Measuring the Ranking Quality of Recommendations in a Two-Dimensional Carousel Setting

Nicolò Felicioni[1], Maurizio Ferrari Dacrema[1], Fernando B. Pérez Maurera[1,2] and Paolo Cremonesi[1]

[1]*Politecnico di Milano, Milan, Italy*
[2]*ContentWise, Milan, Italy*

Abstract
Movie-on-demand and music streaming services usually provide the user with multiple recommendation lists, i.e., carousels, in a two-dimensional user interface, each generated according to different criteria (e.g., TV series, popular artists, etc.). In this two-dimensional setting it is not appropriate to use traditional ranking metrics designed for a single ranking list. It is well known that users do not explore a two-dimensional interface one row at a time, but rather focus their attention in a triangular area at the top-left corner. Furthermore, it is frequent for user interfaces to hide some items or lists due to space constraints, which can be shown by performing certain actions (i.e., click, swipe). In this paper we extend the widely used NDCG to a two-dimensional recommendation setting with a formulation that allows to account both the two-dimensional user exploration behaviour and interface-specific design. We also compare the proposed extension against single-list NDCG highlighting that they can lead to a different choice of the optimal algorithm in offline evaluation.

Keywords
Recommender Systems, User Interface, Evaluation

1. Introduction

Traditionally, in the Information Retrieval and Recommender Systems domains, the objective has been to provide the user with the best possible ranked list of results [1, 2, 3]. For this reason, many metrics were developed to evaluate the quality of a one-dimensional ranked list. A common assumption is that users will navigate the list according to its order, therefore it is better for a correct recommendation to be at the beginning of the list.

There are however several scenarios that do not fit into these assumptions, mainly when the results are presented in a two-dimensional grid rather than a single list. This is true both in information retrieval [4] and in recommendation systems applications, in particular for video-on-demand streaming services [5, 6, 7] and music streaming platforms [8, 9]. Those services usually provide users with multiple rows of thematically coherent recommendations

IIR 2021 – 11th Italian Information Retrieval Workshop, September 13–15, 2021, Bari, Italy
nicolo.felicioni@polimi.it (N. Felicioni); maurizio.ferrari@polimi.it (M. Ferrari Dacrema); fernando.perez@contentwise.com (F. B. Pérez Maurera); paolo.cremonesi@polimi.it (P. Cremonesi)
0000-0002-3555-7760 (N. Felicioni); 0000-0001-7103-2788 (M. Ferrari Dacrema); 0000-0001-6578-7404 (F. B. Pérez Maurera); 0000-0002-1253-8081 (P. Cremonesi)

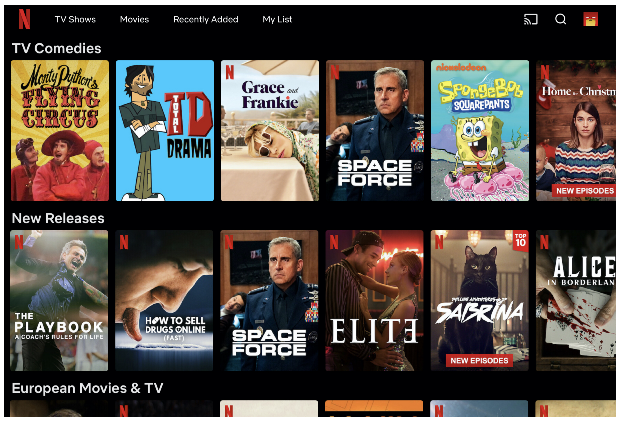


Figure 1: The Netflix homepage, an example of carousel user interface in the multimedia streaming domain.

(e.g., the most popular movies, a specific genre, new releases, and so on, see Figure 1). These rows are referred to as *widgets*, *shelves* or as *carousels*.

A simple way to adapt one-dimensional ranking metrics to a two-dimensional interface is to concatenate all recommendation lists into a single one. This strategy does not make realistic assumptions and, we argue, is not appropriate. First, it is known that users do not explore each carousel sequentially from the first to the last, as concatenating them assumes. Rather, users start from the top-left corner of the screen and proceed to explore the items both to the right and to the bottom [10, 11]. This effect is also known as "golden triangle" or "F-pattern". A visual example from an information retrieval application [4] is shown in Figure 2. Another example from a video streaming service [7] is shown in Figure 3. In addition to this user behaviour, many websites and mobile applications present carousels that are *swipeable* [9], i.e., the user can swipe horizontally or vertically to reveal more items as well as lists that were not previously visible. This is a common way to overcome the limited space available in the user interface allowing to fit more recommendations and carousels that the user can easily browse. However this puts additional overhead on the user that has to actively interact with the system to access the recommendations. Hence, it is preferable for a correct recommendation to be visible with the least possible number of user actions, as also noticed in [12].

