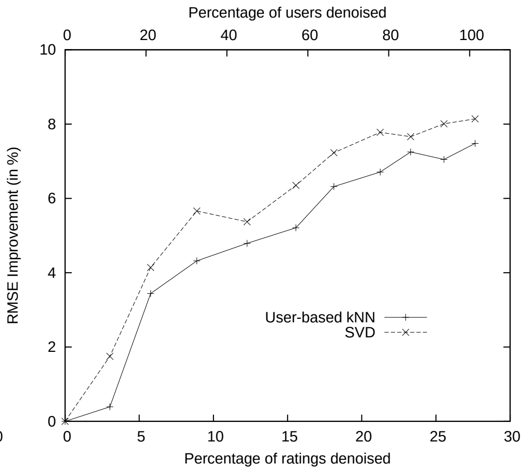
In the case of one-source re-rating, an improvement of 5% is obtained by denoising over 75% of the original ratings, whereas the same improvement is achieved by denoising less than 50% of the original data in the two-source approach.

4.2 Data-dependent Denoising

In the previous analysis, the ratings to be denoised were ran-



(a) One-source denoising (T_1 via T_3)



(b) Two-source denoising (T_1 with T_2 and T_3)

Figure 4: Improvement of the prediction error as a function of the percentage of selected users (top X axis) and the percentage of denoised ratings (bottom X axis) in the dataset. Users are selected according to their "noisiness" and only extreme ratings are denoised.

domly selected, which is probably not the optimal strategy. We propose next a procedure to maximize the gain in performance with a restricted number of re-ratings. The selection of which ratings to re-rate will depend only on the original rating value – hence it is a data-dependent approach. We consider the options of denoising mild (*i.e.* in the middle of the scale) *vs.* extreme (*i.e.* at each end of the scale) ratings. At first sight, it might seem that mild ratings would be the preferred ratings to denoise, given that they are typically more inconsistent. However, the findings of our experimental analysis show the opposite: denoising extreme ratings yields larger performance gains than denoising mild ratings. This behavior might be due to several reasons, including that the contribution of extreme ratings to the final recommendation decisions is generally larger (both positively or negatively) in the algorithms under study. Also, the RMSE measure penalizes large errors that are more likely to occur with extreme ratings. Therefore, errors associated to predicting extreme ratings are likely to impact RMSE more than errors associated with mild ratings.

Figure 3 depicts the RMSE gains when denoising only extreme ratings in T_1 (*i.e.* ratings of 1 or 5 in the scale used in the experiments) using the data from T_3 (Fig. 3a) and T_2 and T_3 (Fig. 3b). In this case, extreme ratings only represent 27% of the total ratings – hence, the x-axis in the graphs runs only up to 30%. A 5% increase in performance is achieved when re-rating less than 25% of the original ratings with one-source denoising and less than 20% with the two-source approach.

The maximum accuracy improvement when denoising all extreme ratings (*i.e.* 27% of all ratings) is between 7 to 9.5%, depending on the algorithm and the denoising approach (one *vs.* two-source). Note that this performance gain is similar to the gain obtained when denoising 80% of the – randomly selected – ratings (see Figure 2).

Interestingly, one-source and two-source denoising achieve comparable accuracy gains: the difference between the curves in Fig. 3a and Fig. 3b is around 1%. From these findings, we conclude that the performance gains obtained with more than one re-rating are almost negligible when using a smart strategy to select the ratings to process.

In sum, our experimental analysis implies that asking users to re-rate (just once) the items with extreme ratings (typically a small percentage of the total number of ratings provided) would yield accuracy gains of at least 7%. These results raise a new research question: would rating the same percentage (around 27%) of unseen items yield higher performance gains than re-rating items with extreme ratings? The experiments discussed in Section 5 aim at providing an answer to this question.

4.3 Data and User-dependent Denoising

In the previous section we have focused on selecting ratings to denoise based on their value. However, users are not equally consistent when providing their ratings, such that noisy data is typically generated by inconsistent users. Therefore, it would make sense to *only* ask inconsistent or unstable users to re-rate past ratings.

