




Mitigating Position Bias in Hotels Recommender Systems

Yinxiao Li()

Meta Platforms, Cambridge, USA
liyinxiao1227@gmail.com

Abstract. Nowadays, search ranking and recommendation systems rely on a lot of data to train machine learning models such as Learning-to-Rank (LTR) models to rank results for a given query, and implicit user feedback (e.g. click data) have become the dominant source of data collection due to its abundance and low cost, especially for major Internet companies. However, a drawback of this data collection approach is the data could be highly biased, and one of the most significant biases is the position bias, where users are biased towards clicking on higher ranked results. In this work, we will investigate the marginal importance of properly mitigating the position bias in an online test environment in Tripadvisor Hotels recommender systems. We propose an empirically effective method of mitigating the position bias that fully leverages the user action data. We take advantage of the fact that when a user clicks a result, they have almost certainly observed all the results above, and the propensities of the results below the clicked result will be estimated by a simple but effective position bias model. The online A/B test results show that this method leads to an improved recommendation model.

Keywords: position bias · recommender systems · learning to rank · unbiased learning · implicit feedback

1 Introduction

With an increasing presence of machine learning in search ranking and recommendation systems, there is a larger demand for data than ever before. Implicit user feedback, such as clicks, conversion and dwell time [38], are cheap and abundant compared with explicit human judgements, especially for large Internet companies. Thus, it has become the dominant source of data collection to solve/improve search ranking and recommendation problems. However, a well-known challenge of implicit user feedback is its inherent bias [8].

Implicit user feedback typically include four types of biases: position bias [20, 29, 37], presentation bias [39], exposure bias [30, 36] and quality-of-context bias [17, 19, 42]. Position bias is that users are biased towards clicking on higher

Y. Li—Work performed while at Tripadvisor, USA.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023
L. Boratto et al. (Eds.): BIAS 2023, CCIS 1840, pp. 74–84, 2023.
https://doi.org/10.1007/978-3-031-37249-0_6

ranked results, either due to laziness [20,37] or due to trust to the search site (trust bias) [1,19,29]. Presentation bias describes an effect where users tend to click on results with seemingly more attractive summaries, which inflates the perceived relevance. Exposure bias occurs as users are only exposed to a part of specific items so that the click data used to learn a ranking function are under-sampled and thus biased. The quality-of-context bias refers to the fact that users make click decisions not only by the relevance of the clicked result, but also by the contextual information, such as the overall quality of the results in the list, and their cross-positional interactions. Among the four biases, the position bias has the strongest effect on what users click [21]. Therefore, there is a stronger need to debias it in order to fully leverage the power of implicit user feedback data.

In this work, we will focus on the position bias due to laziness, where users may not have evaluated the whole list before making a click. This is especially true for long lists such as Tripadvisor Hotels recommendation, where a maximum of 30 hotels are displayed on a certain page. We assume that the user has evaluated all the hotels above the lowest ranked hotel that they clicked on. For hotels below that, we will estimate their propensities through a position bias model. We then apply propensity sampling to generate the training data, which is used to train a pairwise model to serve live-site traffic. The effectiveness of this model will be verified by online A/B testing.

2 Related Work

2.1 Position Bias

To use implicit user data for model training, researchers have adopted three primary approaches of dealing with the position bias. The first approach is to keep all the results in the training data and neglect this position bias. This approach assumes that user has evaluated all the options [9,15,35], and it is only acceptable for a relatively short list such as Facebook ad recommendation [15]. The second approach is to only keep results up to the last result user clicked on [12]. This approach assumes that the user sequentially views the results from top to bottom, and will click the first relevant result as they are scrolling down and stop (similar to Cascade model [10]). This works reasonably well for a relatively long list such as Airbnb search ranking [12]. However, it has been argued that this approach is systematically biased and will lead to a ranking model that tends to reverse the existing order [18,21].

