Attribute-based evaluation for recommender systems: incorporating user and item attributes in evaluation metrics

Pablo Sánchez
pablo.sanchezp@uam.es
Information Retrieval Group
Universidad Autónoma de Madrid
Madrid, Spain

Alejandro Bellogín
alejandro.bellogin@uam.es
Information Retrieval Group
Universidad Autónoma de Madrid
Madrid, Spain

ABSTRACT

Research in Recommender Systems evaluation remains critical to study the efficiency of developed algorithms. Even if different aspects have been addressed and some of its shortcomings – such as biases, robustness, or cold start – have been analyzed and solutions or guidelines have been proposed, there are still some gaps that need to be further investigated. At the same time, the increasing amount of data collected by most recommender systems allows to gather valuable information from users and items which is being neglected by classical offline evaluation metrics. In this work, we integrate such information into the evaluation process in two complementary ways: on the one hand, we aggregate any evaluation metric according to the groups defined by the user attributes, and, on the other hand, we exploit item attributes to consider some recommended items as surrogates of those interacted by the user, with a proper penalization. Our results evidence that this novel evaluation methodology allows to capture different nuances of the algorithms performance, inherent biases in the data, and even fairness of the recommendations.

1 INTRODUCTION

Since the emergence of Recommender Systems (RS) their main objective has remained the same: learn the tastes and needs of the users who access a particular system in order to be able to retrieve items that are hypothetically interesting for them. Ongoing research in this area has led to the emergence of a vast number of recommendation strategies, from trivial algorithms such as popularity, through Collaborative Filtering models to probabilistic and deep learning strategies. At the same time, the way of evaluating the performance of this type of systems has changed over the years. Although the Netflix prize made error metrics such as MAE or RMSE popular, they have now been displaced by classic Information Retrieval metrics such as Precision, Recall, or NDCG [7]. Besides, for a few years now, it has been alerted the necessity of evaluating other aspects of the recommendations, such as novelty, diversity, or freshness [4].

However, despite all these advances there are still some gaps that need to be addressed. Firstly, when analyzing the evaluation results we tend to treat all users equally, ignoring the specific underlying aspects of each user profile. This is something not entirely new, as it is easy to observe that in many datasets there are users that are more difficult to satisfy (e.g., they do not have a mature history of ratings, or they are simply different to the rest of the community, the so-called *gray sheep* effect). Furthermore, when evaluating the relevance of the recommendations, we normally only take into account those items the user has somehow interacted with, either by liking, rating, or clicking on them, ignoring the rest of recommended items, and, thus, considering them as not relevant, which may have a strong impact on the evaluation hypotheses and obtained results [2]. In some domains, like point-of-interest recommendation, this assumption may impose constraints too difficult to satisfy by the algorithms due to the high sparsity of the data, and as a solution to this issue some researchers decided to measure matchings between the item attributes (categories) in the test set and the recommended ones, instead of the actual items [3, 9, 14, 15].

With these ideas in mind, we aim to delve into some aspects of the evaluation of RS that are generally neglected in traditional evaluation. On the one hand, we define a way to generalize classic ranking-based metrics in order to consider as "partially relevant" those items that, although not specifically stated as relevant, are highly similar to the ones the user liked. On the other hand, we formalize the notion of aggregating the evaluation metric values at the user level by assigning users into different groups according to the available attributes (such as age, gender, or their consumption history), which would help us to detect if a recommendation algorithm makes more correct recommendations to users belonging to some specific groups.

2 ATTRIBUTE-BASED EVALUATION

It is generally acknowledged that a common formulation for many evaluation metrics is the following (arithmetic) mean:

$$m(r_u) = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} m(r_u, u) \tag{1}$$

where $m(r_u)$ represents the value of a metric m on the output of some recommendation algorithm in the form of the list r_u, and $m(r_u, u)$ is the user-level metric value. For example, for precision or $P@k$, this function would be defined as $m(r_u, u) = |\{i \in r_u : i \in Te(u)\}|/k$, considering $Te(u)$ the test set or groundtruth (withheld interactions) of user u, whereas for NDCG, it would take the form of $m(r_u, u) = \sum_{i_j \in r_u} (2^{\text{rel}(i_j, u)} - 1)/\log(j+1)/\text{IDCG}(r_u, Te(u))$, where IDCG represents the ideal ranking for each user.

