Effects of Position Bias on Click-Based Recommender Evaluation

Katja Hofmann[1], Anne Schuth[2], Alejandro Bellogin[3], and Maarten de Rijke[2]

[1] Microsoft Research, katja.hofmann@microsoft.com
[2] ISLA, University of Amsterdam, {anne.schuth, derijke}@uva.nl
[3] CWI, alejandro.bellogin@cwi.nl

Abstract. Measuring the quality of recommendations produced by a recommender system (RS) is challenging. Labels used for evaluation are typically obtained from users of a RS, by asking for *explicit* feedback, or inferring labels from *implicit* feedback. Both approaches can introduce significant biases in the evaluation process. We investigate biases that may affect labels inferred from *implicit* feedback. Implicit feedback is easy to collect but can be prone to biases, such as position bias. We examine this bias using click models, and show how bias following these models would affect the outcomes of RS evaluation. We find that evaluation based on implicit and explicit feedback can agree well, but only when the evaluation metrics are designed to take user behavior and preferences into account, stressing the importance of understanding user behavior in deployed RSs.

1 Introduction and Related Work

Recommender systems (RSs) aim to recommend to their users items of interest, such as movies, news articles, or music. Despite the success and popularity of such systems, measuring the quality of an RS is a challenge. In this paper, we examine how bias in user interactions, when used as implicit feedback, may affect RS evaluation.

Traditionally, RS evaluation has built upon *explicit* feedback, often in the form of user ratings, and error-based metrics such as Mean Absolute Error (MAE) or Root Mean Squared Error (RMSE) [9]. Recently, ranking-based metrics, such as precision, or normalized Discounted Cumulative Gain (nDCG) are being considered [1]. In both cases, ground truth typically comes in the form of user ratings for items. Creating explicit ratings for RS evaluation requires substantial user effort. In some applications, such as movie recommenders, users may be willing to expend that effort, especially when the rating process is embedded in an engaging interface. However, in applications such as news or music recommendation, explicit ratings are typically sparse and hard to obtain.

To increase the amount of data available for evaluation (or learning), researchers and practitioners have started to consider *implicit* feedback. Implicit feedback is a natural side-product of user interactions: labels are inferred based on, e.g., how many times an item has been clicked, viewed, or purchased [6, 8]. Given access to a deployed RS, collecting implicit feedback is typically easier, and the amount of data more plentiful, than for explicit feedback. However, accurately interpreting implicit feedback is non-trivial [7, 8].

In this paper we investigate how RS evaluation based on implicit feedback relates to traditional rating-based evaluation, and how evaluation outcomes may be affected by bias in user behavior. Unlike ratings, click feedback does not capture negative feedback, as it

conflates items that were not clicked because they were not examined and items that were examined but not clicked because they were disliked. Interpreting user interactions as feedback therefore requires a good model of how users interact with the system and how those interactions reflect experienced system quality. Here, we specifically focus on the well-known position bias [7]. We investigate this effect using bias models that capture how user behavior is hypothesized to be affected by the rank and quality of previously viewed items. We investigate how the estimated quality of a set of standard retrieval models is affected by these models, compared to the unbiased rating-based estimates.

In our idealized experimental setup, bias in user behavior can strongly affect the inferred outcomes of evaluation. Good agreement between evaluation based on implicit and explicit information can be achieved when metrics reflect user behavior and expectations.

2 Approach

To investigate bias in RS evaluation with implicit feedback, we need a way to generate implicit feedback under varying models of bias. Here, we leverage a simulation setup that was previously proposed for assessing IR evaluation methods [5]. It simulates the interactions between an interactive system and its users, so that the behavior of the system can be tested under varying assumptions about user behavior. We simulate the interactions between a representative set of RSs and user models that capture key aspects of position bias. The RSs are first trained on a representative data set, and then "deployed" to interact with the generative user models. The models simulate interactions that reflect the assumptions under which we want to examine RS evaluation. All simulated interactions are recorded and applied to evaluate each system. Finally, we compare the evaluation outcomes to each other and to ground truth evaluations to identify cases where bias affects outcomes. Next, we detail our user models and their assumptions on user behavior.

2.1 User models. We describe the implemented user models and how they capture user behavior and position bias. These models have been developed for the IR domain, and have been evaluated in large-scale evaluations using click log data [2–4]. We start with the simplest possible model, and build up complexity to cover assumptions that may affect evaluation outcomes based on the resulting clicks on recommended items. We phrase models in terms of clicks, but other user actions could be interpreted in the same way, e.g., watching a recommended movie, saving an item as a favorite, or rating it.