The third commonly adopted approach is to keep all the results in the training data, but use the propensities as weights in the loss function [5,21,37]. Compared with the previous two approaches, this approach aims at debiasing the training data by taking propensities into account. They have shown that this method leads to an unbiased loss function and thus an unbiased model, and referred to this framework as unbiased Learning-to-Rank [5,21,37]. However, this approach has not yet fully leveraged the user feedback data (e.g. when a user clicks on result N, this user has almost certainly evaluated result 1 to result N-1). Besides,

this approach requires propensity estimation, which is another challenging task. In our work, we will incorporate the key ideas from both the second and third approaches for mitigating position bias, and provide propensity estimations by fully leveraging the user actions.

2.2 Propensity Estimation

There has been numerous research on unbiased click propensity estimation, and position bias model [34, 37] is one of the most classical methods. The position bias model assumes that the probability of a click on a given result is the product of the probability of evaluating the result and the probability of clicking the result given that it has been evaluated:

$$P(C = 1|i, u, k) = P(E = 1|k) \cdot P(R = 1|i, u) \quad (1)$$

where C represents whether a result is clicked, E represents whether a result is examined, R represents whether a result is relevant, i and u are the item and user (or their feature representation), and k is the position. This model in general requires result randomization experiments which degrade the user experience [21, 36], although many efforts have been spent on minimizing this degradation effect [31].

To fully remove the degradation effect, Wang et al. proposed a method without result randomization, to estimate position bias from regular clicks [37]. This method uses a regression-based Expectation Maximization (EM) algorithm to extract the position bias and result relevance simultaneously. After that, Ai et al. proposed a dual learning method to jointly learn an unbiased ranker and a propensity model [4]. Hu et al. further developed a general framework for jointly estimating the position bias and training a pairwise ranker from click data [16]. However, we argue that this type of method tends to assign relevance based on how relevant a result is compared with other results at the same position k , potentially overlooking the fact that results appearing at top ranks are generally better than those appearing at the bottom.

Later, Aslanyan et al. proposed a method to estimate click propensities without any intervention in the live search results. This method takes advantage of the fact that the same query-document pair may naturally change ranks over time in eCommerce search, and uses query-document pairs that appear at different ranks to estimate propensities [5, 6]. Similarly, Agarwal et al. proposed an estimating method that also requires no intervention, which uses query-document pairs from different ranking functions [2, 3]. Both methods assume that a document will not change much over time and propensities are estimated based on the CTR of the same document at different positions. However, although documents in search engines are relatively static, the price of a hotel is very dynamic and is one of the key factors to consider when users make click/booking decisions, which makes pair generation very difficult for Hotels search.

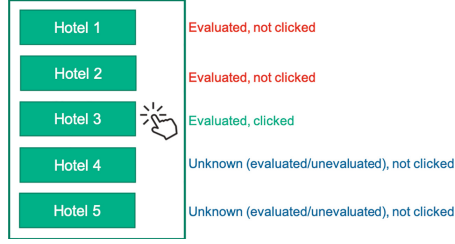


Fig. 1. Demonstration of implicit user click feedback.

3 Method

3.1 Position Bias Mitigation

In this work, we propose to incorporate two of the existing approaches on position bias mitigation (i.e. second and third approaches in Sect. 2.1), to build an unbiased training dataset for recommender systems in Tripadvisor Hotels. Consider an example of implicit user feedback, as shown in Fig. 1, where there are five hotel impressions in the list, and the user makes a click on Hotel 3. Since the user clicks on Hotel 3, Hotel 3 has been evaluated, and we assume Hotel 1 and Hotel 2 are also evaluated [18, 19], while Hotel 4 and Hotel 5 are in an unknown state of being evaluated or not. The sampling rate of each hotel impression is equal to its propensity of being observed, which will be discussed in Sect. 3.2.