In order to take those factors into account, in this paper we propose to extend the one-dimensional NDCG metric to consider both the two-dimensional user exploration behaviour and the user interface characteristics. We show that the two metrics can lead to different results when used to select which recommenders to use in the carousel interface.

The rest of the paper is organized as follows, in Section 2 we summarize the characteristics of a carousel setting, in Section 3 we formulate an extended version of NDCG, in Section 4 we perform an offline comparison of the results in a single list and carousel interface. Finally in Section 5 we draw the conclusions.

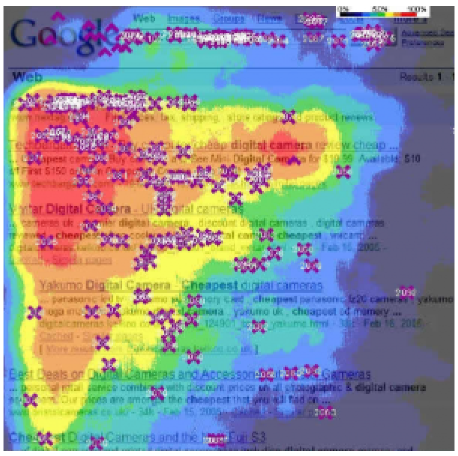


Figure 2: When using a search engine users concentrate their attention on the top-left corner (*golden triangle*) [4].

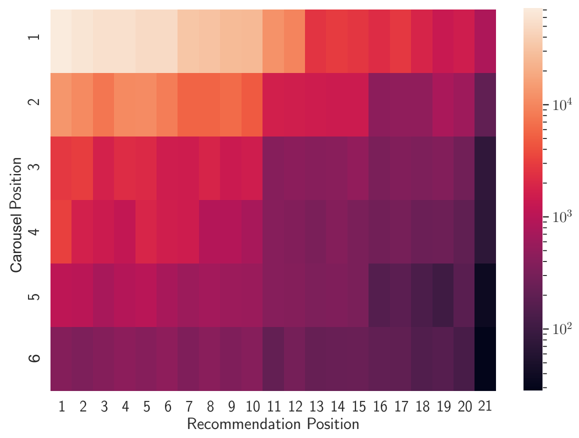


Figure 3: Visualization of the number of user interactions on each position on a user interface [7]. A demarcation between the first and second half of the columns is visible.

2. Characteristics of a Carousel Setting

The carousel interface layout and the way it is usually generated by video-on-demand and music streaming platforms has important characteristics that distinguish it from a single-list setup [13]:

Interface: A two dimensional user interface with multiple carousels. Some carousels or recommendations may be hidden due to limited page size and be accessible only via user actions (i.e., click, swipe).

Recommendations: The lists shown to the users are generated with different algorithms or by different providers and no single post-processing step is applied. While each individual recommendation list does not contain duplicates, the same item may appear in multiple carousels.[1][2]

User Behaviour: The user will focus on the top-left triangle of the screen rather than exploring the carousels sequentially. Furthermore, they will explore the recommendations in different ways according to which actions they need to perform in order to reveal them.[3]

While a carousel layout may seem similar to a traditional merge-list embedding, where multiple recommendation list are combined into one, this is not the case. In a real scenario, there

[1]A significant example are *content aggregators*, which combine carousels from different providers: Netflix, Youtube, Prime Video, etc.

[2]For example, in the Netflix homepage shown in Figure 1 the TV series *Space Force* appears both in the *TV Comedies* and *New Releases* carousels.

[3]Usually users tend to navigate more easily with simple swipes rather than repeated mouse clicks, hence their behaviour, as it is known, will change according to the device.

are multiple constraints. First, the carousels may be generated by different content providers, each of them unaware of how the other lists are generated or by whom. This means that the composition of the layout as well as the recommendations of the other providers are, in general, not known. It is for this reason that different carousels may contain similar recommendations. Furthermore, a content provider that wishes to select the optimal carousels to display has limited degrees of freedom and can only alter the content and relative ordering of those it is tasked to provide. Finding strategies to select the optimal carousel layout is a complex problem [14].