We consider three click models, which we refer to as examination, cascade, and browsing model. All models capture the probability of a click on a given item, $P(C = 1|i)$, by decomposing this probability into examination and relevance components: $P(C = 1|i) = P(E = 1|i)P(R = 1|i)$. Here, $P(E = 1|i)$ is the probability that item i is examined by the user. This may depend on several factors, as will be shown below. The second component, $P(R = 1|i)$, captures the relevance of the item for a given user. Items that are not examined are never clicked, i.e., $E_i = 0 \Rightarrow P(C_i) = 0$.

The user models differ in how they capture items' examination probabilities (see Table 1). In the *examination model* [10], the examination probability depends on the position of an item (represented by a_p), and it is independent of the examination probabilities of all other items. The *browsing model* (based on the User Browsing Model, UBM [3]) represents examination probabilities as a function of the distance from the closest previously clicked item: users may be more motivated to continue examining items if they liked the previous items. The *cascade model* [2] (cf., the extension for multiple

Table 1. User model specifications and definitions of examination probabilities.

model	intuition	parameters	definition
examination [10]	examination probability depends only on item position	item position (a_p)	$P(E=1\|p(i)) = a_p$
browsing [3]	users give up examination after seeing many bad items	distance from previous clicked item (a_d)	$P(C=1\|0) = 1$ $P(E=1\|d) = a_d$
cascade [2, 4]	user step through a list of items from top to bottom, stop when satisfied	satisfaction with previous item (s_{i-1})	$P(E=1\|p(i)=1) = 1$ $P(E=1\|i, s_{i-1}) = 1 - s_{i-1}$

clicks [4]) relaxes this independence assumption, and posits a linear examination order. Users are assumed to examine one item at a time, clicking on promising ones, until a clicked item was found to be satisfactory (s_{i-1} captures the satisfaction probability of the previous item). A special case of all three click models is the *no-bias* model, which sets the examination probability of all items to 1. We use this special case as baseline.

2.2 Ground truth and metrics. Metrics are used in two ways. First, we choose the metrics for implicit and explicit evaluation to reflect a typical RS evaluation setup. Second, we detail how the results are compared to each other, to assess differences and similarities in how the explicit and implicit evaluation metrics judge RS performance.

We assess the following *explicit* metrics, based on explicit user-item ratings provided with our data set: (1) Precision at N (P@N), the portion of highly rated items in the top N recommended items; (2) NDCG@N, a commonly used IR metric that rewards systems for showing highly rated items as high as possible in a recommendation list. We compare these metrics for explicit ratings to the *implicit* click through rate (CTR) at N, the ratio of clicked to non-clicked items in the top N recommended items.

We compare the outcomes obtained using different metrics as follows: (a) Rank the RSs by their evaluation score and compare the resulting rankings; (b) Consider the task of predicting performance based on implicit ratings from those obtained using explicit ratings. We train linear regression models for this prediction task and compare how well the performance under each user model (in terms of CTR) can be predicted from explicit rating scores. We estimate the fit of such a model using the residual standard error (RSE, i.e., how much the predicted performance deviates from actual performance).

3 Experiments

We describe the RSs we used, the data sets and the parameter settings.

3.1 Recommender systems. We compare the following types of recommender: (1) Two non-personalized methods serve as baselines: a random (RND) recommender and a popularity-based recommender (ItemPop) that ranks the item according to the observed popularity in the training set. (2) An item-based (IB) collaborative filtering recommender; we use Pearson's correlation as similarity between the items [9]. (3) A matrix factorization (MF) method as implemented in Mahout[4] using the Expectation Maximization algorithm. We used 100 iterations and 50 features, leaving the rest of the parameters as default (i.e., 0.005 as learning rate, 0.02 as regularization parameter, and 0.005 as random noise). (4) A user-based (UB) collaborative filtering method using 50 neighbors and Pearson's

[4] http://mahout.apache.org

correlation as similarity metric [9]. These four types of recommender (ignoring RND) are among the best performing ones in the literature [9].

3.2 Data. We obtain the ground truth and train the recommenders on the corresponding test and training splits (in an 80–20 ratio) derived from a version of the Movielens dataset;[5] we used Movielens 1M, with 1 million ratings by $6,040$ users to $3,900$ items.

3.3 Parameter settings. For each RS, we obtain the recommendations for all users in the test set. We simulate user interactions with these recommendations for each user model and record the generated interactions. We then compute the performance of each method using this data, and average results over 25 repetitions. We compare the resulting evaluation scores across models and systems, and also compare to ground truth scores based on the explicit ratings provided with the data sets as described in §2.2. We assume that users examine a maximum of 10 items and we compute all metrics on the top $N = 10$ item. P@10 and nDCG@10 are computed using only highly rated items (labels 4 and 5).

