An Unexpectedness-augmented Utility Model for Making Serendipitous Recommendation

Qianru Zheng[1], Chi-Kong Chan[2], and Horace H.S. Ip[1]

[1] Department of Computer Science, City University of Hong Kong, Hong Kong
qrzheng2-c@my.cityu.edu.hk, cship@cityu.edu.hk
[2] Department of Computing, Hang Seng Management College, Hong Kong
chanck@hsmc.edu.hk

Abstract. Many recommendation systems traditionally focus on improving accuracy, while other aspects of recommendation quality are often overlooked, such as serendipity. Intuitively, a serendipitous recommendation is one that provides a pleasant surprise, which means that a suggestion must be unexpected to the user, and yet it must be useful. Based on this principle, we propose a novel serendipity-oriented recommendation mechanism. To model unexpectedness, we combine the concepts of item rareness and dis-similarity: the less popular is an item and the further is its distance from a user's profile, the more unexpected it is assumed to be. To model usefulness, we adopt PureSVD latent factor model, whose effectiveness in capturing user interests has been demonstrated. The effectiveness of our mechanism has been experimentally evaluated based on popular benchmark datasets and the results are encouraging: our approach produced superior results in terms of serendipity, and also leads in terms of accuracy and diversity.

Keywords: Serendipity, Diversity, Recommendation Systems

1 Introduction

Recommendation System (RS) has become a vital part of e-commerce websites. To the users, it provides useful and personalized product recommendations. To the merchants, it provides an effective cross-selling solution. Because of its usefulness, RS has been successfully applied to various application areas, ranging from traditional applications such as movies recommendation [19] in Movielens and Netflix, products recommendation in Amazon [15] and book recommendations [25], to more recent applications in tourism and travel recommendation [16], and social network recommendation [12].

Traditionally, RS algorithms aim at improving recommendation accuracy (e.g, root mean square error (RMSE)), and particularly, recommendation precision. In both collaborative filtering (CF) and content based (CB) methods, precision measures the proportion of the recommended items that are chosen by a user (called the hit items). In order to maintain good accuracies, many recommendation systems tend to recommend only items that are relevant and similar

to the user's previous choices, i.e. those items that match the user's profile. After all, such kind of recommendations is intuitive, safe and usually accurate. However, an over-emphasis on accuracy may restrict the users' choices to the items most similar to his/her previous selections. After all, a user may get bored of the usual item genres, and may want a recommendation off the beaten path. Moreover, some recommendation may simply be too obvious that the user can find it himself even without recommendations. Consider, for example, recommending yet another *Harry Potter* series movie to someone who has already owned a full set of it. The effectiveness of such suggestions is questionable.

To handle this issue, other aspects of recommendation quality should also be taken into account. Indeed, a number of alternative approaches have been proposed in recent years; among them are novelty, diversity and serendipity. The novelty-based (distance-based novelty) approaches consider the newness of the item from the users' prospective. In practice, this is often modeled as the level of dissimilarly (i.e., the distance) between an item and the user's profile. The diversity-based approaches, as the name suggests, aim at providing a diversified list of recommendations to the users. Two main approaches were proposed, namely intra-list diversity, which deals with diversity within a list of recommended items, and aggregate diversity, which deals with the overall diversity across all users. Both novelty and diversity can provide recommendations outside the usual item genres that are previously favored by the user. However, one can argue that either approach, when working on its own, may not necessarily lead to useful recommendation. In light of this, a number of researches have turned to the concept of serendipity. Intuitively, a serendipitous event is one that will result in a pleasant surprise. Serendipitous recommendation algorithms thus aim at providing items which are both unexpected and useful to the users [8]. A good serendipitous recommendation system not only broadens the user's choices (since serendipitous recommendations do not restrict themselves to items similar to the user profiles or the popular items), but also provides a valuable tool for e-retailers to cross-sell their off-the-beaten-track products as well.

In this paper, we propose a scheme for making serendipitous recommendations that are both unexpected and useful to users. First, in order to model unexpectedness, two factors are considered, namely, item rareness and item dissimilarity from the user profile. The rationales are as follows. Recommending popular items will result in low unexpectedness because these items are likely to be well known to the users already, and therefore would bring little surprises even if they may fit a user's profile. Similarly, recommending items that are similar to a user's profiles may result in items already familiar to the user (for instance, a sequel to a user's favorite movie). In both cases, it is likely that the user can find the items easily even without recommendation system. In contrast, less popular items or the items not similar to the user profile would provide higher level of unexpectedness to the user.

Yet, unexpectedness alone is still not sufficient to make the user feel serendipitous. To achieve this goal, one also need to ensure that the recommendations are useful and favored by the user. To model usefulness, we adopted a PureSVD

model. PureSVD is a latent-factor-model based collaborative filtering algorithm that is able to provide high quality recommendation. In this work, the scores for unexpectedness are introduced into the utility model, which forms the basis of our recommendation. The result is a list of items that are not only unexpected to the user but also useful to them as well.