In order to identify inconsistent users in our dataset, we: (1) select the users with the highest levels of inconsistency – *noisiness* – in their ratings; and (2) apply the previously explained *data-dependent* denoising algorithm *only* to the ratings coming from such users. The results of this analysis are summarized in Figure 4. This Figure may be compared to Fig. 3, where the same algorithm was used but without selecting the users. Note that data and user-dependent denoising needs less than 10% of denoised ratings by 30% of the users to yield over 5% of RMSE improvement. Similar results are obtained when denoising around 20% of the ratings – by all the users – in the data-dependent approach (Fig. 3). The data and user-dependent denoising algorithm would be completely unobtrusive to the remaining 70% of the users. Figure 4a reports the results for data and user-dependent one-source denoising of T_1 with data from T_3. Results may be compared to those of Fig. 3a. As in the case of data-dependent denoising, the difference between one-source and two-source denoising is relatively small (compare Figures 4a and 4b).

Detecting "noisy" users.

The data and user-dependent approach assumes that there is an effective method to compute user *inconsistencies* and hence detecting noisy users. However, this is not a trivial question to answer. In our experiment, we detect noisy users *a posteriori* by using the information from different rating trials: user *noisiness* is given by the average per-user prediction error between the different rating trials and we use this value to decide the group of users to denoise.

In a practical setting, an incremental approach could be applied where all users are asked to re-rate a small number of items and

	one-source $T_1 \circledcirc T_3$	two-source $T_1 \circledcirc (T_2, T_3)$
Random	0.74%	1.64%
Data-dependent	5.14%	6.33%
Data and user-dependent	7.25%	8.54%

Table 6: Average improvement in RMS for denoising 20% of the ratings.

this re-rating data is used to estimate their noisiness. However, we leave this analysis for future work.

4.4 Summary

A final summary on partial denoising is included in Table 6 that reports the *average* result when denoising 20% of the ratings with each of the approaches.

5. THE VALUE OF A RATING

Regardless of the approach for generating recommendations, all RS aim at modeling their users' tastes. Therefore, the more a RS knows about its users, the more accurate its predictions should be. In the case of RS based on CF and explicit feedback, this translates into the following – intuitive and somewhat obvious – observation: "The more ratings the system has from a user, the more accurate the predictions should be for that user". In order to gather as much information as possible about the user's likes and dislikes, RS typically ask their users to rate a diverse set of unseen items, particularly to the users that are new to the system [16].

However, the denoising approach proposed in this paper has raised a complementary view to collecting data from users: user feedback is noisy; therefore, the accuracy of a RS might improve more by asking users to re-rate previously rated items – which would allow to denoise the already collected ratings – than by asking them to rate unseen items. In order to validate this hypothesis, we now report the results of an experiment designed to measure the accuracy gain introduced by re-rating when compared to the accuracy gain obtained when adding new ratings to a RS. Our experiment consists of the following steps: **(1)** Measure the value of a new rating in our dataset (T_1 in the experiment) by randomly removeing 50% of the original ratings and measuring the improvement in RMSE as the remaining 50% of ratings are added. **(2)** Measure the value of a re-rating in the same circumstances by again randomly removing 50% of the original ratings but now analyzing the improvement introduced by re-rating a fraction of the remaining ratings.

For simplicity, the results presented in this section correspond to user-based *k*NN. However, we obtained similar results for the other algorithms. Figure 5 depicts the relative improvement in RMSE as new ratings are added or existing ratings are re-rated using the random or data-dependent approach [3]. Note that re-rating outperforms adding new ratings in all cases, and especially in the data-dependent approach. For example, adding less than 300 re-ratings to the original 1064 ratings, yields an RMSE improvement of 8.7% (from 1.0538 RMSE to 0.9626). However, randomly adding almost 700 new ratings barely improves the RMSE by 3% [4].

The figure includes the plot of a linear fit to each of the curves, which allows to compute the value of adding a single rating (new or re-rated): in the case of data-dependent re-rating, the improvement is of 0.035% per rating, which is 10 times *larger* than the improvement obtained when adding new ratings (0.0037% per rating). Thus, our experiments confirm the previously stated hypothesis: in the conditions of our study, re-rating items – and hence denoising the input data – yields larger performance gains than adding ratings of unseen items. Our results are dataset and algorithm-dependent. It must be noted that there is an obvious limitation to this finding: you can only re-rate as many ratings as you have previously rated.

In our future work, we plan to apply the proposed denoising approach to other datasets and algorithms in order to better understand the scope of our findings.

[3]We do not compare to the most favorable user and data-dependent approach since, as we explained, it is not fair to assume we can classify users with re-rating data.

