Utilizing Favorites Lists for Better Recommendations

Mustafa Abualsaud
University of Victoria
Victoria, Canada
Email: mabualsa@uvic.ca

Alex Thomo
University of Victoria
Victoria, Canada
Email: thomo@cs.uvic.ca

***Abstract*—We present two new models that take into account the information available in user-created "favorites" lists for enhancing the quality of item recommendation. The first model uses the popularity and ratings of items in the lists to predict ratings for new items to users that have rated some items on the lists. The second model is a matrix factorization model that incorporates lists as implicit feedback in ratings prediction. We compare our two approaches against another work for utilizing favorites lists, as well as the popular Singular Value Decomposition (SVD) on two large Amazon datasets and show that utilizing favorites lists gives significant improvements, especially in cold-start cases.**

I. Introduction

Over the last few years, many e-commerce and entertainment websites have employed Recommender Systems (RS) to give personalized suggestions to their users about items of potential interest. The number of items in these websites is typically very large, therefore, it is hard for users to manually search for items that are relevant to them. Many RS try to generate a set of items that a user might be interested in by using explicit and/or implicit feedback produced by the website's users [1]. RS have proven to be very effective in helping users find items that are most valuable to them and have been a research topic for many years.

Lately, many websites have enabled users to create lists of preferred items. For example, YouTube, a popular video-sharing website, allows users to create a playlist of videos they like. These playlists usually group items around a certain topic or theme for easier and organized access. Amazon, an online retail store, has also implemented a similar service called Listmania, which allows users to create their own lists that can be viewed by other users and be voted on. The voting on these lists is binary in which a "yes" vote indicates that the list is relevant or likeable, and "no" as not. One can view these lists as external information that can be valuable when used in a RS. CIRC [2], for example, utilizes list information (items and in particular votes) for enhancing the quality of recommendations to users.

In essence, the process of a user creating a list of items around a certain theme is just as letting the user deliberately and voluntarily choose items that fall under the list theme. From a different perspective, this idea can also be seen as the process of users recommending items to themselves and categorizing them under different themes. This, as a result, gives valuable insight into the association of items and what theme they fall under, which can be of great benefit to RS.

Lately, Amazon has removed the list voting function, thus users can no longer vote on a specific list.[1] This might be because items that appear in lists of low likeability (i.e. many "no" votes on the list) might get less sales than what Amazon has expected. Other websites, such as YouTube, considers the favorites lists private by default, so there is no option to vote on (or even see) them. The unavailability of the voting function, unfortunately renders approaches like CIRC [2] inapplicable. In this paper, we remedy the lack of votes on user-created lists by introducing two new algorithms FLARES[2] and SVD-FLARES. The idea for FLARES is to weigh each user-created list by the popularity and the quality of its items. We achieve this by averaging the lower bound of the Wilson score of each item on a list to obtain a weight for the list. Then we appropriately aggregate these weights to produce rating predictions for users that have rated at least one item on one of the lists.

Our second algorithm, SVD-FLARES, extends the Singular Value Decomposition (SVD) algorithm. Because of the success SVD has demonstrated in the Netflix challenge [3], the method has gained a lot of popularity and has been incorporated into different RS [4], [5], [6]. More specifically, SVD-FLARES is a matrix factorization model that is similar in spirit to the SVD++ paradigm advocated by Koren and Bell (c.f. [7]).

Using two large Amazon datasets, we compare FLARES and SVD-FLARES, along with CIRC, against the popular SVD approach, and show that utilizing favorites lists gives significant improvements, especially in cold-start cases.

The contributions of this work are as follows:

1) We propose a new algorithm, FLARES, that does not consider the list-votes information but instead incorporates the popularity and the quality of items to determine the importance weight of the lists.
2) We propose a second algorithm, SVD-FLARES, that incorporates the information contained in the lists as implicit feedback in a matrix factorization model.
3) We provide an extensive evaluation over two Amazon datasets collected in two different time periods. The datasets have been split into different categories to demonstrate the performance of each approach under categories where the "cold-start" problem holds.

The rest of the paper is organized as follows. In Section II, we describe related work. In Section III, we describe our first algorithm, FLARES. In Section 3, we describe our second algorithm, SVD-FLARES. In Section V, we present our experimental evaluation. Finally, Section VI concludes the paper.

[1] The number of old votes can still be found.
[2] Acronym for **F**avourite **L**ist B**A**sed **RE**commender **S**ystem

II. Related Work

The quest for utilizing more information beyond the explicit user-provided ratings is ubiquitous in many works on recommender systems. Among them are works utilizing trust (c.f. [8], [9], [10], [11], [12], [13], [14], [15], [16]), time (c.f. [17], [18], [19], [20]), location (c.f. [21], [22], [23]), and context (c.f. [24], [25], [26], [27], [28]). An overview is given in [29].

Surprisingly, to the best of our knowledge, utilizing user-created lists has not received a lot of attention despite the fact that such lists are present in many important sites, such as Amazon, Netflix, and YouTube. In fact, we are aware of only [2], which is based upon the work of one co-author, that has studied the usability of user-created, favorites lists in improving the quality of recommender systems. The results of [2] are quite impressive especially for cold-start users. However, [2] is inapplicable in cases when there is no possibility to collect user votes on the lists. The current work focuses on filling this gap by remedying the lack of votes using other measures we can collect from the items on the lists.