However, not all users in a recommender system are the same, especially from the system perspective [12]. Some users may have more influence than others – either as power users [10] or influential

RecSys '19, September 16–20, 2019, Copenhagen, Denmark

ACM ISBN 978-1-4503-6243-6/19/09…$15.00
https://doi.org/10.1145/3298689.3347049

in a social network context [20] –, some spend more time in the system, or some could even be easier to satisfy than others [19]. In any case, it is reasonable to assume that, under various settings, the system designer might want to aggregate these user-level metric values according to different schemes. For this, we propose to use a function c that assigns a weight for each user, either based on her behavior in the system or according to her attributes, which is incorporated into Equation 1 as follows:

$$m(r_u) = C^{-1} \sum_{u \in \mathcal{U}} c(u) m(r_u, u) \quad (2)$$

where $C = \sum_u c(u)$. Even though this formulation is reduced to a simple weighted sum of the user-level metric values (note that Equation 1 is recovered by setting $c(u) = 1$), we argue that it consolidates many ad-hoc evaluations performed in the literature: for instance, cold-start evaluation and recommendation fairness can be recovered by setting binary weights on function c such that only the cold-start users or those belonging to a specific group are aggregated; additionally, typical filters applied to the data, such as ignoring users with low ratings in training or test, could be modeled under this formulation by setting the appropriate function c. These examples will be considered later as use cases in the experiments.

Now that we have shown how to incorporate user attributes in any evaluation metric, we shall show how we can do the same with the item attributes by exploiting some concept of similarity in the evaluation metrics. This idea is not completely new, since most of the works on diversity evaluation use some concept of similarity metric [4], the main difference is that in those cases such metric is computed within the recommendation list, to analyze how similar the recommended items are with respect to each other. Moreover, our proposal is inspired by the works of [3, 9], where the item categories instead of the actual items are compared as groundtruth. It should be noted that in a very recently published work, the authors performed a user study where they analyzed the recommended items that were not included in the groundtruth, and found that those items highly similar to those selected by the user were perceived as *acceptable recommendations* [6], hence, validating our proposal.

We define the following three sets of items based on those items included in the recommendation list r_u, the items in the groundtruth of user u ($Te(u)$), and a given item similarity metric $\text{sim}_F(i,j)$ over the feature space F: $I^+(u)$ are those items explicitly interacted by the user and included in the test set, i.e., $I^+(u) = \{i \in r_u : i \in Te(u)\}$; $I^*(u)$ is formed by those items that show a non-zero similarity, $I^*(u) = \{i \in r_u \wedge i \notin I^+(u) : \exists j \in Te^*(u), \text{sim}_F(i,j) > 0\}$, where $Te^*(u) = Te(u) \cap \overline{I^+(u)}$, that is, the subset of the user test that was not recommended; and, finally, the set $I^-(u)$ is formed by those items not included in the previous two sets. Now, taking the formulation previously introduced in Equation 1, we integrate and exploit these sets of items when computing the item-level metric value as follows:

$$m(r_u, u) \propto \sum_{i \in I^+(u)} w^+(u,i) + \sum_{i \in I^*(u)} w^*(u,i) + \sum_{i \in I^-(u)} w^-(u,i) \quad (3)$$