The contributions of this paper are as follow. Firstly, we propose a novel serendipitous recommendation algorithm by considering both unexpectedness and usefulness of the recommended items. Secondly, we also provided a formal model of unexpectedness based on two factors, namely, item-rareness and an item's distance from the user profile. Finally, we provide detailed experimental comparisons with other well-known serendipity-oriented algorithms as well as other baseline methods. Experiments showed that our proposed scheme achieves superior results not only in terms of serendipity, but also lead in other important metrics as well, including precision, intra-list diversity and aggregate diversity.

The rest of the paper is organized as follows: Section 2 reviews the related works and the relevant concepts. Section 3 presents the proposed scheme. Experiment design and results are shown in Section 4. Finally, conclusion is given in Section 5.

2 Related Works

While many works on recommendation systems focused mainly on improving recommendation accuracy [4, 23, 22], some researchers [17, 8, 9, 18, 5] have argued that accuracy alone is not sufficient in evaluating recommendation quality. Instead, several new concepts have been proposed recently, namely, diversity, unexpectedness and serendipity.

The first related concept is diversity. There are two main approaches, namely, aggregate diversity and intra-list diversity. Intra-list diversity [25, 10] refers to the difference between each pair of items in a recommendation list. In [25], Ziegler et al. proposed a scheme for improving the intra-list diversity by diversifying the topic of the recommendations. In [10], Zhang et al. proposed a method which optimizes both accuracy and diversity. However, intra-list diversity does not necessarily give rise to serendipity. For example, providing a user with a list of movies of various genres (e.g. animation, adventure and action) would certainly increase the intra-list diversity. Yet, such a recommendation could still be similar to the user's previous choices if the user has watched all these types of movies. Such a list would still have high diversity and reasonable accuracy, but would not surprise the user. Different from intra-list diversity, aggregate diversity [3, 2] measures the total number of distinct recommended items across all users. For example, Adomavicius et al. [3] argued that many recommendation algorithms tend to have a bias toward the more popular items because those items have more historical data (i.e., user ratings) and hence they would be recommended more frequently. As pointed out by the authors, such kind of recommendations would reduce the aggregate diversity because the same pieces of popular items would tend to be recommended to multiple users. To solve this issue, several

re-ranking methods have been proposed. This includes, for example, a method that ranks the recommendations, where the predicted ratings are higher than a certain threshold, in reverse to their popularity. It was then argued that a high aggregate diversity could help to expand the user's horizon because the recommendations would not be restricted to the popular items. Moreover, it would also be beneficial to the merchants because they can profit from not only the popular items but also from the 'long-tail' items (the items located in the tail of the sales distribution). However, despite the claimed benefits, aggregate diversity is not a replacement for serendipity. A list of recommendations with high aggregate diversity, which provides many distinct items across a large group of users, does not necessarily provide items that are both unexpected and useful to the individuals. Moreover, aggregate diversity is a measurement calculated across from all users, while the serendipity is calculated for individual users.

Another related concept is unexpectedness. In [1], Adamopoulos et al. summarized various proposed definitions of unexpectedness, including associating unexpectedness to the prior background knowledge of decision makers [20], and measuring unexpectedness by taking multi-f acets (frequent itemsets, tiles, association rule and classification rule) into account. In [1], Adamopoulos et al. argued that unexpectedness should consider the expectation of users, where unexpectedness is obtained by generating the recommendation significantly depart from the user expectedness. In [8], Ge et al. discussed that unexpected recommendation could be viewed as the recommendations which do not belong to the primitive prediction model. Although various definition of unexpectedness are proposed, unexpectedness is not equal to serendipity. According to Ge et al. [8], unexpectedness is one of the most important components of serendipity.

Serendipity differs from diversity and unexpectedness in that it attempts to model the users' level of positive surprise toward the items. Literally speaking, the word serendipity denotes a pleasant surprise, or a fortunate yet unexpected discovery by chance. Thus, a serendipitous discovery should be unexpected, yet useful. This idea has been explored by a number of works. For example, in [8], Ge et al. discussed the idea that serendipity should cover unexpectedness (i.e., items that are not yet discovered and unexpected by the user) and usefulness (i.e., items that are of interest to the user). However, unexpectedness and usefulness are not clearly defined in this work. Some other works have further elaborated the definition of unexpectedness and usefulness. In [1], Adamopoulos et al. adopted the definition of usefulness of an item that is determined by its average rating: if its average rating among the users is larger than a certain threshold, then it is considered to be useful for all the users. In [21] usefulness of an item is determined by the user's rating for the recommended item. In this work, we follow a definition of usefulness similar to an approach adopted in [21], which provides personalized usefulness estimation and better reflects real life situation. Regarding unexpectedness, it was defined differently in [1] and [21]. In [1], Adamopoulos et al. proposed that high unexpectedness can be obtained by recommending items that are different from the set of the expected items. In [21], Lu et al. argued that since the popular items are so well-known, they are

easy to find, hence they would lead to low unexpectedness. Partially inspired by these works, we argue in this paper that the unexpectedness of an item should be associated with both the item's popularity (or rareness) and the item's level of dissimilarity (i.e., its distance) from the user profile.