Specifically, while constructing the training examples, our approach first discards all the search logs where no click/booking happened, and for the remaining searches, we divide the list based on the lowest hotel position that was clicked/booked by the user: the hotel impressions on or above the lowest clicked/booked position will be kept in the training dataset without sampling, while hotel impressions below that will be sampled based on their estimated propensities. Then, the hotel impressions that remain will be used to create training pairs, as shown in Fig. 2. It is noted that since this approach mitigates the position bias via debiasing the training dataset, it does not require any change on the model training/serving infrastructure or impact the complexity of the model.

3.2 Propensity Estimation

Extending the classical position bias model as shown in Eq. 1, when the user clicks on the hotel at position k_u , they have evaluated all the results on or above k_u :

$$P(E = 1 | k \leq k_u) = 1 \quad (2)$$

On the other hand, the propensity of observing results below the lowest clicked hotel position k_u can be calculated as:

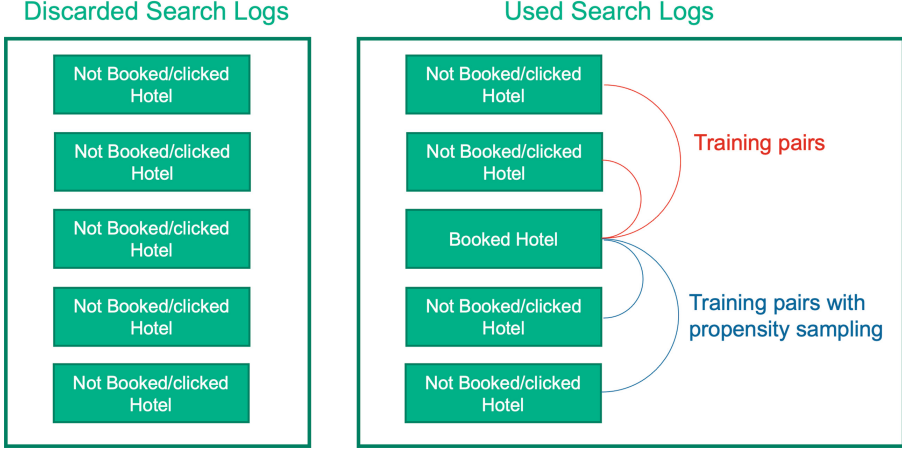


Fig. 2. Construction of training dataset from logged search results.

$$\begin{aligned}
 P(E = 1|k > k_u) &= P(E = 1|k, k_u) \\
 &= \frac{P(E = 1|k)}{P(E = 1|k_u)}
 \end{aligned} \tag{3}$$

Apparently, the precise calculation of $P(E = 1|k > k_u)$ relies on an accurate estimation of $P(E = 1|k)$. As discussed in Sect. 2.2, estimating propensities with the help of result randomization degrades the user experience, while the existing evaluation methods from regular clicks suffer from the difficulty in separating hotel relevance from propensity. Here, we will use a simple relevance assignment strategy based on the historical number of bookings, which will be shown to be good enough for evaluating the average relevance of hotels at a certain position. According to the position bias model, we have:

$$P(E = 1|k) = \frac{E[P(C = 1|i, u, k)]}{E[P(R = 1|i, u, k)]} \tag{4}$$

where we let $E[P(R = 1|i, u, k)] = \text{mean of historical bookings at position } k$. We have found out that for online travel agencies (OTAs), the number of bookings (conversions) is a very strong signal of hotel relevance and is aligned with our final business goal. Figure 3 shows the measured click curve ($P(C = 1|k)$ vs position) and the calculated propensity curve ($P(E = 1|k)$ vs position) based on Eq. 4. The click curve confirms that users are highly biased towards clicking on higher ranked hotels, and since the click curve is steeper than the calculated propensity curve, it indicates that, in general, we are already promoting more relevant hotels to the top of the list.