III. FLARES

The FLARES algorithm harnesses the information found in lists to determine an association between items and users. Since users usually group items of a certain topic to create a list, these items are most likely to be relevant to other users who are interested in the same topic. A user that gives high ratings to items of a certain topic may indicate that he is interested in new items that fall under the same topic as well. These new items, along with the items that the user has already rated, most likely share the same lists or appear in other similar lists. However, not all lists are created equal. Lists that contain popular and good quality items should be weighed higher than other lists that do not. FLARES determines how "interested" a user is in a topic by looking at what items the user has rated, the ratings of these items, and what lists did these items appear in. FLARES uses this information to determine the "strength" of lists and to predict the ratings of unknown items to users.

A. Preliminaries

Figure 1 illustrates the proposed graphical model that incorporates lists, items, and users. We denote the set of users by $U = \{u_1, \ldots, u_{|U|}\}$, the set of items by $I = \{i_1, \ldots, i_{|I|}\}$, the set of lists by $L = \{l_1, \ldots, l_{|L|}\}$, the set of items that has been rated by a specific user u by R_u, the set of items in a list l by I_l and the set of lists that contain item i by L_i.

We represent users, items, and lists as nodes in a graph G of relationships. An edge connecting an item i with a user u is denoted by a triple $(u, i, r_{u,i})$, where $r_{u,i}$ is the known rating that user u has given to item i. There is also an unweighted edge connecting a list l to an item i, if l contains i. The set of edges is denoted by E. Finally, we generate the list-weight set

$$\mathcal{L} = \{(l_1, w_1), \ldots, (l_f, w_f)\}$$

where the weights describe the extent of how popular and good quality items each list contains. The exact method of computing the weights of the lists is given in the next section. These weights are represented as node labels in our graph.

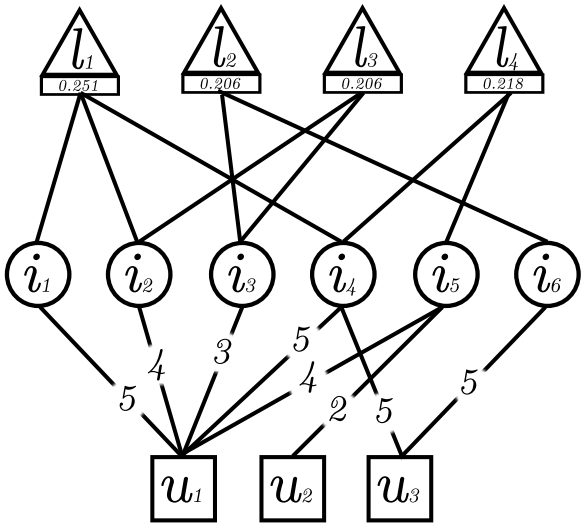


Fig. 1: The graphical model of FLARES, comprising of User-Item-Rating and Item-List relationships. Each item can occur in zero or many lists. The list weight indicates the list's "strength", and is based on the items that the list contains.

B. Item Scores

Consider the fraction of positive ratings (FPR) for an item i

$$\text{FPR}(i) = \frac{\text{Number of positive ratings for } i}{\text{Total number of ratings for } i}.$$

The problem with FPR can be illustrated by the following example. Let us consider a positive rating to be ≥ 4 (in a 1 to 5 scale). Assume we have items i_1 and i_2 and i_1 has been given a rating of 5 by one user out of one user, and i_2 has been given a rating of 5 by ten users out of ten users. Using the above formula we have $\text{FPR}(i_1) = 1/1 = 1$, and $\text{FPR}(i_2) = 10/10 = 1$. This as a result would assign the score of both items to be equal ignoring the fact that ten users have agreed on i_2 to be of good quality, while only one user has considered i_1 to be of good quality. A similar argument can be made for cases where there are many negative ratings and few positive ratings of an item.

In statistics, a confidence interval allows us to estimate the "true" value of an observation by providing an upper and lower bound of estimated values. In our case, we are given the ratings of items by multiple users and we need to estimate the confidence interval for the true value of the item's FPR. In other words, we need to determine how certain we are that an item is of good quality.

There are different methods to calculating confidence intervals. We consider the Wilson's score interval. Given an observation $\hat{p}_i$ for FPR for item i, we need to determine the confidence interval for $\hat{p}_i$ with respect to the true fraction p_i. This can be done using the Wilson's score to determine the interval $p_i^- \leq p_i \leq p_i^+$ ([30]). Since we want to determine how *least* confident we are, we only consider the lower bound

of the interval

$$p_i^- = \frac{\hat{p}_i + \frac{z^2_{\alpha/2}}{2n_i} - z_{\alpha/2}\sqrt{\frac{\hat{p}_i(1-\hat{p}_i)}{n_i} + \frac{z^2_{\alpha/2}}{4n_i^2}}}{1 + \frac{z^2_{\alpha/2}}{n_i}}$$

where $\hat{p}_i$ = positive ratings/total ratings for i, n_i is the total number of ratings for i, and $z_{\alpha/2}$ is the $(1-\alpha)$ quantile of the stranded normal distribution. We pick a typical α value of 0.95 for a 95% confidence level.