3 The Proposed Scheme

In this section, we formally present our proposed scheme based on the ideas developed in the previous section. First, the unexpectedness of an item is defined in Section 3.1. After that, item utility is presented in Section 3.2. And finally, our optimization process is described in Section 3.3.

3.1 Unexpectedness

As explained in Section 2, the unexpectedness of an item depends on two factors: the item popularity (or rareness) and the item's dissimilarity from the user profile. Firstly, regarding item popularity, the more popular is an item, the lower is its unexpectedness for a user because such items would be so well known that user could find them easily even without recommendations. This idea is implemented in Eq.1, where $Pop(i)$ denotes the number of users who have selected item i, and $|U|$ is the number of all users.

$$Unexpectedness(u,i) \propto 1 - \frac{Pop(i)}{|U|} \tag{1}$$

Secondly, regarding an item's dissimilarity from the user profile, the concept is depicted in Eq.2, where $S(u)$ is the set of items chosen by u (the user), and *diff(i,j)* denotes the degree of dissimilarity between item i and item j (see below). As seen from Eq.2, the dissimilarity of an item i to a user u is high if i is different from the other items that s/he has chosen before.

$$Unexpectedness(u,i) \propto \frac{\sum_{j \in S(u)} \textit{diff(i,j)}}{|S(u)|} \tag{2}$$

The dissimilarity function *diff(i,j)* can be obtained by $\textit{diff(i,j)} = 1 - sim(i,j)$, where $sim(i,j)$ denotes the degree of the similarity between i and j. In the literature, there are various possible ways for computing the similarity between a pair of items, including both content dependent [4] and content independent metrics [23]. In this paper we adopt a content independent metric [23] for our similarity function, which is illustrated in Eq.3, where $S(i,j)$ is the set of users (co-rated users) who have chosen both item i and item j, $r_{u,i}$ is user u's rating for item i, and $\overline{r_u}$ is the average rating of u for his rated items. Basically, Eq.3 measures the rating consistency among the co-rated users on i and j. If the co-rated users consistently give high (or low) ratings to both i and j, it would indicate that i and j are similar.

$$sim(i,j) = \frac{\sum\limits_{u \in S(i,j)} (r_{u,i} - \overline{r_u}) \cdot (r_{u,j} - \overline{r_u})}{\sqrt{\sum\limits_{u \in S(i,j)} (r_{u,i} - \overline{r_u})^2} \sqrt{\sum\limits_{u \in S(i,j)} (r_{u,j} - \overline{r_u})^2}} \tag{3}$$

Finally, the unexpectedness of an item i to user u is defined as a linear combination of the item's rareness and its dissimilarity from the user's profile:

$$Unexpectedness(u,i) = (1 - \frac{Pop(i)}{|U|}) + \frac{\sum_{j \in S(u)} diff(i,j)}{|S(u)|} \tag{4}$$

3.2 Utility

Recommending an item based solely on unexpectedness may lead to a risk. That is, the items could be too unexpected and deviate too far from the user's interest. In either case, the user's trust and satisfaction for the system would decrease. Hence, in addition to unexpectedness, we must also consider the utility of the recommendation items. Utility measures an item's relevance and usefulness to the user. In practice, utility is usually measured by the predicted rating for an item by a given user. To predict an item's utility, we apply latent factor models which are good at predicting items utility and perform well in capturing user future interests. The model we adopt in this paper is PureSVD [7].

The majority of latent factor models are based on the factorization of the user-item rating matrix by Singular Value Decomposition (SVD). The main idea of SVD models is to factorize the user-item rating matrix into three low rank matrices (Eq.5). U is $n \times k$ orthonormal matrix, Q is $m \times k$ orthonormal matrix and Σ is $k \times k$ diagonal matrix with the top k singular values. k is the number of latent factors. Alternatively, $\hat{R}$ can be represented by Eq.6. $\hat{R}$ is the estimated utility matrix.

$$\hat{R} = U \cdot \Sigma \cdot Q^T \tag{5}$$

$$\hat{R} = P \cdot Q^T \tag{6}$$

After factorization, each user is associated with a k-d vector p_u, representing the user u's preference for k factors. And each item is also associated with a k-d vector q_i, describing i's importance weight for k factors. The number of latent factors (k) is 50 in this paper. PureSVD is a standard latent factor model that measures the utility between i (the item) and u (the user) by the product of user-factor vector p_u and item-factor vector q_i (Eq.7).