where each w^+, w^*, and w^- are properly adjusted weights; typically, $w^+(u,i) = 1$ and $w^-(u,i) = 0$, we propose to define w^* as a function τ that depends on the similarity with respect to the closest item not interacted by the user, that is: $w^*(u,i) = \tau(\text{sim}^*_F(i,j;\alpha))$, where $\text{sim}^*_F(i,j;\alpha) = \max_{j \in Te^*(u)} \alpha \cdot \text{sim}_F(i,j)$, where we use a penalization weight α. Note that this item set could use as many similarity metrics as desired, associating each similarity with a different penalization; for instance, as we shall show in the experiments, we may consider two items as similar when they share the same director or the same genre, but obviously the penalization should be lower in the latter case. It is now straightforward to obtain those metrics defined in the literature where authors matched items at the category level [3, 9]: by simply setting $w^+(u,i) = w^*(u,i) = 1, w^-(u,i) = 0$ and taking function $\text{sim}_F(i,j)$ as the binary mapping that outputs 1 if two items share the same category and 0 otherwise. We should take care, however, when we create item set $I^*(u)$ that each item j found to have the highest similarity with respect to a recommended item i, is only considered once; this can be achieved by iterating through the recommendation list in order and removing the items from $Te^*(u)$ once they are used.

We want to emphasize that our proposal does not extend the actual groundtruth being exploited, as the number of relevant items is still limited by the items found in the test set of the user.

3 EXPERIMENTS AND RESULTS

3.1 Datasets description

We have conducted our experiments in two datasets: Movielens1M [8] (1M ratings by 6K users in 3.7K items) and Foursquare (dataset provided by the authors of [21], with 33M interactions by 267k users on 3.7M venues). The reason we selected these datasets is twofold: firstly, in addition to having the preferences of the users, we also have additional information (e.g., age and gender for Movielens and gender for Foursquare) and both datasets belong to different contexts (movies and venues respectively). Moreover, for Foursquare we have only worked with the New York check-ins since it is common in the POI recommendation area to work with each city as if it was an independent dataset, also New York is often analyzed in many articles due to its strong tourist component. Besides, we removed all repetitions (a user visiting the same POI more than once) and applied a 2-core (removing all users and items with less than 2 interactions). Those users whose attributes (age or gender) were not available in the data were also removed.

Regarding the user attributes, we decided to group the user ages in four intervals: [1, 18), [18, 35), [35, 56) and [56, $+\infty$). We have also created 4 user groups based on the quartiles obtained from the number of items consumed by each user, either in the training or in the test set. Hence, users are grouped in quartiles Q1-Q4 according to the number of preferences of each user in the corresponding set (ordered from lowest to highest number of preferences).

For the item attributes, we work with a main feature and a secondary one: directors and genres in Movielens and level 3 and level 1 categories in Foursquare. We use as similarity function sim_F the Jaccard coefficient between the features of each pair of items; we set the penalization weight α as 0.8 and 0.6 for the main and secondary feature. When using both features at the same time, we first compute the similarity using the main feature, if there is no matching we use the secondary feature, and if there is no matching, we assume the items have a zero similarity. Additionally, we need to specify the τ function used to map the maximum similarity values into relevance weights; for this work we use the following simple mapping, in the future we aim to study how to better define such correspondence: $\tau(s) = 0.25$

if $0 < s \leq 0.5$, $\tau(s) = 0.5$ if $0.5 < s \leq 0.75$, $\tau(s) = 0.75$ if $s > 0.75$, and $\tau(s) = 0$ otherwise. Abusing the notation, we shall use τ_m, τ_s, or $\tau_{m,s}$ when this function is applied to the output of the Jaccard similarity as explained before to the main, secondary, or combined features.

3.2 Experimental setup

We compare the following algorithms: Random (Rnd) and Popularity (Pop) recommenders, two neighborhood-based approaches (UB, a user-based technique and IB, an item-based one), a pure content-based approach using a Vector Space Model to represent users and items (denoted as CB), the hybrid CB and CF approach from [1] and two matrix factorization techniques (HKV, the matrix factorization from [11] and the BPR recommender using the optimization method from [17]). We also report a recommender that returns the test results (denoted as Sky) to serve as a reference for the optimal metric values.