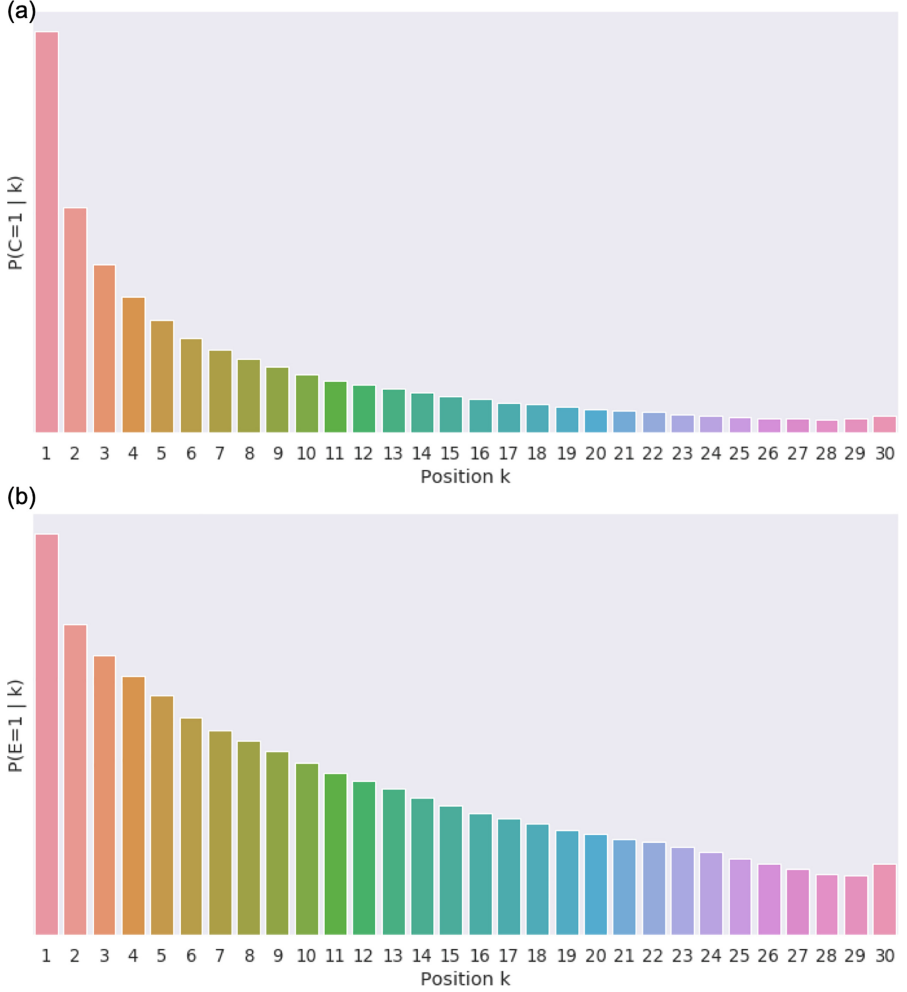


Fig. 3. a) Click curve, and b) propensity curve.

4 Experiments and Results

4.1 Model Implementation

The training dataset of our recommendation model includes search queries in the past 28 d, where at least one click/booking happens. We will be optimizing for bookings, similar to other OTAs [7, 12, 13], but will use clicks as supplemental data to facilitate the training process as we have far more clicks than bookings [12, 22]. Specifically, two different types of clicks will be used: booking page clicks, and hotel review page clicks, as shown in Fig. 4. Moreover, optimizing for bookings to some extent addresses the concern of presentation bias. We use NDCG

as our primary ranking metric, and the labels of hotel impressions are assigned based on the following rules:

- Booking: label = 5
- Click into booking page: label = 2
- Click into hotel review page: label = 1
- No click: label = 0

We then create pairs of hotel impressions whenever their labels do not match, to train a pairwise recommender model.