3. Extending one-dimensional NDCG

One of the most used metrics for ranked list evaluation is the *Discounted Cumulated Gain* (DCG), as well as its *Normalized* version (NDCG) [15, 16]. This metric comes from the information retrieval domain and is widely used to evaluate recommendation systems. The DCG metric relies on two assumptions:

1. highly relevant results are more valuable for a user;
2. within a list of results, it is preferable to have relevant results in the first positions

Let c be the recommendation list length, i.e., cutoff, and $rel(i)$ the relevance of the item in position i. The DCG is defined as the following discounted sum of gains:

$$DCG = \sum_{i=1}^{c} gain(i) \cdot discount(i)$$

The $gain$ function is responsible for rewarding highly relevant results, while the $discount$ function introduces a penalization that should increase the further the item is from the beginning of the list.

One of the most used formulations for the DCG is the following [17]:

$$DCG = \sum_{i=1}^{c} \frac{2^{rel(i)} - 1}{\log_2(i+1)}$$

Hence, $gain(i) = 2^{rel(i)} - 1$ and $discount(i) = \frac{1}{\log_2(i+1)}$. Notice that this formulation is only one of many possible formulations for the DCG. Several other ways of rewarding and discounting results have been proposed in previous research [18, 19]. In the following, we will start from this formulation and extend it since it is one of the most used. Other types of gain and discount functions can be extended in an analogous way. We leave the analysis of different gains and discounts as future work.

In a two-dimensional scenario, the standard DCG definition could be naively adapted in the following way. Let h be the horizontal dimension of the interface (i.e., the length of each carousel) and v the vertical dimension of the interface (i.e., the number of carousels). The carousels can be concatenated in a single list of length $c = v \cdot h$ items on which the standard DCG formulation can be applied. This strategy assumes that the users will explore all carousels sequentially, from the first to the last, which, as previously discussed, is not consistent to the user behaviour and does not account for the interface navigation constraints. Therefore, we suggest researchers *do not* apply this strategy as it does not represent a realistic scenario.

Thus, inspired by [15], we make the following assumptions the two-dimensional DCG should meet:

1. highly relevant results are more valuable for a user;
2. a relevant result is valuable to the user only when it is first seen;
3. within a grid of results, it is preferable to have relevant results close to the top-left corner
4. it is preferable that relevant items are immediately visible to the user or can be made visible with few user actions

In order to account for this set of assumptions, we propose to extend the metric in the following way:

$$2DCG = \sum_{i=1}^{v} \sum_{j=1}^{h} gain(i,j) \cdot discount(i,j)$$

As in the one-dimensional version, the $gain$ function is responsible for rewarding highly relevant results, according to assumptions (1) and (2). The $discount$ function, instead, should account for the penalty related to the position and number of user actions, according to assumptions (3) and (4).

Inspired by the one-dimensional version, we fix $gain(i,j) = 2^{rel(i,j)} - 1$. Instead, the $discount$ will depend on the position in the layout, allowing ample freedom on how to define it in different use cases.

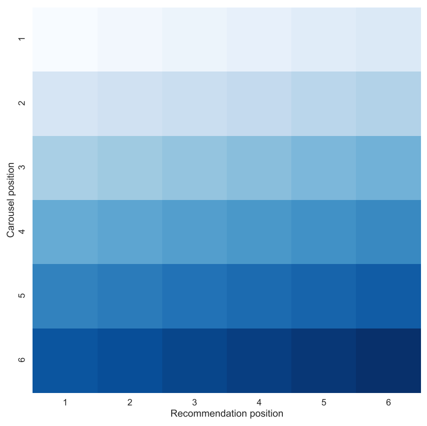
The normalized version of this metric, *N2DCG* will be defined as as $N2DCG = 2DCG/I2DCG$. I2DCG will be the 2DCG of the *ideal ranking*. In a single list setting the ideal ranking is the list which contains the relevant items in decreasing relevance from the beginning of the list. In the generalized two-dimensional layout it contains the user's most relevant items, ranked according to decreasing relevance in positions with decreasing position discount. The ideal ranking meets the following constraints: for any pair of cells (i,j), (k,l) of the matrix, $gain(i,j) \geq gain(k,l)$ if $discount(i,j) > discount(k,l)$.

Relevance As stated in assumption (2), a relevant item is valuable for the user only when it is first encountered. This means that if a relevant item appears multiple times, each in a different carousels, it should be considered as relevant only in its *best* position. We define such position as the one with the highest *discount*. Function $rel(i,j)$ should be modified accordingly.