[4]It should be noted that there might exist strategies to add more informative new ratings that maximize information gain. We leave this research for future work.

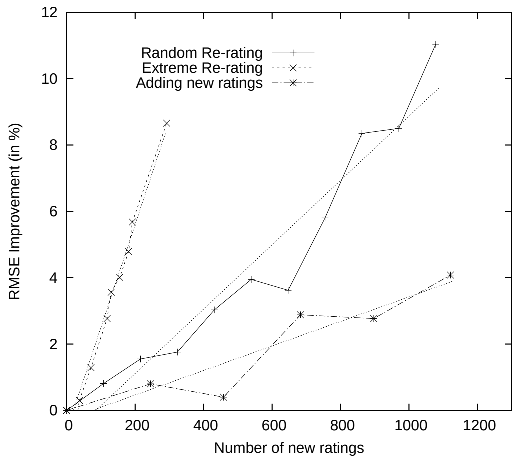


Figure 5: Improvement in RMSE when adding new ratings *vs.* re-rating existing ones. Results correspond to predicting dataset T_1 with the re-rating values drawn from T_3.

6. RELATED WORK

The bias introduced in RS by noise in explicit user feedback has been known for some time. However, regardless of the large and growing bibliography in the area of RS and the importance of this issue in the design of effective strategies, there are not many references in the literature. To the best of our knowledge, the only work that proposes an approach to deal with natural noise directly is that of Mahony *et al.* [13]. In their work, they classify noise in RS into *natural* and *malicious*. The former refers to the definition of natural noise used in this paper, while the latter refers to noise that is deliberately introduced in a system in order to bias the results. Even though the focus of their work is on *malicious* noise, they do propose a denoising algorithm that can be used to detect *natural* noise. Assuming that their recommendation algorithm is always accurate within a certain threshold (γ) they consider any rating that deviates more than γ as containing natural noise and consequently disregard it in the prediction process. The authors do not analyze the effects of this denoising algorithm but they do report a marginal improvement in their baseline algorithm for certain values of γ.

Other authors do not address the issue of reducing the effect of natural noise but do aim at quantifying it. Hill *et al.* [11], for instance, were aware of the effect of rating inconsistencies in prediction accuracy and designed a small scale experiment to measure the reliability in user ratings. They carried out a two trial user study with 22 participants and a time difference of 6 weeks between trials. Unfortunately, the noise in user ratings was a side issue in their overall study. Cosley *et al.* [7] carried out a similar experiment using a rate re-rate procedure with two trials and 212 participants. They selected 40 random movies in the center of the rating scale (*i.e.* 2,3 or 4 rating) which participants had already rated in the past – months or even years earlier, according to the authors. They reported participants being consistent only 60% of the time. Finally, Herlocker *et al.* [10] discuss the noise in user ratings in their review of evaluating methods for recommender systems. In particular, they introduce the concept of the "magic barrier".

Our approach to denoise user feedback data via a re-rating strategy is motivated by the *test-retest* procedure used in Social and Information Sciences to measure the reliability and stability of surveys and respondents [17]. In addition, our strategy to denoise extreme ratings is inspired by the notion of *Extreme Response Style* (ERS) in the Social Sciences literature [8]. The importance of extreme ratings has also been discussed in recent recommender system publications [6] [15].

7. CONCLUSIONS

In this paper, we have analyzed the impact that natural noise has in common RS and proposed novel strategies to remove part of this noise in a preprocessing step.

We have first measured the RMSE of five RS algorithms (two baselines and three popular CF algorithms) in the presence of natural noise and have found that: (1) SVD and kNN were less sensitive to this type of noise than our baselines; and (2) all the algorithms are more sensitive to noise in the target (test) dataset than in the training set.

Next, we have proposed a novel algorithm to denoise rating datasets from re-rating data: *i.e.* by asking users to rate again previously rated items. We have measured accuracy improvements above 14% in the original RMSE in the case where each item is rated three times (complete denoising). We have shown that this gain is not dependent on the particular algorithm used for the prediction. However, in most situat ions it is unlikely to have re-ratings for every item in the dataset. Therefore, we have also studied the behavior of the proposed denoising strategy when processing only part of the ratings (partial denoising). We have shown that simple strategies, such as denoising only extreme ratings (data-dependent denoising), which is equivalent to denoising less than 20% of the ratings in our dataset, yields over 5% of improvement with respect to the original RMSE. Furthermore, we have proposed a user-dependent denoising strategy where similar improvements of 5% are achieved when re-rating less than 10% of the items by only 30% of the users. In this case, the denoising process would not affect most of the users.