The parameters of the recommenders were selected according to the optimal values obtained in the NDCG@5 metric. For all the results, we have followed the TrainItems methodology [18] so that every item in the training set is considered as candidate except the ones already consumed by the user under a random split (80% of the preferences for training, the rest for test). The metrics are computed for all the users having at least one relevant item in the test set. Finally, we decided to use a threshold of 4 in Movielens dataset so that only items whose rating is higher or equal than this value are considered as relevant. In the case of Foursquare, since we only have binary interactions, we have considered every test item as relevant.

Scripts and source code to replicate the results can be found in the following Bitbucket repository: PabloSanchezP/AttrEval4RecSys.

3.3 User attributes in evaluation metrics

In this section we discuss the performance of the recommenders when analyzed according to different user groups. Table 1 shows the results in the Movielens dataset comparing the performance of NDCG@5 (denoted as Std) against different aggregations based on user attributes. By analyzing the first attribute (gender), we observe the performance is generally much lower for females than for males in every recommender except Rnd. This can be explained if we take into account that there is a large difference between males and females in both the number of users and their activity in the system – 28% of the total users are defined as women, representing a 25% of the total interactions – so the results on the least represented group are eclipsed by the rest. However, this performance difference does not only appear for the gender attribute. When comparing the different age intervals (the next four columns), we observe a similar behavior. In this case, the results for users with less than 18 years and more than 56 years (columns 1 and 56) are markedly inferior than for the rest of the users. In this case, both groups combined represent approximately 10% of the users in the system, but only 6.6% of the preferences. This is an evidence – also raised by recent research on fairness-aware recommendation [5] – that most recommenders are not able to make enough relevant recommendations to those users who belong to under-represented groups in the system.

Additionally, we have considered the activity of the user in the system as another attribute, by dividing the users on four quartiles according to the number of preferences in training or test; in this way, we are able to simulate evaluation of cold and warm users [13] through the training quartiles and typical filters applied to the data (such as ignoring users with low ratings in test) through the test quartiles. Hence, we observe in the last 8 columns that the performance increases for all recommenders as the quartiles increase (i.e., the more preferences the system stores about a user, the higher the results); this happens for both training and test quartiles. While these results are reasonable for the training quartiles since the algorithms have more training data to learn the preferences from, we hypothesize those user with many ratings in test correspond in fact to those users with many ratings in training, due to the random split that we have performed and, because of this, the same rationale applies to why the algorithms perform better in the last test quartiles.

Table 2 shows the results in the Foursquare dataset. In this case, the results are remarkably different from those obtained in Movielens, for both the gender and user activity attributes (age is not available in this dataset). Even though the percentage of men and women is similar in both datasets (36% of users are women in Foursquare), the difference in performance is not as large as before. This might be attributed to the fact that in location-based systems, visits to certain places do not depend on specific aspects of users, since tourists tend to make similar check-ins in venues (such as restaurants, museums, hotels, and so on) regardless of their gender. Moreover, we observe now that those users who have more preferences in both training and test (quartiles Q3 and Q4) have a worse performance than users with a less mature history. One possible reason for this result is the strong popularity bias present in this domain – where only one recommender outperforms Pop under the standard evaluation –, which is further emphasized due to the large sparsity of the data; these two effects (combined with the use of the TrainItems methodology) promote that users with few ratings get recommended popular (and relevant) items, whereas those with a more mature history cannot receive those items as recommendations because they have previously visited them, forcing to recommend items from the long tail which are, in general, more difficult to guess correctly (in part, because of the popularity bias).

3.4 Item attributes in evaluation metrics

In this section we present the results obtained when item attributes are exploited within the evaluation metrics, as defined in Equation 3. Table 3 shows the experiments in the Movielens and Foursquare datasets based on NDCG@5; the first column ($\tau = 0$) corresponds to the evaluation with no item attributes, whereas in the other columns either the main feature (τ_m), the secondary one (τ_s), or both ($\tau_{m,s}$) are incorporated in the metric computation, as explained in Section 3.1.