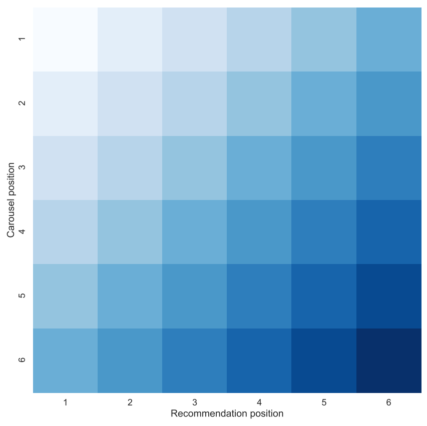
Single List Discount It is possible to represent in this formulation the traditional single list DCG by calculating the position of cell in coordinates i, j if all carousels lists would be concatenated:

$$discount_{singleList}(i,j) = (log_2((i-1) \cdot h + j + 1))^{-1}$$

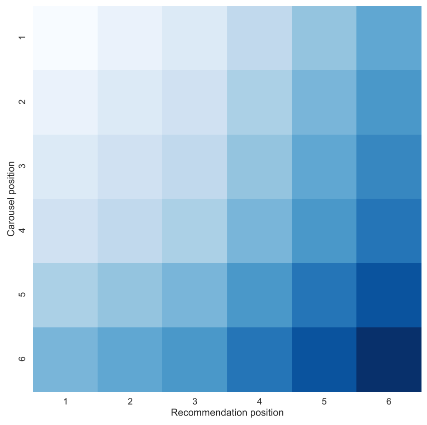
As previously mentioned, this formulation is not grounded in a realistic scenario because it does not reflect the user behaviour (see Figure 4a), therefore we argue it should not be applied.



(a) Single list.



(b) *Golden triangle* behaviour.



(c) *Golden triangle* and user actions penalty.

Figure 4: A visual comparison of the two-dimensional penalty function under different assumptions. Figure 4a refers to carousels concatenated in a single list. The other figures refer to the two-dimensional penalty which accounts for the *golden triangle* behaviour only, see Figure 4b as well as the number of user actions, see Figure 4c.

Golden Triangle Discount In order to account for the *golden triangle* behaviour, as per assumption (3), the position discount should decrease as the distance of the cell from the top-right corner increases:

$$discount_{triangle}(i,j) = (log_2(\alpha \cdot i + \beta \cdot j))^{-1}$$

The coefficients α, β are two weights that can be used to account for different types of user behaviors. For instance, let us assume a scenario where users are more inclined to explore the

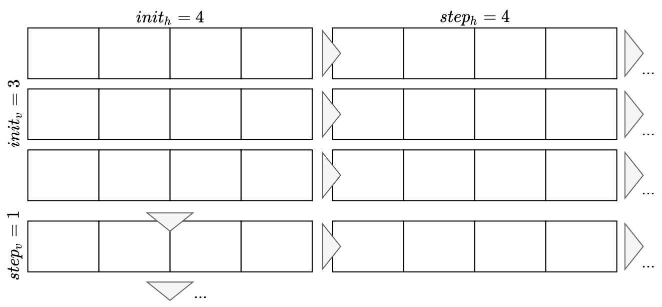


Figure 5: An example interface where 3 carousels, with 4 items each, are visible. A horizontal swipe reveals 4 items, while a vertical swipe reveals one additional carousel.

vertical dimension. In this case, α should be set to a low value in order to penalize less the vertical dimension. In order to make the discount start from 1, α and β should be ≥ 1 since the base of the logarithm used is 2. Notice that this is true only because we are extending a logarithmic discount function. For other discount functions [18, 19] the constraints can change.

The resulting discount is shown in Figure 4b (we set $\alpha = \beta = 1$ for simplicity).

User Actions Discount Lastly, in order to account for assumption (4) the position discount should decrease the more actions are required by the user to make that position visible. In a carousel interface there is an initial rectangular portion of the recommendations that are immediately shown to the user. We refer to the number of items visible as $init_h$ and to the number of carousels visible as $init_v$, see Figure 5. In order to reveal more items, the user needs to perform a certain action, i.e., click on a desktop, swipe on mobile devices. Each of these actions will reveal a certain number of new items within the currently visualized recommendation lists. Different platforms and devices will correspond to different *swipe steps*, i.e., the number of items that will be revealed after a single swipe. We will call this quantity $step_h \in \{1, 2, \ldots, init_h\}$. For example, on Netflix every click will replace all items displayed on the clicked carousel, in which case $step_h = init_h$. The same principle holds for the vertical dimension, where the user can navigate performing actions that will each display $step_v$ new carousels.