Finally, we have analyzed the value of a new rating *vs.* a re-rating from the perspective of accuracy gains. In our experiments, we have found that when following the right denoising strategies a re-rating yields up to 4 times larger accuracy gains than a new rating. Therefore, in such circumstances asking users to re-rate items would allow to gather more information about the user than asking them to rate new and unrelated items.

Note that the dataset used in this work resembles typical RS rating datasets, such as the one from the Netflix Prize. Although we believe that the qualitative results reported in this paper should generalize to other datasets and domains, we plan to study other datasets in order to cross-validate results, build more accurate models of the natural noise and propose better denoising approaches. In this work – as it is commonly done in the RS literature, we have assumed that RMSE is an appropriate success measure. In future work we plan to validate the effect of denoising through targeted user studies. Finally, both the proposed denoising algorithm and the strategies for partial denoising are amenable to variations that should be explored in the future.

We believe that the work presented in this paper opens up a new avenue of research in explicit-feedback RS that is complementary to one of the common approaches: instead of focusing on improving a RS algorithm and its response to a specific dataset, we have shown that improving the *quality* of the original dataset via a denoising step yields significant performance gains in the most common CF algorithms. In addition, the proposed approach is not intrusive to the user nor to the existing RS algorithm/system, as it simply adds re-ratings where appropriate.

Acknowledgements.

This work has been partially funded by an ICREA grant from the Generalitat de Catalunya.

8. REFERENCES

[1] 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. on Knowl. and Data Eng.*, 17(6):734–749, 2005.

[2] X. Amatriain, J. Pujol, and N. Oliver. I like it... i like it not: Evaluating user ratings noise in recommender systems. In *Proc. of UMAP'09*, 2009.

[3] C. Anderson. *The Long Tail: Why the Future of Business is Selling Less of More*. Hyperion, 2006.

[4] R. M. Bell and Y. Koren. Improved neighborhood-based collaborative filtering. In *Proc. of KDD Cup*, 2007.

[5] J. Bennet and S. Lanning. The netflix prize. In *Proc. of KDD Work. on Large-scale Rec.. Sys.*, 2007.

[6] S. Berkovsky, Y. Eytani, T. Kuflik, and F. Ricci. Enhancing privacy and preserving accuracy of a distributed collaborative filtering. In *Proc. of Recsys '07*, 2007.

[7] D. Cosley, S. K. Lam, I. Albert, J. A. Konstan, and J. Riedl. Is seeing believing? how recommender system interfaces affect users' opinions. In *Proc. of CHI'03*, pages 585–592, 2003.

[8] E. A. Greenleaf. Measuring extreme response style. *The Public Opinion Quarterly*, 56:328–351, 1992.

[9] M. Harper, X. Li, Y. Chen, and J. Konstan. An economic model of user rating in an online recommender system. In *Proc. of UM 05*, 2005.

[10] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. on Inf. Syst.*, 22(1):5–53, 2004.

[11] W. Hill, L. Stead, M. Rosenstein, and G. Furnas. Recommending and evaluating choices in a virtual community of use. In *Proc. of CHI '95*, pages 194–201, 1995.

[12] D. W. Oard and J. Kim. Implicit feedback for recommender systems. In *AAAI Works. on Rec. Sys.*, 1998.

[13] M. P. O'Mahony. Detecting noise in recommender system databases. In *Proc. of IUI'06*, pages 109–115, 2006.

[14] A. Paterek. Improving regularized singular value decomposition for collaborative filtering. In *Proceedings of KDD Cup and Workshop 2007*, 2007.

[15] R. Rafter, M. O'Mahony, N.J.Hurley, and B.Smyth. What have the neighbours ever done for us? a collaborative filtering perspective. In *Proc. of UMAP '09*, 2009.

[16] A. M. Rashid, I. Albert, D. Cosley, S. K. Lam, S. M. Mcnee, J. A. Konstan, and J. Riedl. Getting to know you: Learning new user preferences in recommender systems. In *Proc. of IUI '02*, 2002.

[17] G. Torkzadeh and W. J. Doll. The test-retest reliability of user involvement instruments. *Inf. Manag.*, 26(1):21–31, 1994.