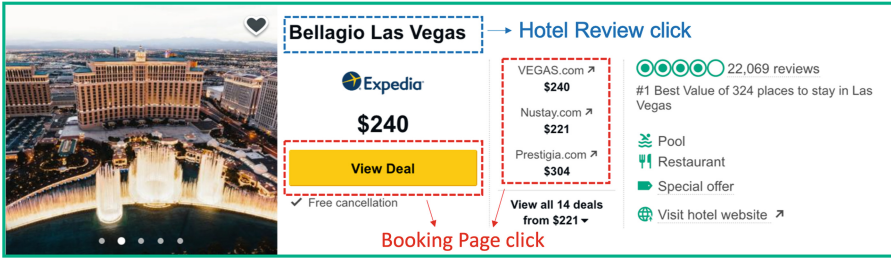


Fig. 4. Booking page clicks vs hotel review page clicks.

We use LGBMRanker from lightgbm library [23] to train a pairwise GBDT model. In industry, both pointwise formulation [9, 26–28, 41] and pairwise formulation [12, 13, 16, 26, 33] are widely used to solve real-world search ranking problems, but in this work we will use pairwise methods due to its three advantages over pointwise methods: (a) Focus on learning relevant things: for example, hotels in Boston have higher CTR than hotels in Maine. A pointwise model is supposed to predict such details correctly, which is unnecessary since we will never compare a hotel in Boston with a hotel in Maine, while pairwise learning will focus on solving problems that you will actually encounter [13]. (b) Quality-of-context bias: users make click decisions not only by the relevance of the clicked result, but also by the overall quality of the surrounding results in the list [19]. The pairwise formulation measures the relative relevance by constructing pairs. (c) Intention bias: for example, hotels on the second page generally have higher CTR than those on the first page because users entering second pages are more engaged. A pointwise model tends to incorrectly favor the best hotels on the second page.

To create personalization features, we trained hotel embeddings using word2vec [24, 25] with within-geo negative sampling, to account for congregated search (i.e. users frequently search only within a certain geographical region but not across regions), and these hotels embeddings were used to generate similarity features based on user behavior for real-time personalization similar to [12]. We modified the gensim library [32] to allow this within-geo negative sampling.

To reduce latency and better serve the online traffic, we use a two-stage ranking algorithm in our online ranking serving system, where an initial simple model retrieves a set of hotels (candidate generation) and a second complex model re-ranks them before presenting to users [11]. This approach allows fast real-time recommendations and has been widely used in Google [9], Facebook [15] and Pinterest [26, 40]. In this work, the two models are trained with the same training dataset.

4.2 Experimental Setup

To evaluate the effectiveness of propensity sampling, we ran three online experiments in Tripadvisor Hotels Best Value Sort for 2 weeks, and evaluated the performance by the number of clicks (both booking page clicks and hotel review page clicks). For these experiments, the sampling rate of results on or above the lowest clicked hotel is kept as 1 (Eq. 2), while three different variants on how to sample results below the lowest clicked hotel will be tested. Specifically, we have chosen the 100% sampling strategy as the control, where all the hotel impressions of a query with at least one click/booking are kept in the training dataset. In Test 1, we use a sampling rate of 80% for the hotels below the lowest clicked position. In Test 2, we apply propensity sampling to hotel impressions based on our estimated propensity of observation.

- Control: 100% sampling, $P(E = 1|k > k_u) = 100\%$.
- Test 1: 80% sampling, $P(E = 1|k > k_u) = 80\%$.
- Test 2: propensity sampling, where $P(E = 1|k > k_u) = \frac{P(E=1|k)}{P(E=1|k_u)}$, and $P(E = 1|k)$ is estimated based on Eq. 4.

Since we use all the raw search logs from the past 28 d to construct the training pairs, the total number of pairs used to train the models in Test 1 and Test 2 is slightly lower than that in Control.