C. List Weights

A list l containing popular and good quality items should be weighed higher than other lists that do not. Using the lower bound of the Wilson's score interval, we calculate the weight of the list as follows:

$$w_l = \frac{\sum_{i \in I_l} p_i^-}{|I_l|}.$$

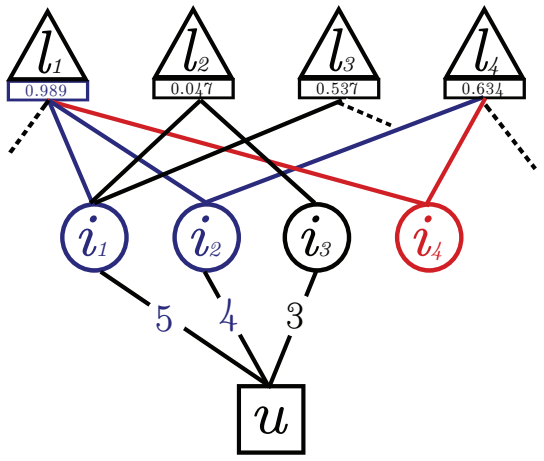
In other words, we compute the average of the lower bounds of the Wilson scores of the items on the list. Thus, the weight of the list would give us a good gauge for the overall popularity and quality of the list's items. This would be useful for predicting a rating that a user would give to an unknown item that appears on a list.

1) Rating Prediction: The exact rating prediction is provided in Algorithm 1, which generates a prediction for rating of item i by user u. The main idea of the algorithm is to look for items j, rated by user u, that appear in lists that contain the unknown item i, and multiply the given rating $r_{u,j}$ by the weight of the list that item j and i appear in. The result is then averaged by the sum of those lists weights. As mentioned earlier, lists of popular and good quality items have a larger weight, thus this method will contribute to a larger predicted rating for items that appear in such lists, yet still controlled by the ratings user u has given. We show an example in Figure 2.

Algorithm 1: Predicts rating of item i by user u

```
Input: A user-item-list graph G, user u, and item i
Output: Predicted rating r̂_{u,i}
Let L_i ⊆ L be the set of lists that contain i;
Initialize variables, nom, denom = 0;
for item j in R_u do
    for list l in L_j do
        if l ∈ L_i then
            nom += w_l × r_{u,j};
            denom += w_l;
if denom ≠ 0 then
    return nom/denom;
return 0;
```

Put differently, the list weight can be seen as a collaborative technique to collect the opinions of many users on how they feel about the list's items, which we use to derive some insight from the list as a whole. Moreover, when predicting a rating of an unknown item to a user, we need to consider how the user feels about items of a similar topic, thus we take into account the user ratings of items that share the same list as the unknown item.



$$\hat{r}_{u,i_4} = \frac{(5)(0.989) + (4)(0.989) + (4)(0.634)}{0.989 + 0.989 + 0.634} = 4.3786$$

Fig. 2: The graphical model of Algorithm 1. The item in red, i_4, is the unknown item that we need to predict a rating for. The blue items, i_1,i_2, and i_3 are items user u has already rated and appear in lists that contains i_4. The calculation of $\hat{r}_{u,i_4}$ is done by computing the weighted average of the blue items.

IV. SVD-FLARES

SVD is a matrix factorization method that has gained a lot of popularity following its wide success in the Netflix Prize competition. The approach is used to map both users and items to a joint latent factor space of f dimensions, such that the estimate of an unknown rating for item i for user u would be computed as the inner-products of factors.

The SVD approach described by Koren and Bell in [7] is added to the baseline $\mu + b_i + b_u$ predictors to compute a prediction for unknown items, as follows:

$$\hat{r}_{u,i} = \mu + b_i + b_u + \mathbf{q}_i^{\mathbf{T}} \cdot \mathbf{p}_u$$

where the terms are

Term	Definition
μ	Training overall average rating
b_i	Deviation of item i from the average
b_u	Deviation of user u from the average
$\mathbf{q}_i$	Vector of latent factors for item i
$\mathbf{p}_u$	Vector of latent factors for user u

In order to determine the model parameters, b_i, b_u, $\mathbf{q}_i$, and $\mathbf{p}_u$, stochastic gradient descent (SGD), a common parameter estimation algorithm, can be applied to minimize the squared error $e^2_{u,i} = (r_{u,i} - \hat{r}_{u,i})^2$. The SGD algorithm loops through the training set for a number of iterations and updates the parameters at each iteration. Error $e_{u,i}$ is used to learn the parameters by a magnitude of γ, the learning step size, yielding:

$$b_u \leftarrow b_u + \gamma(e_{u,i} - \lambda_1 b_u) \quad (1)$$

$$b_i \leftarrow b_i + \gamma(e_{u,i} - \lambda_2 b_i) \quad (2)$$

$$\mathbf{p}_u \leftarrow \mathbf{p}_u + \gamma(e_{u,i}\mathbf{q}_i - \lambda_3 \mathbf{p}_u) \quad (3)$$

$$\mathbf{q}_i \leftarrow \mathbf{q}_i + \gamma(e_{u,i}\mathbf{p}_u - \lambda_4 \mathbf{q}_i) \quad (4)$$

where λ_1, λ_2, λ_3, and λ_4 are regularization parameters. SVD works well for quantifying the user's preference level when given a dataset that directly reflects the user preference (i.e. explicit feedback), but can also be modified to consider indirect information (i.e. implicit feedback), as in SVD++ [7], which takes into account the particular set of items that a user has already rated.

Our SVD-FLARES algorithm takes advantage of the similarity of items in lists, along with the interests of the user to produce the final predicted rating for the unknown item. The algorithm finds the extent of similarity of the unknown item to other items of the same theme and adds in the interest value of the user to that theme. We believe that adding the user's interest value to a particular theme gives a more accurate prediction, as the more interested a user to a theme, the more likely that he will like the item that fall under that theme. Similarly, a user that dislikes a certain theme is more likely to give a low rating to the item.