Based on this definition, we now add to the triangle penalty a term to account for the number of actions that the user will need to perform in order to visualize the item. To do so we define some auxiliary functions. The first one is used to check whether at least a user action, i.e., swipe, is needed to visualize that item in a certain position p given that the interface initially shows $init$ positions:

$$\text{isSwipeNeeded}(p, init) = \begin{cases} 1, & \text{if } p - init > 0 \\ 0, & \text{otherwise} \end{cases}$$

Then, we define a function to count the number of actions needed to visualize an item, given that each action shows $step$ positions:

$$\text{swipes}(p, init, step) = \text{isSwipeNeeded}(p, init) \cdot \left\lceil \frac{p - init}{step} \right\rceil$$

In the particular case where $init = step$, calculating the number of swipes becomes simpler:

$$\text{swipes}(p, step) = \left\lfloor \frac{p}{step} \right\rfloor$$

The final discount will account for both the triangle discount and the number of user actions, as previously defined:

$$discount_{actions}(i, j) = (log_2(\alpha \cdot i + \beta \cdot j + \gamma \cdot \text{swipes}(i, init_v, step_v)) \\ + \lambda \cdot \text{swipes}(j, init_h, step_h))^{-1}$$

Notice that this formulation accounts for both vertical and horizontal swipes. The coefficients α, β, γ, λ are four positive weights that can be used to account for different types of user behaviors. The first two weights (α and β) control the general penalization of the vertical and horizontal dimensions, respectively. As we previously said, they should be ≥ 1 in order for the total discount to start from 1. Controlling γ and λ, instead, it is possible to penalize more or less the user actions needed to reveal a certain item. For example, it could be that items presented together in the same carousel have a similar probability of interaction (see the first 10 elements of the first carousel in Figure 3). Hence, the horizontal dimension should be penalized less. Another possibility is that, on a desktop device, the horizontal swipe done with a mouse click will have a higher weight than the same swipe done with a touch on a mobile device.

For illustrative purposes, let us consider a possible scenario for a mobile device, where the screen contains 4 carousels and 3 recommendations each. We set the horizontal and vertical steps to 1, $\alpha, \beta, \gamma, \lambda$ are set to 1 as well. The resulting discount is shown in Figure 4c.

4. Experiments

In this section we provide an example of the different behaviour of NDCG and N2DCG in an offline experimental scenario. We consider a setting where given a set of recommendation models and a certain number of carousels, the goal is to select which models to use to generate each carousel. We show that the two metrics yield to different carousel layouts. In order to represent a scenario where a carousel interface would be used, we selected the widely known movie recommendations dataset *MovieLens10M* dataset [20], containing 70k users, 10k items and 10M ratings.

The set of models that can be selected, i.e., *M*, contains several simple and widely known models that have shown to provide competitive results in recent evaluations [21]. For *Non-Personalized* models we selected a TopPopular recommender. As *KNN* models we included ItemKNN [22] and UserKNN [23], both computing the similarity with cosine and shrinkage. We included the *Graph-based* models $\text{P}^3\alpha$ [24] and $\text{RP}^3\beta$ [25], which define a bipartite graph of users and items and simulate a random walk. We added various *Matrix Factorization* models, some developed for explicit interactions: PureSVD [2], FunkSVD [21] and Non-negative MF (NMF) [26]; as well as others developed for implicit interactions: MF BPR [27], IALS [28]. We included the widely known *Item-Based machine learning* models SLIM [29], SLIM BPR and the more recent EASE^R [30]. Finally, we included the *Content-based* model ItemKNN CBF, which computes the item similarities from item features. using cosine similarity with shrinkage.

Optimizing NDCG	Optimizing N2DCG
UserKNN	**SLIM**
FunkSVD	FunkSVD
NMF	**UserKNN**
IALS	**MF BPR**
MF BPR	**NMF**
SLIM	**IALS**

Table 1
Layouts obtained optimizing NDCG and N2DCG.

We split the data by randomly selecting 80% of interactions for the training set and 10% for validation and test set. Each model was optimized on the validation data, following the best practices and value ranges reported in [21], using a Bayesian search with 50 cases.

Since the purpose of this paper is not to propose an algorithm for the selection of carousels but to show that the two metrics lead to different results, we rely on a simple greedy strategy. At the beginning the page is empty and all candidate algorithms are evaluated independently on the validation data. The model with the best recommendation quality is selected as first carousel. The process repeats for the following carousels, however, in this case, the candidate model will be evaluated by taking into account all the *previous* carousels. According to the definition of relevance provided in Section 3, a correct recommendation of an item by the candidate model may overtake another of the same item in a previous carousels if it has a better position discount. For example, a correct recommendation at the end of the second carousel could be overtaken by the same recommendation but at the beginning of the third carousel, if it has a better position discount.