$$Utility(u,i) = p_u \cdot {q_i}^T \tag{7}$$

3.3 Optimization

In recommendation systems, the utility of an item refers to the attractiveness of the item to a user. In practice, this is often estimated based on the observed user-item ratings. In order to predict the ratings accurately, a model must first be trained based on known historical data. A typical approach is shown in Eq.8. Here, $r(u,i)$ is the observed rating of user u for item i, and $\lambda(\|p_u\|^2 + \|q_i\|^2)$ is a regularizing term to prevent overfitting.

$$min\sum_{u}\sum_{i\in S(u)}(r(u,i) - p_u \cdot {q_i}^T)^2 + \lambda(\|p_u\|^2 + \|q_i\|^2) \tag{8}$$

Eq.8 illustrates the traditional approach, where unexpectedness is not considered. In order to take serendipity into account, we can employ a weight $w_{ui} = Unexpectedness(u,i)$ for penalizing items that are popular and similar to the user's profile. Also, note that Eq.8 only optimizes the errors on the observed items ($i \in S(u)$). However, as pointed out by [24], both the unobserved and observed items contribute to recommendation accuracy (e.g., the top n recommendation accuracy). In light of this, Eq.8 has been readapted accordingly to include all items. The revised version is shown in Eq.9 and the corresponding learning process is depicted in algorithm 1, where γ is the learning rate. For distinguish purpose, we used $\tilde{r}(u,i)$ instead of $r(u,i)$, where $\tilde{r}(u,i)$ represents both observed and unobserved ratings. Our proposed method is simple to implement, and can easily be applied to real life e-commerce systems. Most importantly, experimental results indicate that the proposal method performs well in both accuracy and diversity. The detailed findings will be presented in the next section.

$$min\sum_{u}\sum_{i\in I}(\tilde{r}(u,i) - p_u \cdot {q_i}^T)^2 \cdot w_{ui} + \lambda(\|p_u\|^2 + \|q_i\|^2) \tag{9}$$

Algorithm 1 Update of p_u and q_i

for $u \in U$ **do**
  for $i \in I$ **do**
    $err(u,i) = (\tilde{r}(u,i) - p_u \cdot {q_i}^T) \cdot w_{ui}$
    $p_u \leftarrow p_u + \gamma(err(u,i) \cdot q_i - \lambda \cdot p_u)$
    $q_i \leftarrow q_i + \gamma(err(u,i) \cdot p_u - \lambda \cdot q_i)$
  end for
end for

4 Experiment

To evaluate our proposed method, we conducted a series of experiments on two representative datasets. In Section 4.1, the adopted datasets are first introduced.

Experiment setup is presented in Section 4.2. Finally, experiment results for our scheme as well as those of representative approaches are discussed in Section 4.3.

4.1 Datasets

Two representative datasets were chosen to evaluate our proposed scheme, namely, Netflix [11] and Movielens [6]. Both datasets contain user rating data collected over long periods and they are both widely used for evaluation in the literature. There are 2,113 users, 10,197 items and more than 800k ratings in Movielens dataset, dating from October 1997 to December 2008. Its sparsity is about 3.976%. The original Netflix data set contains over 17k items, 480k users and 100M ratings dated from 1997 to 2008. For the sake of scalability and comparability, we randomly sampled the ratings from 2000 users from the original Netflix dataset. The resulting dataset contains 5,260 items and 632,335 ratings. The statistical properties of these two datasets are summarized in Table.1.

Table 1: Statistical properties of two datasets

	# of users	# of items	# of ratings	Sparsity
Movielens	2,113	10,197	800k	3.976%
Netflix	2,000	5,260	635k	6.01%

4.2 Experiment Setup

Each dataset was split into two disjoint sets chronologically, with the older data in the training set and the remaining data in the test set. Recommendations were generated based on the training set. Each user was provided with 10 lists of recommendations, with size of 10, 20,..., and 100 items respectively. The value of α is 0.5.

A number of metrics were employed to evaluate the recommendation quality. The first one was accuracy. Two accuracy metrics were adopted, namely, precision and recall, which are defined by Eq.10 and 11. Here, $RS(u, N)$ represents the top N recommendations in the recommendation list of user u. $TestSet(u)$ is the set of items in the test set that are chosen by user u. The precision metric measures the proportion of recommendations among the recommendation list which are actually selected by the users (the proportion of hit items). Recall measures the proportion of the recommendations which are actually selected by the users among the items relevant to the users.

$$Prec@N = \frac{\sum_u |RS(u, N) \cap TestSet(u)|}{N \cdot |U|} \tag{10}$$

$$Recall@N = \frac{\sum_u |RS(u, N) \cap TestSet(u)|}{|TestSet(u)| \cdot |U|} \tag{11}$$