We instantiate the parameters of the user models (cf., §2.1) to capture a wide spectrum of possible user behavior. For the examination model we use *logarithmic* (log) ($a_p = 1/\log_2(p+1)$) or *quadratic* (quad) ($a_p = 1/2^{(p-1)}$) decay, $p > 0$. For the cascade model we use, for a previous item, if it is non-clicked, $s_{i-1} = 0$; otherwise, we instantiate the satisfaction $s_{i-1}(r)$, depending on the rating r of that item as follows; (*low*): $s_{i-1}(2) = 0.2$, $s_{i-1}(3) = 0.3$, $s_{i-1}(4) = 0.4$, and $s_{i-1}(5) = 0.5$; (*high*): $s_{i-1}(2) = 0.3$, $s_{i-1}(3) = 0.5$, $s_{i-1}(4) = 0.7$, and $s_{i-1}(5) = 0.9$. For the browsing model, a_d decays with the distance from the closest click above item i with logarithmic or quadratic decay, as defined for the examination model. All models use the same instantiations of probabilities $P(R = 1|i) = c_i(r)$ for examined items i with rating r, viz. $c_i(1) = 0.0$, $c_i(2) = 0.1$, $c_i(3) = 0.2$, $c_i(4) = 0.8$, and $c_i(5) = 1.0$ (chosen to reflect the quadratic gain values of nDCG). Unrated items are not clicked.[6]

4 Results

Table 2 shows the evaluation scores for each RS under the different evaluation approaches. For rating-based evaluation, we see good agreement between system performance under nDCG and Precision. In line with the RS literature, ItemPop achieves the highest scores. IB performs lowest: it recommends a high portion of items for which no ratings are available, a well-known problem. Good agreement is also observed between rating-based precision and CTR under the no-bias model (RSE $= 0.00017$). Thus, when no position bias is present, precision can be used as an almost perfect predictor of CTR.

When position bias is present, agreement with explicit scores can still be high, if the assumptions about user behavior underlying the metric properly reflect user behavior. This can be seen for the *browse-log* and *exam-log* models, which both show good agreement with nDCG. Both models and nDCG assume that users are more likely to examine, and therefore benefit from, highly-ranked items. However, the decay in examination probability is weak (logarithmic). As expected, the fit between explicit metrics and CTR worsens as assumptions about user behavior deviate. This can be seen for the *exam-quad* and *browse-quad* models, where position bias is much stronger than assumed by nDCG.

[5] http://www.grouplens.org/node/73

[6] All our code is open source and available at https://bitbucket.org/ilps/lerot [11].

Table 2. Performance of systems when measured using clicks produced by several click models, compared to the rating-based equivalent. Changes in system rank with respect to rating-based evaluation, and models with the highest RSE per metric are highlighted in **bold.**

	Rating-based		Click-based (CTR)						
	nDCG	Precision	no-bias	exam-log	exam-quad	browse-log	browse-quad	cascade-low	cascade-high
ItemPop	1 (0.150)	1 (0.146)	1 (0.139)	1 (0.081)	1 (0.020)	1 (0.088)	1 (0.020)	1 (0.087)	1 (0.067)
MF	2 (0.043)	2 (0.049)	2 (0.048)	2 (0.023)	**3** (0.001)	2 (0.025)	**3** (0.001)	2 (0.039)	2 (0.034)
UB	3 (0.023)	3 (0.028)	3 (0.026)	3 (0.014)	**2** (0.003)	3 (0.015)	**2** (0.003)	3 (0.022)	3 (0.020)
RND	4 (0.005)	4 (0.006)	4 (0.006)	4 (0.003)	4 (0.000)	4 (0.003)	4 (0.000)	4 (0.006)	4 (0.006)
IB	5 (0.000)	5 (0.000)	5 (0.001)	5 (0.000)	5 (0.000)	5 (0.000)	5 (0.000)	5 (0.001)	5 (0.001)
RSE nDCG			0.00363	0.00083	0.00242	0.00070	0.00242	0.00683	**0.00711**
RSE Precision			0.00017	0.00197	0.00286	0.00254	0.00286	0.00471	**0.00550**

The RSE when predicting CTR under quadratic decay in examination probabilities is up to 3.5 times (0.00254) higher than under the log decay. In addition, two RSs, viz. MF and UB, switch ranks under these models. This means that, if explicit-rating scores were used to decide which of these models to use in production, the decision may not agree with the performance that would be observed under implicit feedback. The worst fit is observed under the cascade model, where the examination probabilities depend on whether users were satisfied with higher-ranked items. Neither the strong position bias nor the dependencies between items are captured by the rating-based metrics.

Fig. 1 examines the fit of CTR under *no-bias*, *cascade-high*, and *browse-quad*. We can see the good agreement between precision and CTR for the *no-bias* model. These setups agree in the magnitude of the scores; the residuals are close to zero for all systems. But if user behavior is best reflected by the cascade model and high satisfaction probability, precision systematically over-estimates system performance. The system for which predicted performance deviates most from the observed CTR is MF, but relatively high residuals are observed for all systems. Performance would be most strongly over-estimated in cases where the *browse-quad* model best reflects user behavior. Under this model, the performance of MF is most strongly affected, with precision over-estimating the score (this reflects that MF is not trained specifically for ranking [6]).