We repeated this procedure first optimizing NDCG, and then optimizing N2DCG. We consider a hypothetical interface with a total of 6 carousels, each composed of 10 items. The interface will initially show 3 carousels and 2 items. The user can display 1 additional item in a given carousel with each horizontal swipe and 1 new carousel with a vertical swipe. For this interface, we set $\alpha = \beta = 1$ and $\gamma = \lambda = 2$, in order to penalize more the swipes.

The resulting layouts are shown in Table 1. As we can see, the layouts have almost completely different orders of the chosen algorithms. For instance, optimizing N2DCG results in selecting SLIM as the first carousel, while the same algorithm was selected at the bottom of the layout that optimizes one-dimensional NDCG. UserKNN instead was the first algorithm when optimizing NDCG, but it is only the third carousel during N2DCG optimization.

Notice also how the 6 algorithms selected in both procedures are the same, only the order changes. Indeed, it is expected that NDCG and N2DCG will not produce completely different layouts but will differ the longer and more pronounced the effects of user actions become.

5. Conclusions

In this paper we have described a user interface with multiple carousels, typical of movie-on-demand and music streaming services, and based on its characteristics proposed an extended version of the widely used NDCG metric. The proposed formulation accounts for the known user behaviour of exploring the pages not one row at a time but focusing on the top-left corner and then navigating in both directions. The proposed formulation also allows to penalize correct recommendations that are only visible to the user after performing actions. Lastly, we show that the two metrics can lead to the selection of a different carousel layout. Future works include validating the proposed metric with user studies as well as applying it to select the optimal carousel layout, by defining which is the best carousel to put in a certain position or which is the best ordering of a given set of carousels. Also, further studies can be done on different gain and discount functions, similar to previous research works conducted on the one-dimensional DCG.

References

[1] J. L. Herlocker, J. A. Konstan, L. G. Terveen, J. Riedl, Evaluating collaborative filtering recommender systems, ACM Trans. Inf. Syst. 22 (2004) 5–53. URL: https://doi.org/10.1145/963770.963772. doi:10.1145/963770.963772.

[2] P. Cremonesi, Y. Koren, R. Turrin, Performance of recommender algorithms on top-n recommendation tasks, in: X. Amatriain, M. Torrens, P. Resnick, M. Zanker (Eds.), Proceedings of the 2010 ACM Conference on Recommender Systems, RecSys 2010, Barcelona, Spain, September 26-30, 2010, ACM, 2010, pp. 39–46. URL: https://doi.org/10.1145/1864708.1864721. doi:10.1145/1864708.1864721.

[3] M. Sanderson, W. B. Croft, The history of information retrieval research, Proc. IEEE 100 (2012) 1444–1451. URL: https://doi.org/10.1109/JPROC.2012.2189916. doi:10.1109/JPROC.2012.2189916.

[4] F. Chierichetti, R. Kumar, P. Raghavan, Optimizing two-dimensional search results presentation, in: I. King, W. Nejdl, H. Li (Eds.), Proceedings of the Forth International Conference on Web Search and Web Data Mining, WSDM 2011, Hong Kong, China, February 9-12, 2011, ACM, 2011, pp. 257–266. URL: https://doi.org/10.1145/1935826.1935873. doi:10.1145/1935826.1935873.

[5] C. Wu, C. V. Alvino, A. J. Smola, J. Basilico, Using navigation to improve recommendations in real-time, in: S. Sen, W. Geyer, J. Freyne, P. Castells (Eds.), Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, September 15-19, 2016, ACM, 2016, pp. 341–348. URL: https://doi.org/10.1145/2959100.2959174. doi:10.1145/2959100.2959174.

[6] E. Elahi, A. Chandrashekar, Learning representations of hierarchical slates in collaborative filtering, in: R. L. T. Santos, L. B. Marinho, E. M. Daly, L. Chen, K. Falk, N. Koenigstein, E. S. de Moura (Eds.), RecSys 2020: Fourteenth ACM Conference on Recommender Systems, Virtual Event, Brazil, September 22-26, 2020, ACM, 2020, pp. 703–707. URL: https://doi.org/10.1145/3383313.3418484. doi:10.1145/3383313.3418484.

[7] F. B. Pérez Maurera, M. Ferrari Dacrema, L. Saule, M. Scriminaci, P. Cremonesi, Contentwise impressions: An industrial dataset with impressions included, in: M. d'Aquin, S. Dietze, C. Hauff, E. Curry, P. Cudré-Mauroux (Eds.), CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020, ACM, 2020, pp. 3093–3100. URL: https://doi.org/10.1145/3340531.3412774. doi:10.1145/3340531.3412774.