The second evaluation metric is serendipity (Eq.13). Later in this section, we shall present our experimental evaluation results alongside with a number of representative approaches, including two other serendipity based models. In order to provide fair and meaningful comparisons, we decided to adopt the top N serendipity metric that has also been utilized by these benchmark approaches for evaluation purpose [8, 1, 21]. The top N serendipity metric is in some way similar to precision, except that it is stricter. In precision, one only counts the number of hit items in a recommendation list. In a serendipity-oriented metric such as the top N serendipity, on the other hand, one also needs to determine whether the hit items are unexpected and useful for the user. Recall from previous section that serendipity depends on two factors, namely unexpectedness and usefulness. To evaluate unexpectedness, a model (Predictive Model (PM)) consisting of a set of items which are assumed to be expected for the users is first constructed. And any recommended items that are not included in the set of recommendations generated by the Predictive Model (PM) is treated as the unexpected ones. The concept is illustrated in Eq.12. Following the practice of [1, 21], the set of expected items generated by PM consists of 100 items, which includes the top 50 items with highest average rating and the top 50 items with highest popularity value. To measure the usefulness of the recommendations, we observe whether the user selects the recommended item and favors it (i.e., gives it a high rating). In this metric, the set of high-rating-items are those items with rating larger than a given threshold θ. The set of useful items is then defined as $USEFUL(u) = \{i \in TestSet(u) | r(u,i) > \theta\}$, where θ is the threshold rating. In our experiment, the ratings of two adopted datasets have a range of zero to five, and θ's value is 3.

$$UNEXP(u,N) = RS(u,N) \backslash PM \tag{12}$$

$$SRDP@N(u) = \sum_{u} \frac{|UNEXP(u,N) \cap USEFUL(u)|}{N \cdot |U|} \times 100\% \tag{13}$$

The third metric is intra-list diversity. The calculation is shown in (Eq.14), where *diff(i,j)* is the dissimilarity between item i and item j. We adopted a content-independent metric [10] to calculate the (dis)-similarity between any two items, which is illustrated by Eq.4. The difference function *diff(i,j)* is then obtained by $1 - sim(i,j)$. (A point of note, the dissimilarity in intra-list diversity is not the same as the dissimilarity that is used to compute unexpectedness. Intra-list diversity measures the difference between each pair of items in the recommendation list, while unexpectedness concerns the difference between the candidate recommended item and the user's previous chosen items.)

$$\begin{aligned} IntralistDiversity@N = \frac{1}{|U| \cdot N(N-1)} \cdot \\ \sum_{u} \sum_{i \in RS(u,N)} \sum_{j \neq i \in RS(u,N)} diff(i,j) \end{aligned} \tag{14}$$

A related metric is the aggregate diversity [3], which measures the number of distinct items recommended across all users (Eq.15). A high aggregate diversity in recommendation is beneficial to the e-retailer since it indicates that more distinct items are recommended to the users, thus increases the sale potential.

$$AggDiversity@N = |\cup_{u \in U} RS(u, N)| \tag{15}$$

To evaluate our scheme, we compared the performance of our method with a number of representative methods. Four schemes for making personalized recommendations are included in this study, including two latent factors models, namely SVD (bias) and SVDpp [7, 13, 14] and two serendipitous recommendation algorithms, namely Adamopoulos's method [1] and Lu's method [21]. The latent factor based models have gained a lot of attentions in RS because of their significant performance in top N recommendations. Adamopoulos's method and Lu's method are both two representative serendipitous recommendation algorithms, whose performance in making serendipitous recommendations has been demonstrated. Apart from these representative methods, we also implemented three other non-personalized approaches to serve as benchmark algorithms, namely, AvgRating, Random and Toppop. AvgRating recommends the items which have the highest average ratings to the user. Random uses a random algorithm to recommends the non-chosen items to the users. Toppop recommends the most popular items to the users.

4.3 Experiment Results

Accuracy Performance In this section, we will show the comparison of our method and other baseline methods in top n accuracy. Figure 1 shows the top N precision of various methods on Movielens and Netflix datasets, Figure 2 shows the recall. The performance of Random and Avgrating turned out to be very close in this case. Thus, for clarity, only the Random method is shown.

Several observations can be made. Firstly, all personalized algorithms performed better than the non-personalized benchmark methods. Secondly, among the personalized methods, latent-factor-based methods (which include our method, Lu's method, SVDpp and SVD(bias)) produced the best performance. (A side note, interested readers may refer to [7] for a detailed discussion on the effectiveness of the latent factor models). Thirdly and most importantly, our method performed the best on both datasets. For example, in Movielens dataset, our method achieved a 10% improvement in precision over the second best method (Lu's method), and up to 50% better than the third best method (SVD (bias)) for $N = 10$. Similar results can be observed from Netflix dataset. Results of other metrics also support similar conclusions. We attribute the good top n accuracy performance of our method to two reasons. The first one is the adopted utility model-PureSVD. It is reported that PureSVD performs well in top n accuracy [7]. The second reason is that, as mentioned in Section 3.3, our method explicitly models both observed and unobserved data. According to the work of Steck [24], in applications where the data are not *missing at random* (in the context

of recommendation systems, *missing at random* means that the probability of a rating to be missing does not depend on its value), the missing data may contain hidden implications (e.g., many users simply do not provide ratings for the movies they do not like). Such missing ratings have to be modeled as to obtain better results, and they are included in our model for this reason.