SVD-FLARES updates the μ , b_i, b_u, $\mathbf{q}_i$, $\mathbf{p}_u$ parameters the same way SVD does (usually using the SGD method), but differs on how it computes the final predicted value $\hat{r}_{u,i}$. Namely, SVD-FLARES adds in the model the list information to predict the final rating $\hat{r}_{u,i}$. The exact formula for the model is as follows

$$\hat{r}_{u,i} = \mu + b_i + b_u + (\mathbf{q}_i^{\mathbf{T}} \cdot \mathbf{p}_u) + (\mathbf{l}_i^{\mathbf{T}} \cdot \mathbf{p}_u) + \|\mathbf{l}_i\|$$

where the new vector $\mathbf{l}_i$ is the sum of all latent factor vectors of items that co-appear with item i in different lists, divided by the number of lists item i appears in. We experimented with the same equation but dividing by the total number of items in these lists to get $\mathbf{l}_i$, however, we found that the contribution of $\mathbf{l}_i$ in this way was too small to make a difference in the final $\hat{r}_{u,i}$. We therefore choose the proposed method as it contributes more to the final predicted rating $\hat{r}_{u,i}$. Since the latent factor vector of an item is extracted from the ratings matrix, and $\mathbf{l}_i$ sums the latent factor vectors of items that fall under the same theme as item i (i.e. similar items to i), the $\|\mathbf{l}_i\|$ term can be seen as a broad representation of how the community feels about the item. Note that $\mathbf{l}_i$ is initially a zero vector of length equal to the number of factors f. By doing this, SVD-FLARES disallows items that do not appear in any list to contribute anything more to the computation. Figure 3 shows the graphical model of how SVD-FLARES uses list information.

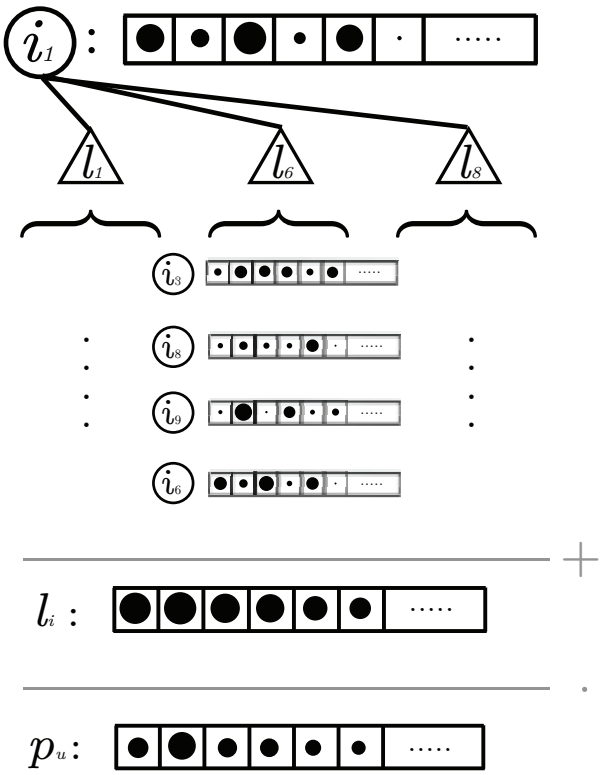


Fig. 3: The graphical model of how SVD-FLARES integrates list information. The item which we need to predict a rating for can occur in zero or many lists. SVD-FLARES sums the vectors of latent factors of items that appear in such lists and averages the sum by the number of lists. In this case, the number of lists item i_1 appears in is three, so we divide the factors by 3 to obtain the final vector of latent factors $\mathbf{l}_i$. We then dot product p_u by $\mathbf{l}_i$ to get the user interest towards items similar to i_1.

V. Experimental Evaluation

We compared our algorithms, FLARES and SVD-FLARES versus CIRC ([2]) and SVD ([7]). The CIRC algorithm utilizes user-created lists to improve recommendation quality, however, it employs the number of votes on the lists to come up with importance scores for the lists. This approach is inapplicable for sites, such as Amazon, that have lately disabled the ability to vote on the lists. Nevertheless, we compare with CIRC on an Amazon dataset that was collected before the voting was disabled.

For SVD and SVD-FLARES, in our experiment, we used a factor dimension $f = 35$ and set $\gamma = 0.005$, $\lambda_1 = 0.02$, $\lambda_2 = 0.02$, $\lambda_3 = 0.05$, and $\lambda_4 = 0.05$. These parameters were tuned as suggested by [7].

A. Datasets

We conduct our experiments on two Amazon book ratings datasets which were collected in two periods[3]. Our first dataset was collected in 2009 using Amazon Web Services (AWS) and contains User-Item-Rating, List-Item, and List-Votes information, back when Amazon allowed voting on the lists themselves. The second dataset is a subset of User-Item-Rating dataset from [31] and spans a period from 1995 to 2013. Since Amazon's API no longer provides direct calls to get list information, we crawled around 38,000 random list pages and got the List-Item information. Table I shows some statistics on the two datasets.

B. Methods of evaluation

In addition to evaluating the Mean Average Error (MAE), we also use the common metrics of precision, recall, F-measure

[3]The two datasets (with the list data) are available upon request.

Data type	Dataset 1	Dataset 2
Items (Books)	405,238	1,475,420
Users	530,160	906,937
Lists	58,618	38,546
Ratings	1,188,435	1,488,083
(list,item) pairs	1,056,932	339,328

TABLE I: Statistics on both datasets

and accuracy with respect to a threshold of 3.5 to classify positive and negative ratings (in a 1-5 scale).