4.3 Results

The online A/B test result is shown in Table 1. Among the three experiments, the model with propensity sampling (Test 2) has the best performance in terms of clicks. Compared with the control model with a 100% sampling rate, it improves the clicks by 1.5%, which is statistically significant (p-value < 0.05), despite a lower number of training pairs used in model training. This model also outperforms the model with a constant sampling rate of 80% (Test 1), by 1.7% in clicks. The model with 80% sampling shows flat results compared with the control model. This result shows that our approach is effective at mitigating position bias, as it has generated more user clicks and engagements in the online test environment.

Table 1. Online test results of three experiments. Gain is relative to control.

Experiment	Clicks
Control: 100% sampling	0.0%
Test 1: 80% sampling	−0.2%
Test 2: propensity sampling	+1.5% (stats sig)

5 Conclusion

Although there is no widely accepted way of correcting position bias in training LTR models, the importance of mitigating such bias should not be overlooked. In this work, we put forward a simple and easily adoptable method that fully leverages user actions with propensity sampling, and prove that it is effective through an online experiment. Online test results show that this method leads to significant performance improvements. Compared with large investments in infrastructures to support more complex models [14], this method requires minimal efforts without a higher level of model complexity, but is still able to improve the recommender system significantly.

References

1. Agarwal, A., Wang, X., Li, C., Bendersky, M., Najork, M.: Addressing trust bias for unbiased learning-to-rank. In: The World Wide Web Conference, pp. 4–14 (2019)
2. Agarwal, A., Zaitsev, I., Joachims, T.: Consistent position bias estimation without online interventions for learning-to-rank. arXiv preprint [arXiv:1806.03555](https://arxiv.org/abs/1806.03555) (2018)
3. Agarwal, A., Zaitsev, I., Wang, X., Li, C., Najork, M., Joachims, T.: Estimating position bias without intrusive interventions. In: Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, pp. 474–482 (2019)
4. Ai, Q., Bi, K., Luo, C., Guo, J., Croft, W.B.: Unbiased learning to rank with unbiased propensity estimation. In: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, pp. 385–394 (2018)
5. Aslanyan, G., Porwal, U.: Direct estimation of position bias for unbiased learning-to-rank without intervention. arXiv preprint [arXiv:1812.09338](https://arxiv.org/abs/1812.09338) (2018)
6. Aslanyan, G., Porwal, U.: Position bias estimation for unbiased learning-to-rank in ecommerce search. In: International Symposium on String Processing and Information Retrieval. pp. 47–64. Springer (2019)
7. Bernardi, L., Mavridis, T., Estevez, P.: 150 successful machine learning models: 6 lessons learned at booking. com. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1743–1751. ACM (2019)
8. Chen, J., Dong, H., Wang, X., Feng, F., Wang, M., He, X.: Bias and debias in recommender system: A survey and future directions. arXiv preprint [arXiv:2010.03240](https://arxiv.org/abs/2010.03240) (2020)
9. Covington, P., Adams, J., Sargin, E.: Deep neural networks for youtube recommendations. In: Proceedings of the 10th ACM Conference on Recommender Systems, pp. 191–198. ACM (2016)