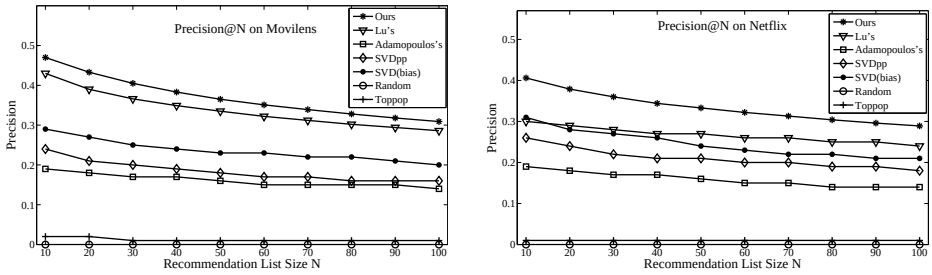

Fig. 1: Comparison of Precision on Movielens and Netflix datasets

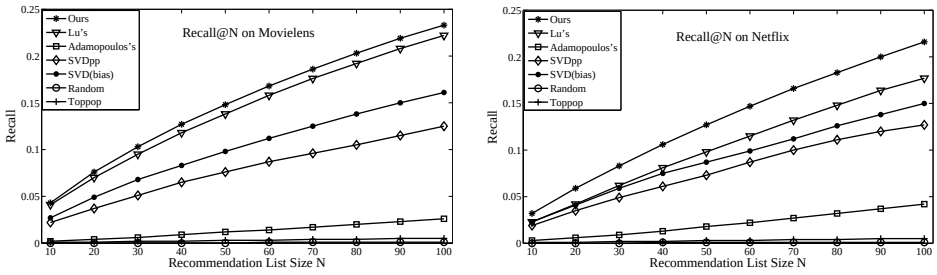

Fig. 2: Comparison of Recall on Movielens and Netflix datasets

Serendipity Performance Next, we evaluated the serendipity performance of the various methods using top N serendipity (Eq.13). The results are shown in Figure 3. Once again the results of Random and Avgrating are very close to each other, so for clarity, only the result for Random is shown in the figure.

From Figure 3, we observe that our method outperforms all other methods on the two datasets in serendipity significantly. For example, in the Movielens dataset, for $N = 100$, our method led the second best method (Lu's method [21]) by 32%, and third best method (SVD(bias)) by 60%. For smaller Ns, the difference is even larger (for instance, our score for $N = 10$ is 2.57 times of that of the second highest method). Similar results can be observed from Netflix dataset. For example, when $N = 50$, our performance was 76% higher than the second best method (SVD(bias)). This result is very encouraging because normally one

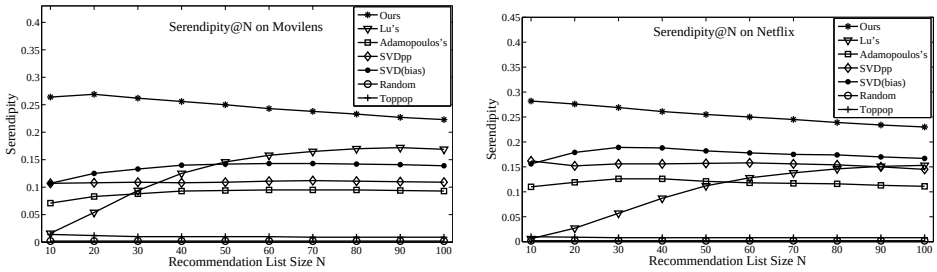

Fig. 3: Comparison of Serendipity on Movielens and Netflix datasets

would expect a method that does well in precision would not necessarily achieve high scores in serendipity. The reason is that by offering off-the-beaten-track recommendations, one would think that the precision would suffer because the most popular and straightforward (and hence "safe") recommendations are now excluded. Yet, our results seem to suggest that it is not necessarily the case, and it is possible to achieve good precision and good serendipity by considering both utility and unexpectedness in the framework.

Additionally, several other observations can be made from Figure 3. Firstly, as in precision, personalized methods performed better than non-personalized methods. Secondly, among the personalized methods, it is interesting to note that the two non-serendipitous methods (SVD(bias) and SVDpp) actually performed quite well in serendipity (for example, in the Netflix dataset, the two non-serendipitous methods actually outperformed the remaining two serendipity-oriented approaches for all $N < 90$, although this was not the case in the Movielens dataset). Thirdly, in all approaches except Lu's method [21], the top N serendipity scores were quite stable for different values of N, whereas in Lu's method, it started at a low value but increased as the recommendation list size grew. The reason may be that in Lu's method, the recommendation lists mainly consist of popular or highly-rated items when N is small, which led to low unexpectedness values and resulted in low serendipity. Overall, our method has produced the best performance for all list sizes.