We observe that when we use the item attributes, the performance of all recommenders increases; this is an expected result since in each column the matching process is more and more flexible – note that secondary features are less specific than the main features –, by allowing to consider a larger amount of items as relevant (even though those items would be penalized). A clear example of such effect is the performance of Rnd, which is increased to an extent where some algorithms are outperformed by it (this only happens in Foursquare). Because of this, when using the proposed methodology the Rnd algorithm should always be included to take its value as a reference, otherwise, we would not be able to discriminate between a method that simply follows the inherent biases on the item attributes of a dataset and another one that actually exploits user and item patterns.

Table 1: Performance of the recommenders in Movielens. Std denotes the standard evaluation (all users are considered in the evaluation) measured with NDCG@5. The rest of the columns denote aggregations on different user attributes: F and M for females and males (gender), four age groups, and training and test quartiles. In bold, best results ignoring Sky recommender.

		Gender		Age				Training Quartile				Test Quartile			
Rec	Std	F	M	1	18	35	56	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
Rnd	0.004	0.004	0.004	0.001	0.005	0.004	0.004	0.001	0.002	0.004	0.010	0.001	0.002	0.004	0.011
Pop	0.141	0.093	0.159	0.108	0.168	0.116	0.072	0.066	0.082	0.136	0.278	0.064	0.084	0.141	0.275
UB	0.300	0.250	0.320	0.258	0.320	0.276	0.254	0.210	0.237	0.302	0.419	0.205	0.241	0.307	0.419
IB	0.241	0.196	0.259	0.220	0.259	0.227	0.186	0.142	0.192	0.250	0.381	0.142	0.194	0.254	0.380
HKV	**0.314**	**0.267**	**0.334**	**0.266**	**0.334**	**0.299**	**0.267**	**0.218**	**0.258**	**0.329**	**0.458**	**0.212**	**0.263**	**0.335**	**0.459**
BPR	0.241	0.211	0.253	0.211	0.257	0.229	0.201	0.140	0.177	0.232	0.336	0.135	0.181	0.239	0.335
CB	0.018	0.014	0.020	0.013	0.019	0.017	0.020	0.010	0.014	0.022	0.027	0.009	0.015	0.023	0.026
CBCF	0.249	0.204	0.267	0.208	0.270	0.225	0.179	0.135	0.182	0.262	0.414	0.137	0.184	0.265	0.412
Sky	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000

Table 2: Performance of the recommenders in Foursquare. Same notation as Table 1.

		Gender		Training Quartile				Test Quartile			
Rec	Std	F	M	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
Rnd	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Pop	0.088	0.088	0.087	**0.115**	0.091	0.083	0.069	0.093	**0.099**	0.079	0.079
UB	**0.096**	**0.093**	**0.097**	0.108	**0.099**	**0.095**	**0.086**	**0.095**	0.097	**0.094**	**0.096**
IB	0.037	0.030	0.041	0.043	0.039	0.037	0.032	0.037	0.041	0.037	0.035
HKV	0.080	0.075	0.083	0.088	0.085	0.079	0.073	0.081	0.078	0.077	0.081
BPR	0.081	0.079	0.082	0.109	0.084	0.080	0.061	0.088	0.088	0.075	0.071
CB	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
CBCF	0.074	0.074	0.074	0.076	0.088	0.074	0.069	0.071	0.078	0.069	0.079
Sky	0.997	0.997	0.996	0.997	0.997	0.995	0.997	1.000	0.992	0.992	0.998

Furthermore, since the order of the algorithms does not change considerably when the item attributes are used, we conclude this methodology is complementary to the standard one, as our approach first considers the exact matchings, thus, depending on how many unmatched items are left, the recommenders would have more or less opportunities to fill the gap with the similar items (surrogates).