[8] A. Gruson, P. Chandar, C. Charbuillet, J. McInerney, S. Hansen, D. Tardieu, B. Carterette, Offline evaluation to make decisions about playlistrecommendation algorithms, in: J. S. Culpepper, A. Moffat, P. N. Bennett, K. Lerman (Eds.), Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, WSDM 2019, Melbourne, VIC, Australia, February 11-15, 2019, ACM, 2019, pp. 420–428. URL: https://doi.org/10.1145/3289600.3291027. doi:10.1145/3289600.3291027.

[9] W. Bendada, G. Salha, T. Bontempelli, Carousel personalization in music streaming apps with contextual bandits, in: R. L. T. Santos, L. B. Marinho, E. M. Daly, L. Chen, K. Falk, N. Koenigstein, E. S. de Moura (Eds.), RecSys 2020: Fourteenth ACM Conference on Recommender Systems, Virtual Event, Brazil, September 22-26, 2020, ACM, 2020, pp. 420–

425. URL: https://doi.org/10.1145/3383313.3412217. doi:10.1145/3383313.3412217.

[10] Y. Kammerer, P. Gerjets, How the interface design influences users' spontaneous trustworthiness evaluations of web search results: comparing a list and a grid interface, in: C. H. Morimoto, H. O. Istance, A. Hyrskykari, Q. Ji (Eds.), Proceedings of the 2010 Symposium on Eye-Tracking Research & Applications, ETRA 2010, Austin, Texas, USA, March 22-24, 2010, ACM, 2010, pp. 299–306. URL: https://doi.org/10.1145/1743666.1743736. doi:10.1145/1743666.1743736.

[11] Q. Zhao, S. Chang, F. M. Harper, J. A. Konstan, Gaze prediction for recommender systems, in: S. Sen, W. Geyer, J. Freyne, P. Castells (Eds.), Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, September 15-19, 2016, ACM, 2016, pp. 131–138. URL: https://doi.org/10.1145/2959100.2959150. doi:10.1145/2959100.2959150.

[12] K. Järvelin, S. L. Price, L. M. L. Delcambre, M. L. Nielsen, Discounted cumulated gain based evaluation of multiple-query IR sessions, in: C. Macdonald, I. Ounis, V. Plachouras, I. Ruthven, R. W. White (Eds.), Advances in Information Retrieval , 30th European Conference on IR Research, ECIR 2008, Glasgow, UK, March 30-April 3, 2008. Proceedings, volume 4956 of *Lecture Notes in Computer Science*, Springer, 2008, pp. 4–15. URL: https://doi.org/10.1007/978-3-540-78646-7_4. doi:10.1007/978-3-540-78646-7_4.

[13] N. Felicioni, M. Ferrari Dacrema, P. Cremonesi, A methodology for the offline evaluation of recommender systems in a user interface with multiple carousels, in: J. Masthoff, E. Herder, N. Tintarev, M. Tkalcic (Eds.), Adjunct Publication of the 29th ACM Conference on User Modeling, Adaptation and Personalization, UMAP 2021, Utrecht, The Netherlands, June 21-25, 2021, ACM, 2021, pp. 10–15. URL: https://doi.org/10.1145/3450614.3461680. doi:10.1145/3450614.3461680.

[14] M. Ferrari Dacrema, N. Felicioni, P. Cremonesi, Optimizing the selection of recommendation carousels with quantum computing, in: Proceedings of the Fifteenth ACM Conference on Recommender Systems, 2021. doi:10.1145/3460231.3478853.

[15] K. Järvelin, J. Kekäläinen, IR evaluation methods for retrieving highly relevant documents, in: E. J. Yannakoudakis, N. J. Belkin, P. Ingwersen, M. Leong (Eds.), SIGIR 2000: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, July 24-28, 2000, Athens, Greece, ACM, 2000, pp. 41–48. URL: https://doi.org/10.1145/345508.345545. doi:10.1145/345508.345545.

[16] K. Järvelin, J. Kekäläinen, Cumulated gain-based evaluation of IR techniques, ACM Trans. Inf. Syst. 20 (2002) 422–446. URL: http://doi.acm.org/10.1145/582415.582418. doi:10.1145/582415.582418.

[17] C. J. C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, G. N. Hullender, Learning to rank using gradient descent, in: L. D. Raedt, S. Wrobel (Eds.), Machine Learning, Proceedings of the Twenty-Second International Conference (ICML 2005), Bonn, Germany, August 7-11, 2005, volume 119 of *ACM International Conference Proceeding Series*, ACM, 2005, pp. 89–96. URL: https://doi.org/10.1145/1102351.1102363. doi:10.1145/1102351.1102363.