The two datasets have been divided into five categories (see Table II) based on the number of items a user has rated. The reason we divide the dataset is to measure the performance of the models under different situations, such as "cold-start" cases, or on cases when users have rated a large number of items. Categories one and two are those that fall the most under the "cold-start" case, as they contain users that have not rated many items. The "cold-start" case is very common in many domains and is considered a difficult challenge in RS (c.f [32], [33]). Table II shows some statistics on the total number of items and the population of each category of both datasets.

Category	# of ratings (R)	Population	
		Dataset 1	Dataset 2
C1	R <5	497,664	876,975
C2	5 $\leq$ R <10	16383	19,923
C3	10 $\leq$ R <50	9834	9085
C4	50 $\leq$ R <500	1442	945
C5	R $\geq$ 500	28	9

	Total number of ratings per category	
	Dataset 1	Dataset 2
C1	687,079	1,081,872
C2	132,818	124,899
C3	179,973	164,723
C4	158,994	92,356
C5	29,571	11,569

TABLE II: Statistics on the population and the number of rated items in each category

C. Results and discussions

For our evaluation of the four models, CIRC, FLARES, SVD-FLARES, and SVD, we use 5-fold cross-validation. Each model has been evaluated with the same test sets of user-item-ratings. In this section, we describe Figures 4-9 and the conclusions drawn from them. Note that CIRC has not been evaluated on dataset 2 because the model requires the list-vote information, which Amazon no longer provides.

1) The MAE values of the four models are shown in Figure 4. The less the MAE value (i.e the shorter the bar), the better the recommendation quality. We can see that CIRC and FLARES preform significantly better in categories C1 and C2 of the first dataset. Namely, CIRC and FLARES are about 40% better than SVD for these categories. We consider this to be important as it is these categories where the "cold-start" problem occurs. Although SVD-FLARES does not preform as good as CIRC and FLARES, it still produces more accurate predictions than SVD in categories C1, C2, and C3. We also observe that our new model, FLARES, performs as good as CIRC, but without using any list-votes information that CIRC needs. This result shows that it is possible to still achieve good prediction accuracy without using the user votes data on these list, but instead, using the information of items that appear in a list to determine the "strength" of a list. The proposed model, FLARES, is less restrictive in term of the dataset it needs than CIRC, as it does not require any votes on the lists, thus making FLARES more applicable to other domains.

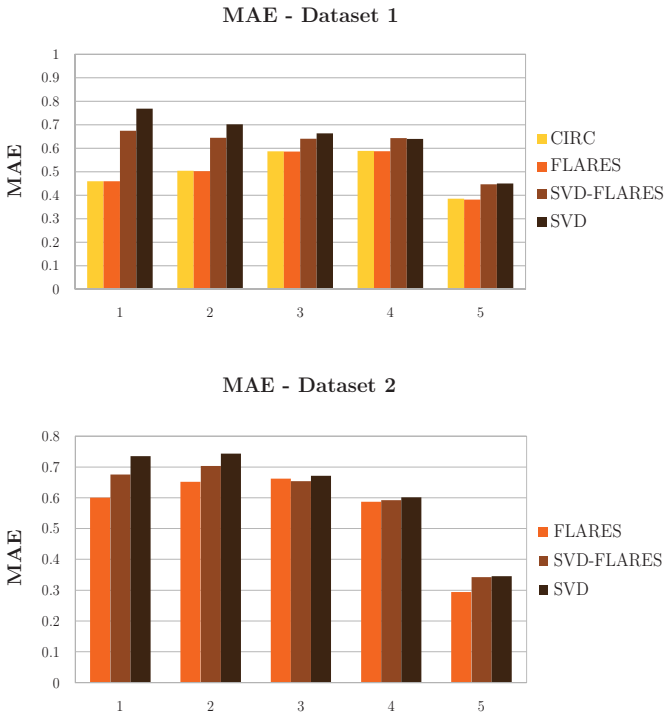


Fig. 4: **MAE for the four models on the two datasets.** The numbers in the horizontal axis correspond to categories C1 to C5. CIRC is inapplicable for Dataset 2 as there are no votes for the lists. Observe that on Dataset 1, where we can compare FLARES with CIRC, FLARES does as well as CIRC, but without using votes.

Regardless of their prediction error, it is interesting to see that CIRC and FLARES outperform the matrix factorization models SVD-FLARES and SVD in both datasets. In the case of dataset 2, where newer ratings and newer lists are provided, FLARES still performs significantly better in categories C1, C2. As shown in Table I in section 4.1, the number of pairs $(list, item)$ in dataset 2 is significantly less than the number of pairs in dataset 1. Since FLARES depends heavily on this information, as shown in Figure 2, we believe

Fig. 5: **MAE percentage improvement over SVD.** Observe that on Dataset 1, FLARES and CIRC do about 40% better than SVD for C1 (cold-start users). The performance of FLARES is similar to CIRC, even though FLARES does not use list-votes. FLARES also shows significant improvement for Dataset 2, albeit less pronounced. SVD-FLARES does better than SVD in all catagories of both Datasets, but C4 of Dataset 1.