Nonetheless, those cases where the ranking does change deserve further analysis since they could explain where some recommenders are failing or succeeding. In particular, in our results we observe that only IB and BPR change their positions in Movielens, indicating that BPR is better at finding correct genres than IB. In Foursquare there are more examples where algorithms change their relative performance, evidencing the increment in difficulty of the recommendation task in this domain, as discussed before, mostly due to the high sparsity and strong popularity bias. In any case, when categories from level 3 are considered (τ_m), Pop decreases its (relative) performance in favor of HKV and CBCF, whereas when categories from level 1 are incorporated, BPR and CB decrease their relative ranking and Rnd and IB obtain better results, in particular IB ends up as the best recommender after Sky; this, in part, is caused by the fact that some categories are much more popular than others, which, together with the large number of items and very small number of categories available in this dataset, make easier for item-based similarities to promote relevant items learned through collaborative patterns.

Table 3: Performance of the recommenders in Movielens and Foursquare measured with NDCG@5 and $w^+ = 1$ and $w^- = 0$. In bold, best results ignoring Sky recommender.

Rec	Movielens				Foursquare			
	$\tau = 0$	τ_m	τ_s	$\tau_{m,s}$	$\tau = 0$	τ_m	τ_s	$\tau_{m,s}$
Rnd	0.003	0.012	0.218	0.222	0.000	0.029	0.216	0.224
Pop	0.141	0.165	0.288	0.298	0.088	0.098	0.205	0.208
UB	0.300	0.319	0.435	0.444	**0.096**	**0.126**	0.244	0.252
IB	0.241	0.261	0.381	0.403	0.037	0.081	**0.246**	**0.258**
HKV	**0.314**	**0.334**	**0.456**	**0.451**	0.081	0.114	0.232	0.240
BPR	0.241	0.261	0.394	0.389	0.081	0.094	0.208	0.212
CB	0.018	0.030	0.233	0.239	0.000	0.048	0.216	0.225
CBCF	0.249	0.267	0.406	0.414	0.074	0.100	0.218	0.224
Sky	1.000	1.000	1.000	1.000	0.997	0.997	0.997	0.997

4 CONCLUSIONS AND FUTURE WORK

The evaluation of RS remains as a fruitful research area where several issues are still open. In this work we address the problem of integrating user and item attributes in standard evaluation metrics, so that researchers and practitioners could gain more insights from their evaluations, by adding domain knowledge to this critical step in any development and validation process.

Our results evidence that the presented methodology is valid and allows to simulate different evaluation scenarios without having to re-run the recommendation algorithms. First of all, by incorporating user attributes the evaluation metrics could focus on specific groups that share the same attribute; in particular, we have shown the cases of using gender, age, and user consumption as those dimensions where it might be interesting to assess how good or bad the algorithms perform in each user group; however, there are many other use cases that have been left out of our analysis for the sake of space, such as discriminating between active or influential users, or between bots (or any other type of attacker) or locals in the tourism domain [16].

Finally, by incorporating item attributes into the evaluation metrics many possibilities open up to better understand the behavior of the recommendation algorithms, such as how to decide when an item should be selected as a surrogate of another item and how much it should weight (in this work we used simple item attributes such as genres or categories); in particular, we are interested in incorporating such knowledge back into the recommendation algorithms to improve their performance since, as we have found in our experiments, some techniques may perform quite well when surrogates are exploited but very poorly when traditional evaluation is used.

ACKNOWLEDGMENTS

This work has been funded by the Ministerio de Ciencia, Innovación y Universidades (reference TIN2016-80630-P) and by the European Social Fund (ESF), within the 2017 call for predoctoral contracts.

REFERENCES

[1] Marko Balabanovic and Yoav Shoham. 1997. Content-Based, Collaborative Recommendation. *Commun. ACM* 40, 3 (1997), 66–72.