[18] E. Kanoulas, J. A. Aslam, Empirical justification of the gain and discount function for ndcg, in: D. W. Cheung, I. Song, W. W. Chu, X. Hu, J. J. Lin (Eds.), Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM 2009, Hong Kong, China, November 2-6, 2009, ACM, 2009, pp. 611–620. URL: https://doi.org/10.1145/1645953.1646032.

doi:10.1145/1645953.1646032.

[19] K. Zhou, H. Zha, Y. Chang, G. Xue, Learning the gain values and discount factors of discounted cumulative gains, IEEE Trans. Knowl. Data Eng. 26 (2014) 391–404. URL: https://doi.org/10.1109/TKDE.2012.252. doi:10.1109/TKDE.2012.252.

[20] F. M. Harper, J. A. Konstan, The movielens datasets: History and context, ACM Trans. Interact. Intell. Syst. 5 (2016) 19:1–19:19. URL: https://doi.org/10.1145/2827872. doi:10.1145/2827872.

[21] M. Ferrari Dacrema, S. Boglio, P. Cremonesi, D. Jannach, A troubling analysis of reproducibility and progress in recommender systems research, ACM Trans. Inf. Syst. 39 (2021). URL: https://doi.org/10.1145/3434185. doi:10.1145/3434185.

[22] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, Item-based collaborative filtering recommendation algorithms, in: Proceedings of the 10th International Conference on World Wide Web (WWW '01), 2001, pp. 285–295.

[23] B. M. Sarwar, G. Karypis, J. A. Konstan, J. Riedl, Item-based collaborative filtering recommendation algorithms, in: V. Y. Shen, N. Saito, M. R. Lyu, M. E. Zurko (Eds.), Proceedings of the Tenth International World Wide Web Conference, WWW 10, Hong Kong, China, May 1-5, 2001, ACM, 2001, pp. 285–295. URL: https://doi.org/10.1145/371920.372071. doi:10.1145/371920.372071.

[24] C. Cooper, S. Lee, T. Radzik, Y. Siantos, Random walks in recommender systems: exact computation and simulations, in: C. Chung, A. Z. Broder, K. Shim, T. Suel (Eds.), 23rd International World Wide Web Conference, WWW '14, Seoul, Republic of Korea, April 7-11, 2014, Companion Volume, ACM, 2014, pp. 811–816. URL: https://doi.org/10.1145/2567948.2579244. doi:10.1145/2567948.2579244.

[25] B. Paudel, F. Christoffel, C. Newell, A. Bernstein, Updatable, accurate, diverse, and scalable recommendations for interactive applications, ACM Trans. Interact. Intell. Syst. 7 (2017) 1:1–1:34. URL: https://doi.org/10.1145/2955101. doi:10.1145/2955101.

[26] A. Cichocki, A. H. Phan, Fast local algorithms for large scale nonnegative matrix and tensor factorizations, IEICE Trans. Fundam. Electron. Commun. Comput. Sci. 92-A (2009) 708–721. URL: https://doi.org/10.1587/transfun.E92.A.708. doi:10.1587/transfun.E92.A.708.

[27] S. Rendle, C. Freudenthaler, Z. Gantner, L. Schmidt-Thieme, BPR: bayesian personalized ranking from implicit feedback, in: J. A. Bilmes, A. Y. Ng (Eds.), UAI 2009, Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, Montreal, QC, Canada, June 18-21, 2009, AUAI Press, 2009, pp. 452–461. URL: https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1&smnu=2&article_id=1630&proceeding_id=25.

[28] Y. Hu, Y. Koren, C. Volinsky, Collaborative filtering for implicit feedback datasets, in: Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008), December 15-19, 2008, Pisa, Italy, IEEE Computer Society, 2008, pp. 263–272. URL: https://doi.org/10.1109/ICDM.2008.22. doi:10.1109/ICDM.2008.22.

[29] X. Ning, G. Karypis, SLIM: Sparse linear methods for top-n recommender systems, in: Proceedings of the 11th IEEE International Conference on Data Mining (ICDM '11), 2011, pp. 497–506.

[30] H. Steck, Embarrassingly shallow autoencoders for sparse data, in: L. Liu, R. W. White, A. Mantrach, F. Silvestri, J. J. McAuley, R. Baeza-Yates, L. Zia (Eds.), The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019, ACM, 2019, pp. 3251–

3257. URL: https://doi.org/10.1145/3308558.3313710. doi:10.1145/3308558.3313710.