Diversity Performance Finally, we measured the diversity performance of the various methods. In previous studies, it has been suggested that diversity is achieved at the expense of accuracy [3, 25]. However, we argue that accuracy should be the premise of diversity. A list of recommendations achieving high diversity but low accuracy would indicate that recommendations are diverse but do not fit the user's preference. For this reason, we only further evaluated the diversities of the three methods that produced highest accuracy, namely our method, Lu's method [21] and SVD(bias). The results for both intra-list and aggregate diversity are shown in Tables 2-5.

From the results, we see that our method achieved the highest diversity (both intra-list and aggregate) among the methods that produced the highest

Table 2: Intra list diversity on Movielens dataset

N	10	20	30	40	50	60	70	80	90	100
Ours	**.768**	**.807**	**.831**	**.847**	**.861**	**.871**	**.879**	**.887**	**.893**	**.899**
Lu's Method	.674	.744	.779	.812	.832	.846	.861	.870	.880	.887
SVD(bias)	.434	.479	.501	.520	.539	.555	.569	.583	.593	.602

Table 3: Intra list diversity on Netflix dataset

N	10	20	30	40	50	60	70	80	90	100
Ours	**.909**	**.933**	**.945**	**.954**	**.961**	**.966**	**.970**	**.974**	**.977**	**.980**
Lu's Method	.881	.909	.918	.930	.940	.950	.958	.964	.967	.970
SVD(bias)	.474	.551	.613	.650	.669	.686	.700	.714	.726	.737

Table 4: Aggregate diversity on Movielens dataset

N	10	20	30	40	50	60	70	80	90	100
Ours	**769**	**1036**	**1244**	**1385**	**1511**	**1637**	**1755**	**1865**	**1961**	**2049**
Lu's Method	135	200	257	308	351	395	435	468	503	535
SVD(bias)	83	124	155	180	206	237	258	280	299	321

Table 5: Aggregate list diversity on Netflix dataset

N	10	20	30	40	50	60	70	80	90	100
Ours	**656**	**856**	**985**	**1103**	**1187**	**1259**	**1339**	**1406**	**1468**	**1533**
Lu's Method	88	138	177	216	250	284	314	342	374	404
SVD(bias)	66	102	140	174	208	237	259	282	313	338

accuracy. For example, for intra-list diversity, our method outperformed Lu's method by 14% and SVD(bias) by 77% on the Movielens dataset when N is 10 (Table 2). For aggregate diversity, we obtained values that were more than three times of those obtained by the other two methods for both Netflix dataset (Table 5) and Movielens dataset (Table 4) when $N = 100$. The difference was even more significant for smaller Ns. This is quite remarkable as our method is not primarily designed to improve diversity. The good results can be explained as follows. Recall that our model for unexpectedness consists of two components, namely item (un)popularity and dissimilarity from the user profile. According to the work of Adomavicius [3], recommending less popular items results in higher aggregate diversity, which helps to explain our results in Table 4 and 5. Also, regarding intra-list diversity, recommending the items different from the user profile means that system provides more diverse recommendations that are not restricted to items similar to the user profile. As a result, the intra-list diversity increases. Overall, the results suggest that we have a new approach for providing accurate, serendipitous, and diverse recommendations.

5 Conclusions

In this paper, we proposed a recommendation scheme based on serendipity. There are two requirements for a serendipitous recommendation, namely, that the items must be unexpected, and that the items must be useful to the user. There are two elements that constitute unexpectedness in our model. The first element is item rareness. It is likely that popular items are already well-known to the users, who can find them easily even without recommendation. The second element is the level of dissimilarity to the user-profile, as recommending items that are similar to a user's profiles may also result in items already familiar to the user (for instance, a sequel to a user's favorite movie). Usefulness (or utility) refers to the level of attractiveness of an item to a user. In practice, usefulness is often measured indirectly by the predicted user-item-ratings. In this paper, item utility is modeled using a PureSVD latent factor model, which has been demonstrated to perform well in capturing user future interests. The unexpectedness value and the utility are then combined to obtain a serendipitous score of an item. To evaluate the proposed scheme, its performance is compared with two representative serendipitous algorithms and two popular latent factor models using popular benchmark datasets. The obtained results are very encouraging. Experiment results suggested that, our scheme not only achieved the best performance in terms of serendipity, but it also performed well in term of precision and diversity also. This is significant because it has been previously suggested that serendipity and diversity are achieved at a price of accuracy. Our results seem to suggest a new and effective direction in serendipity-oriented recommendation.