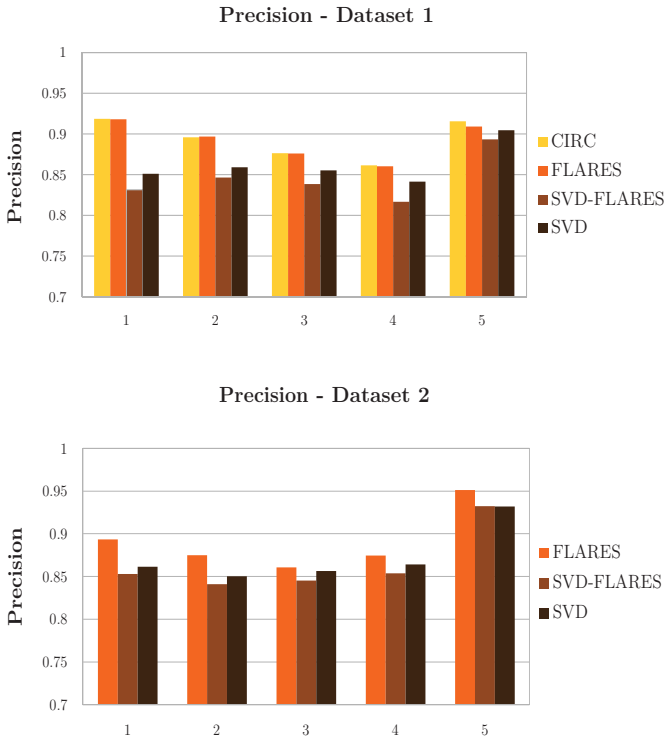


Fig. 6: **Precision of the four models on the two datasets.** Observe that CIRC and FLARES do better than matrix-factorization models. FLARES is about 8% better than SVD.

that the more $(list, item)$ pairs data provided, the more accurate predictions FLARES would make. In terms of the matrix factorization models, SVD-FLARES overall predictions are either the same or more accurate than SVD in both datasets.

2) Figure 5 shows the improvement percentage over SVD. As shown, the improvement of the CIRC and FLARES are much more significant than SVD-FLARES over SVD. For dataset 1, the prediction error of both CIRC and FLARES in C1 has a 40% improvement. We also notice that the improvement of CIRC and FLARES is significant in all categories of dataset 1, where we have a 27% improvement in category C2 and 15% percent improvement for CIRC and FLARES in category C5. Both CIRC and FLARES outperform SVD-FLARES in all categories of dataset 1 in terms of their improvements. Also, SVD-FLARES performs better than SVD in all categories except C4 for dataset 1. We see similar results for dataset 2, albeit less pronounced than those for dataset 1.
3) The precision values of the four models are shown in Figure 6. It is clear that the precision values of the list based models are better than the matrix factorization models. For example, for C1 (cold-start) in dataset 1, the precision of FLARES is about 8% better than the precision of SVD. SVD-FLARES precision is very similar to SVD, but SVD's precision is slightly higher than SVD-FLARES.
4) The recall values of the four models are shown in Figure 7. We notice that SVD-FLARES and SVD perform better in terms of recall than FLARES and CIRC. SVD-FLARES has higher recall than all models regardless of dataset used.
5) The F-Measure values of the four models are shown in Figure 8. Category C5 scores the highest values, with values higher than 0.93 for all models, regardless of which dataset is used. SVD-FLARES and SVD perform very similarly across all categories of the two datasets. The same argument is true for CIRC and FLARES.
6) The accuracy values of the four models are shown in Figure 9. Our experiment shows that the accuracy values of all models are in the 0.8 range or above, regardless of which category or dataset is used. We observe that the values of all models are quite similar to each other in both datasets.

VI. Conclusions

In this paper, we presented two new models, a list based model and a matrix factorization model, that take advantage of the list data available to predict more accurate ratings.

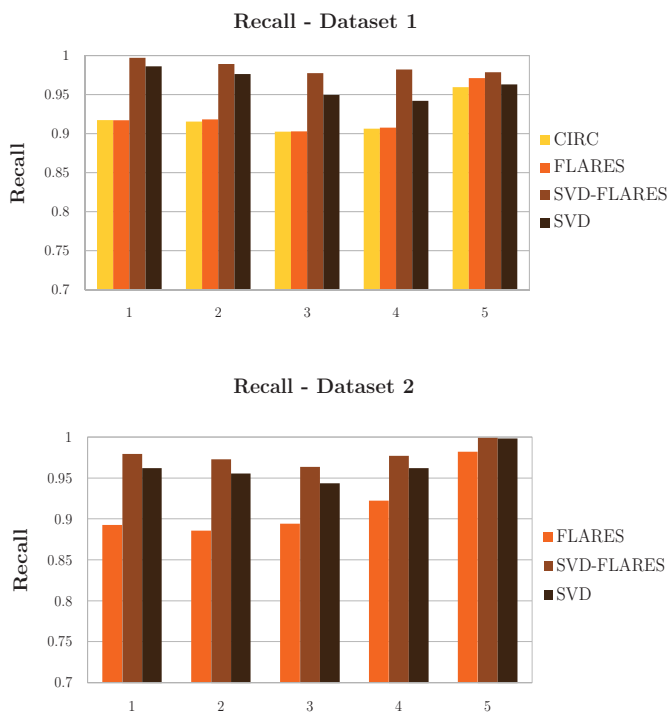


Fig. 7: **Recall of the four models on the two datasets.** Observe that it is Recall where the matrix-factorization methods outperform CIRC and FLARES. The SVD-FLARES model, while close to SVD, outperforms the latter in all categories for both datasets.