[2] Alejandro Bellogín, Pablo Castells, and Iván Cantador. 2017. Statistical biases in Information Retrieval metrics for recommender systems. *Inf. Retr. Journal* 20, 6 (2017), 606–634.

[3] Igo Ramalho Brilhante, José Antônio Fernandes de Macêdo, Franco Maria Nardini, Raffaele Perego, and Chiara Renso. 2013. Where shall we go today?: planning touristic tours with tripbuilder. In *CIKM*. ACM, 757–762.

[4] Pablo Castells, Neil J. Hurley, and Saul Vargas. 2015. Novelty and Diversity in Recommender Systems. In *Recommender Systems Handbook*. Springer, 881–918.

[5] Golnoosh Farnadi, Pigi Kouki, Spencer K. Thompson, Sriram Srinivasan, and Lise Getoor. 2018. A Fairness-aware Hybrid Recommender System. *CoRR* abs/1809.09030 (2018).

[6] Shir Frumerman, Guy Shani, Bracha Shapira, and Oren Sar Shalom. 2019. Are All Rejected Recommendations Equally Bad?: Towards Analysing Rejected Recommendations. In *UMAP*. ACM, 157–165.

[7] Asela Gunawardana and Guy Shani. 2015. Evaluating Recommender Systems. In *Recommender Systems Handbook*. Springer, 265–308.

[8] F. Maxwell Harper and Joseph A. Konstan. 2016. The MovieLens Datasets: History and Context. *TiiS* 5, 4 (2016), 19:1–19:19.

[9] Jing He, Xin Li, and Lejian Liao. 2017. Category-aware Next Point-of-Interest Recommendation via Listwise Bayesian Personalized Ranking. In *IJCAI*. ijcai.org, 1837–1843.

[10] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John Riedl. 2004. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.* 22, 1 (2004), 5–53.

[11] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *ICDM*. IEEE Computer Society, 263–272.

[12] Dietmar Jannach and Gediminas Adomavicius. 2016. Recommendations with a Purpose. In *RecSys*. ACM, 7–10.

[13] Daniel Kluver and Joseph A. Konstan. 2014. Evaluating recommender behavior for new users. In *RecSys*. ACM, 121–128.

[14] Kwan Hui Lim, Jeffrey Chan, Christopher Leckie, and Shanika Karunasekera. 2015. Personalized Tour Recommendation Based on User Interests and Points of Interest Visit Durations. In *IJCAI*. AAAI Press, 1778–1784.

[15] Enrico Palumbo, Giuseppe Rizzo, Raphaël Troncy, and Elena Baralis. 2017. Predicting Your Next Stop-over from Location-based Social Network Data with Recurrent Neural Networks. In *RecTour@RecSys (CEUR Workshop Proceedings)*, Vol. 1906. CEUR-WS.org, 1–8.

[16] Evangelos E. Papalexakis, Konstantinos Pelechrinis, and Christos Faloutsos. 2014. Spotting misbehaviors in location-based social networks using tensors. In *WWW (Companion Volume)*. ACM, 551–552.

[17] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI*. AUAI Press, 452–461.

[18] Alan Said and Alejandro Bellogín. 2014. Comparative recommender system evaluation: benchmarking recommendation frameworks. In *RecSys*. ACM, 129–136.

[19] Alan Said and Alejandro Bellogín. 2018. Coherence and inconsistencies in rating behavior: estimating the magic barrier of recommender systems. *User Model. User-Adapt. Interact.* 28, 2 (2018), 97–125.

[20] Michael Trusov, Anand V Bodapati, and Randolph E Bucklin. 2010. Determining influential users in internet social networks. *Journal of Marketing Research* 47, 4 (2010), 643–658.

[21] Dingqi Yang, Daqing Zhang, and Bingqing Qu. 2016. Participatory Cultural Mapping Based on Collective Behavior Data in Location-Based Social Networks. *ACM TIST* 7, 3 (2016), 30:1–30:23.