References

1. Adamopoulos, P., Tuzhilin, A.: On unexpectedness in recommender systems: Or how to expect the unexpected. In: Proc. RecSys 11, Chicago, IL, USA, October 23-27, pp. 11–18. ACM, New York, NY, USA (2011)
2. Adomavicius, G., Kwon, Y.: Toward more diverse recommendations: Item reranking methods for recommender systems. In: Proc. WITS 09, Phoenix, AZ, USA, December 14-15. SSRN, New York, NY, USA (2009)
3. Adomavicius, G., Kwon, Y.: Improving aggregate recommendation diversity using ranking-based techniques. IEEE Trans. Knowl. Data Eng. **24**(5), 896–911 (2012)
4. Balabanović, M., Shoham, Y.: Fab: content-based, collaborative recommendation. ACM Commun. **40**(3), 66–72 (1997)
5. Benjamin, W., Chandrasegaran, S., Ramanujan, D., Elmqvist, N., Vishwanathan, S., Ramani, K.: Juxtapoze: Supporting serendipity and creative expression in clipart compositions. In: Proc. CHI '14, Tronto, Canada, April 26-May 1, pp. 341–350. ACM, New York, NY, USA (2014)
6. Cantador, I., Brusilovsky, P., Kuflik, T.: 2nd workshop on information heterogeneity and fusion in recommender systems. In: Proc. RecSys 11, Chicago, IL, USA, October 23-27. ACM, New York, NY, USA (2011)
7. Cremonesi, P., Koren, Y., Turrin, R.: Performance of recommender algorithms on top-n recommendation tasks. In: Proc. RecSys 10, Barcelona, Spain, September 26-30, pp. 39–46. ACM, New York, NY, USA (2010)

8. Ge, M., Delgado-Battenfeld, C., Jannach, D.: Beyond accuracy: evaluating recommender systems by coverage and serendipity. In: Proc. RecSys 10, Barcelona, Spain, September 26-30, pp. 257–260. ACM, New York, NY, USA (2010)
9. Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating collaborative filtering recommender systems. ACM Trans. Inf. Syst. **22**(1), 5–53 (2004)
10. Hurley, N., Zhang, M.: Novelty and diversity in top-n recommendation – analysis and evaluation. ACM Trans. Internet Technol. **10**(4) (2011)
11. J.Bennett, Lanning, S.: The netflix prize. In: Proc. KDD Cup and Workshop, California, USA, August 12. ACM, New York, NY, USA (2007)
12. Kim, H.N., Saddik, A.E., Jung, J.G.: Leveraging personal photos to inferring friendships in social network services. Expert Syst. Appl. **39**(8), 6955 – 6966 (2012)
13. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: Proc. SIGKDD 08, Las Vegas, NV, USA, August 24-27, pp. 426–434. ACM, New York, NY, USA (2008)
14. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer **42**(8), 30–37 (2009)
15. Linden, G., Smith, B., York, J.: Amazon.com recommendations: item-to-item collaborative filtering. IEEE Internet Computing **7**(1), 76–80 (2003)
16. Lucas, J.P., Luz, N., Moreno, M.N., Anacleto, R., Almeida Figueiredo, A., Martins, C.: A hybrid recommendation approach for a tourism system. Expert Syst. Appl. **40**(9), 3532 – 3550 (2013)
17. McNee, S.M., Riedl, J., Konstan, J.A.: Being accurate is not enough: how accuracy metrics have hurt recommender systems. In: Proc. CHI EA '06, Montréal, Canada, April 22-27, pp. 1097–1101. ACM, New York, NY, USA (2006)
18. Murakami, T., Mori, K., Orihara, R.: Metrics for evaluating the serendipity of recommendation lists. In: Proc. JSAI 07, Miyazaki, Japan, June 18-22, pp. 40–46. Springer-Verlag, Berlin, Heidelberg (2008)
19. Özbal, G., Karaman, H., Alpaslan, F.N.: A content-boosted collaborative filtering approach for movie recommendation based on local and global similarity and missing data prediction. Comp. J. **54**(9), 1535–1546 (2011)
20. Padmanabhan, B., Tuzhilin, A.: A belief-driven method for discovering unexpected patterns. In: Proc. SIGKDD 98, New York, NY, USA, August 27-31, pp. 94–100. AAAI Press, California, USA (1998)
21. Qiuxia, L., Chen, T., Zhang, W., Yang, D., Yu, Y.: Serendipitous personalized ranking for top-n recommendation. In: Proc. WI'12, Macau, China, December 4-7, pp. 258–264. IEEE Computer Society, Washington, DC, USA (2012)
22. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: Grouplens: an open architecture for collaborative filtering of netnews. In: Proc. CSCW 94, Chapel Hill, North Carolina, USA, October 22-26, pp. 175–186. ACM, New York, NY, USA (1994)
23. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Item-based collaborative filtering recommendation algorithms. In: Proc. WWW 01, Hong Kong, China, May 1-5, pp. 285–295. ACM, New York, NY, USA (2001)
24. Steck, H.: Training and testing of recommender systems on data missing not at random. In: Proc. SIGKDD 10, Washington DC, DC, USA, July 25-28, pp. 713–722. ACM, New York, NY, USA (2010)
25. Ziegler, C.N., McNee, S.M., Konstan, J.A., Lausen, G.: Improving recommendation lists through topic diversification. In: Proc. WWW 05, Chiba, Japan, May 10-14, pp. 22–32. ACM, New York, NY, USA (2005)