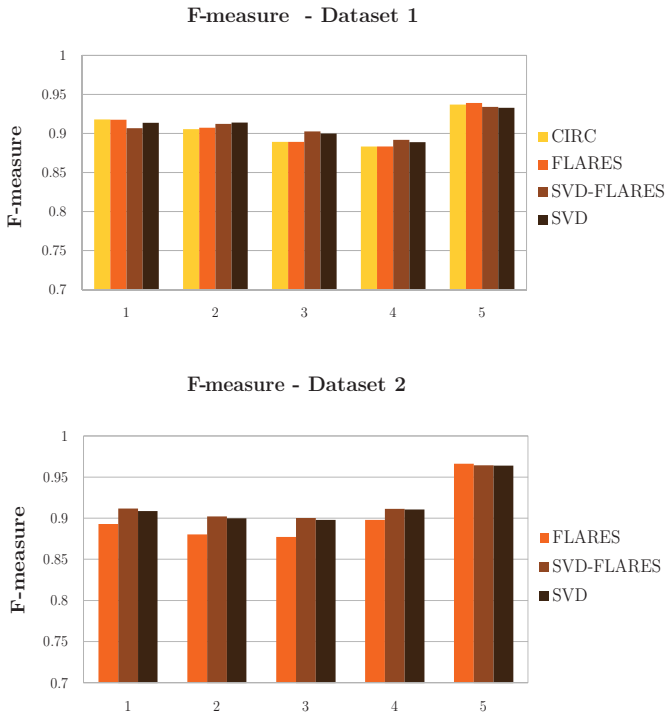


Fig. 8: **F-Measure of the four models in the two datasets.** Observe that the F-Measure values for all the four models are quite close to each other.

The FLARES and SVD-FLARES are less restrictive on their dataset requirement than CIRC because they do not require list-vote data as CIRC does. We compared FLARES and SVD-FLARES versus CIRC and SVD using different evaluation metrics. Our evaluation results for two different Amazon datasets, collected in two different time periods, show that FLARES performs similarly to (and sometimes better than) CIRC even though it uses more restricted information. Both FLARES and CIRC outperform SVD-FLARES and SVD in terms of MAE and precision, while the latter (two) outperform the former (two) in terms of recall. The benefit of using FLARES and CIRC is especially pronounced for cold-start users. In general, we observe that the list-based models, CIRC, FLARES, and SVD-FLARES have better quality than SVD. We believe that organizations that allow users to create their own favorites lists would benefit from using the valuable information found in those lists towards improving their RS.

REFERENCES

[1] G. Karypis, "Evaluation of item-based top-n recommendation algorithms," in *Proceedings of the tenth international conference on Information and knowledge management*. ACM, 2001, pp. 247–254.

[2] M. Khezrzadeh, A. Thomo, and W. W. Wadge, "Harnessing the power of "favorites" lists for recommendation systems," in *RecSys*, L. D. Bergman, A. Tuzhilin, R. D. Burke, A. Felfernig, and L. Schmidt-Thieme, Eds. ACM, 2009, pp. 289–292.

[3] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[4] M. G. Vozalis and K. G. Margaritis, "Applying svd on item-based filtering," in *Intelligent Systems Design and Applications, 2005. ISDA'05. Proceedings. 5th International Conference on*. IEEE, 2005, pp. 464–469.

[5] ——, "Using svd and demographic data for the enhancement of generalized collaborative filtering," *Information Sciences*, vol. 177, no. 15, pp. 3017–3037, 2007.

[6] D. Decoste, D. Gleich, T. Kasturi, S. Keerthi, O. Madani, S.-T. Park, D. M. Pennock, C. Porter, S. Sanghai, F. Shahnaz *et al.*, "Recommender systems research at yahoo! research labs," *Proceedings of Beyond Personalization: The Next Stage of Recommender Systems Research*, 2005.

[7] Y. Koren and R. M. Bell, "Advances in collaborative filtering," in *Recommender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Eds. Springer, 2011, pp. 145–186.

[8] P. Massa and P. Avesani, "Trust-aware collaborative filtering for recommender systems," in *On the Move to Meaningful Internet Systems 2004: CoopIS, DOA, and ODBASE*. Springer, 2004, pp. 492–508.

[9] P. Massa and B. Bhattacharjee, "Using trust in recommender systems: an experimental analysis," in *Trust Management*. Springer, 2004, pp. 221–235.

[10] J. Golbeck and J. Hendler, "Filmtrust: Movie recommendations using trust in web-based social networks," in *Proceedings of the IEEE Consumer communications and networking conference*, vol. 96. Citeseer, 2006.

[11] P. Massa and P. Avesani, "Trust-aware recommender systems," in *Proceedings of the 2007 ACM conference on Recommender systems*. ACM, 2007, pp. 17–24.

[12] F. E. Walter, S. Battiston, and F. Schweitzer, "A model of a trust-based recommendation system on a social network," *Autonomous Agents and Multi-Agent Systems*, vol. 16, no. 1, pp. 57–74, 2008.

[13] M. Jamali and M. Ester, "Trustwalker: a random walk model for combining trust-based and item-based recommendation," in *Proceedings*