Agreement between explicit rating-based evaluation and click-based evaluation depends on the degree to which assumptions about user behavior are captured by evaluation metrics. We found best agreement between precision and CTR when no position bias is present. Best agreement with nDCG was observed for the *exam-log* and *browse-log* models. Despite the small number of systems compared, we observe changes in system rankings under the *browse-quad* and *exam-quad* user models. The lowest agreement was between explicit ratings and the cascade model.

5 Conclusion

We studied the effect of position bias on RS evaluation using implicit user feedback. We modeled position bias using click models that capture key aspects of user behavior, and compared RS performance under these models to that obtained under traditional, rating-based evaluation. Good agreement between click-based and rating-based evaluation is possible if rating-based metrics properly reflect user behavior. System performance can be under- or over-estimated when user behavior strongly deviates from the assumptions underlying the metrics; this effect was strongest under steep position bias.

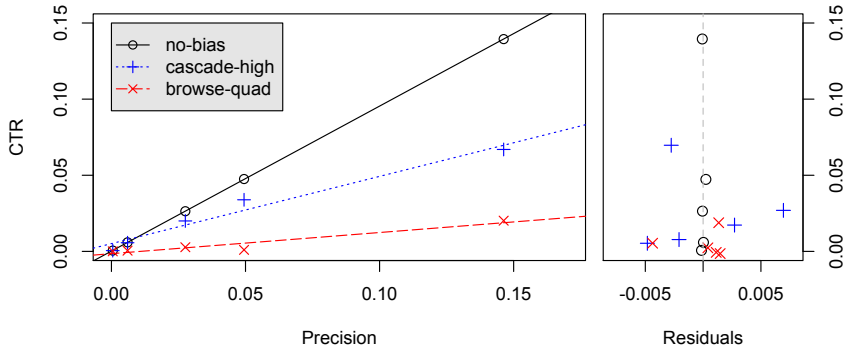

Fig. 1. Precision and CTR scores for selected user models with fitted linear model and residuals.

Our results have implications for theory and practice of rating-based and click-based RS evaluation. First, they highlight the importance of understanding user behavior in deployed systems. Only when interactions are well understood can we design metrics that accurately reflect users' perception of RS quality. A direction for future work is to investigate how bias affects RS when implicit feedback is used for training.

Acknowledgments. This research was supported by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement nr 288024 (LiMoSINe project), the Netherlands Organisation for Scientific Research (NWO) under project nrs 640.004.802, 727.011.005, 612.001.116, HOR-11-10, the Center for Creation, Content and Technology, the QuaMerdes project funded by the CLARIN-nl program, the TROVe project funded by the CLARIAH program, the Dutch national program COMMIT, the ESF Research Network Program ELIAS, the Elite Network Shifts project funded by the Royal Dutch Academy of Sciences, the Netherlands eScience Center under project nr 027.012.105 and the Yahoo! Faculty Research and Engagement Program.

References

[1] A. Bellogín, P. Castells, and I. Cantador. Precision-oriented evaluation of recommender systems: an algorithmic comparison. In *RecSys '11*, pages 333–336, 2011.

[2] N. Craswell, O. Zoeter, M. Taylor, and B. Ramsey. An experimental comparison of click position-bias models. In *WSDM '08*, pages 87–94, 2008.

[3] G. E. Dupret and B. Piwowarski. A user browsing model to predict search engine click data from past observations. In *SIGIR '08*, pages 331–338, 2008.

[4] F. Guo, C. Liu, and Y. M. Wang. Efficient multiple-click models in web search. In *WSDM '09*, pages 124–131, 2009.

[5] K. Hofmann, S. Whiteson, and M. de Rijke. A probabilistic method for inferring preferences from clicks. In *CIKM '11*, pages 249–258, 2011.

[6] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *ICDM '08*, pages 263–272. IEEE Computer Society, 2008.

[7] T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting clickthrough data as implicit feedback. In *SIGIR '05*, pages 154–161. ACM, 2005.

[8] D. Oard and J. Kim. Implicit feedback for recommender systems. In *AAAI Workshop on Recommender Systems*, pages 81–83, 1998.

[9] F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor. *Recommender Systems Handbook*. Springer, 2011.

[10] M. Richardson, E. Dominowska, and R. Ragno. Predicting clicks: estimating the click-through rate for new ads. In *WWW '07*, pages 521–530. ACM, 2007.

[11] A. Schuth, K. Hofmann, S. Whiteson, and M. de Rijke. Lerot: an Online Learning to Rank Framework. In *LivingLab '13*, pages 23–26. ACM, 2013.