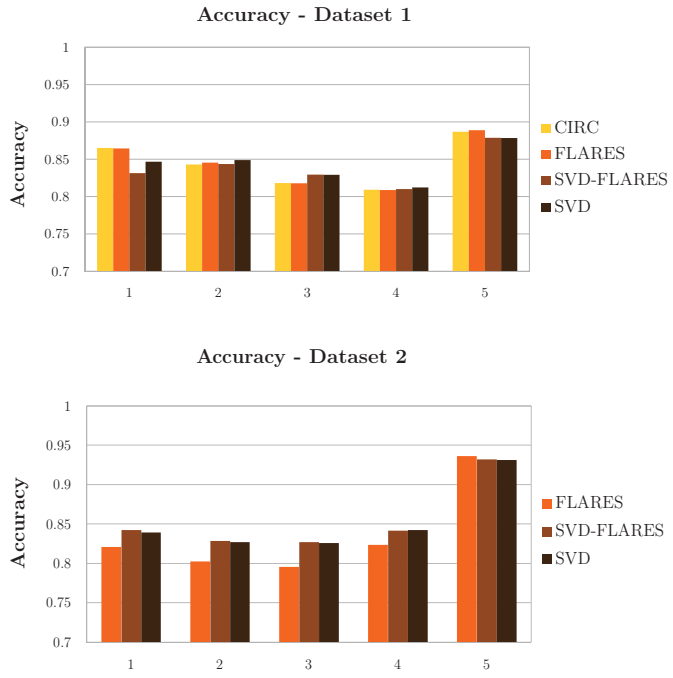


Fig. 9: **Accuracy of the four models in the two datasets**

of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2009, pp. 397–406.

[14] M. Chowdhury, A. Thomo, and W. W. Wadge, "Trust-based infinitesimals for enhanced collaborative filtering," in *COMAD*, S. Chawla, K. Karlapalem, and V. Pudi, Eds. Computer Society of India, 2009.

[15] N. Korovaiko and A. Thomo, "Predictingtrust from user ratings," in *Proceedings of the 3rd International Conference on Ambient Systems, Networks and Technologies (ANT 2012), the 9th International Conference on Mobile Web Information Systems (MobiWIS-2012), Niagara Falls, Ontario, Canada, August 27-29, 2012*, 2012, pp. 263–271. [Online]. Available: http://dx.doi.org/10.1016/j.procs.2012.06.036

[16] ——, "Trust prediction from user-item ratings," *Social Netw. Analys. Mining*, vol. 3, no. 3, pp. 749–759, 2013. [Online]. Available: http://dx.doi.org/10.1007/s13278-013-0122-z

[17] L. Baltrunas and X. Amatriain, "Towards time-dependant recommendation based on implicit feedback," in *Workshop on context-aware recommender systems (CARS'09)*, 2009.

[18] Z. Gantner, S. Rendle, and L. Schmidt-Thieme, "Factorization models for context-/time-aware movie recommendations," in *Proceedings of the Workshop on Context-Aware Movie Recommendation*. ACM, 2010, pp. 14–19.

[19] Y. Koren, "Collaborative filtering with temporal dynamics," *Communications of the ACM*, vol. 53, no. 4, pp. 89–97, 2010.

[20] P. G. Campos, F. Díez, and I. Cantador, "Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols," *User Modeling and User-Adapted Interaction*, pp. 1–53, 2013.

[21] W.-S. Yang, H.-C. Cheng, and J.-B. Dia, "A location-aware recommender system for mobile shopping environments," *Expert Systems with Applications*, vol. 34, no. 1, pp. 437–445, 2008.

[22] J. J. Levandoski, M. Sarwat, A. Eldawy, and M. F. Mokbel, "Lars: A location-aware recommender system," in *Data Engineering (ICDE), 2012 IEEE 28th International Conference on*. IEEE, 2012, pp. 450–461.

[23] J. Bao, Y. Zheng, and M. F. Mokbel, "Location-based and preference-aware recommendation using sparse geo-social networking data," in *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*. ACM, 2012, pp. 199–208.

[24] G. Adomavicius and A. Tuzhilin, "Context-aware recommender systems," in *Recommender systems handbook*. Springer, 2011, pp. 217–253.

[25] C. Biancalana, F. Gasparetti, A. Micarelli, A. Miola, and G. Sansonetti, "Context-aware movie recommendation based on signal processing and machine learning," in *Proceedings of the 2nd Challenge on Context-Aware Movie Recommendation*. ACM, 2011, pp. 5–10.

[26] K. Verbert, N. Manouselis, X. Ochoa, M. Wolpers, H. Drachsler, I. Bosnic, and E. Duval, "Context-aware recommender systems for learning: a survey and future challenges," *Learning Technologies, IEEE Transactions on*, vol. 5, no. 4, pp. 318–335, 2012.

[27] S. Ebrahimi, N. M. Villegas, H. A. Müller, and A. Thomo, "Smarterdeals: a context-aware deal recommendation system based on the smartercontext engine," in *Proceedings of the 2012 Conference of the Center for Advanced Studies on Collaborative Research*. IBM Corp., 2012, pp. 116–130.

[28] N. Yazdanfar and A. Thomo, "Link recommender: Collaborative-filtering for recommending urls to twitter users," *Procedia Computer Science*, vol. 19, pp. 412–419, 2013.

[29] M. Shoaran, A. Thomo, and J. H. Weber, "Social web search," in *Encyclopedia of Social Network Analysis and Mining*, 2014, pp. 1930–1935.

[30] E. B. Wilson, "Probable inference, the law of succession, and statistical inference," *Journal of the American Statistical Association*, vol. 22, no. 158, pp. 209–212, 1927.

[31] J. J. McAuley and J. Leskovec, "Hidden factors and hidden topics: understanding rating dimensions with review text," in *RecSys*, Q. Yang, I. King, Q. Li, P. Pu, and G. Karypis, Eds. ACM, 2013, pp. 165–172.

[32] B. Lika, K. Kolomvatsos, and S. Hadjiefthymiades, "Facing the cold start problem in recommender systems," *Expert Systems with Applications*, vol. 41, no. 4, pp. 2065–2073, 2014.

[33] H. Gao, J. Tang, and H. Liu, "Addressing the cold-start problem in location recommendation using geo-social correlations," *Data Mining and Knowledge Discovery*, pp. 1–25, 2014.