10. Craswell, N., Zoeter, O., Taylor, M., Ramsey, B.: An experimental comparison of click position-bias models. In: Proceedings of the 2008 International Conference on Web Search and Data Mining, pp. 87–94. ACM (2008)
11. Dang, V., Bendersky, M., Croft, W.B.: Two-stage learning to rank for information retrieval. In: Serdyukov, P., Braslavski, P., Kuznetsov, S.O., Kamps, J., Rüger, S., Agichtein, E., Segalovich, I., Yilmaz, E. (eds.) ECIR 2013. LNCS, vol. 7814, pp. 423–434. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36973-5_36
12. Grbovic, M., Cheng, H.: Real-time personalization using embeddings for search ranking at airbnb. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 311–320. ACM (2018)
13. Haldar, M., et al.: Applying deep learning to airbnb search. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1927–1935. ACM (2019)
14. Hazelwood, K., et al.: Applied machine learning at facebook: a datacenter infrastructure perspective. In: 2018 IEEE International Symposium on High Performance Computer Architecture (HPCA), pp. 620–629. IEEE (2018)
15. He, X., et al.: Practical lessons from predicting clicks on ads at facebook. In: Proceedings of the Eighth International Workshop on Data Mining for Online Advertising, pp. 1–9. ACM (2014)
16. Hu, Z., Wang, Y., Peng, Q., Li, H.: Unbiased lambdamart: an unbiased pairwise learning-to-rank algorithm. In: The World Wide Web Conference, pp. 2830–2836 (2019)
17. Jin, J., et al.: A deep recurrent survival model for unbiased ranking. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 29–38 (2020)
18. Joachims, T.: Optimizing search engines using clickthrough data. In: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 133–142. ACM (2002)
19. Joachims, T., Granka, L., Pan, B., Hembrooke, H., Radlinski, F., Gay, G.: Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search. *ACM Trans. Inf. Syst. (TOIS)* **25**(2), 7 (2007)
20. Joachims, T., Granka, L.A., Pan, B., Hembrooke, H., Gay, G.: Accurately interpreting clickthrough data as implicit feedback. In: *Sigir*, vol. 5, pp. 154–161 (2005)
21. Joachims, T., Swaminathan, A., Schnabel, T.: Unbiased learning-to-rank with biased feedback. In: Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, pp. 781–789. ACM (2017)
22. Karmaker Santu, S.K., Sondhi, P., Zhai, C.: On application of learning to rank for e-commerce search. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 475–484. ACM (2017)
23. Ke, G., et al.: Lightgbm: a highly efficient gradient boosting decision tree. In: *Advances in Neural Information Processing Systems*, pp. 3146–3154 (2017)
24. Li, Y., Anderson, J.: Introducing openstreetmap user embeddings: promising steps toward automated vandalism and community detection (2021)
25. Li, Y., Anderson, J., Niu, Y.: Vandalism detection in openstreetmap via user embeddings. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, pp. 3232–3236 (2021)
26. Liu, D.C., et al.: Related pins at pinterest: the evolution of a real-world recommender system. In: Proceedings of the 26th International Conference on World

- Wide Web Companion, pp. 583–592. International World Wide Web Conferences Steering Committee (2017)
27. Ma, X., et al.: Entire space multi-task model: an effective approach for estimating post-click conversion rate. In: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, pp. 1137–1140. ACM (2018)
 28. Naumov, M., et al.: Deep learning recommendation model for personalization and recommendation systems. arXiv preprint [arXiv:1906.00091](https://arxiv.org/abs/1906.00091) (2019)
 29. O’Brien, M., Keane, M.T.: Modeling result-list searching in the world wide web: the role of relevance topologies and trust bias. In: Proceedings of the 28th annual conference of the cognitive science society, vol. 28, pp. 1881–1886. Citeseer (2006)
 30. Ovaisi, Z., Ahsan, R., Zhang, Y., Vasilaky, K., Zheleva, E.: Correcting for selection bias in learning-to-rank systems. In: Proceedings of The Web Conference 2020, pp. 1863–1873 (2020)
 31. Radlinski, F., Joachims, T.: Minimally invasive randomization for collecting unbiased preferences from clickthrough. In: Logs, Proceedings of the 21st National Conference on Artificial Intelligence (AAAI). Citeseer (2006)
 32. Řehůřek, R., Sojka, P.: Software framework for topic modelling with large corpora. In: Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks, pp. 45–50. ELRA, Valletta, Malta, May 2010. <http://is.muni.cz/publication/884893/en>
 33. Ren, Y., Tang, H., Zhu, S.: Unbiased learning to rank with biased continuous feedback. In: Proceedings of the 31st ACM International Conference on Information & Knowledge Management, pp. 1716–1725 (2022)
 34. Richardson, M., Dominowska, E., Ragno, R.: Predicting clicks: estimating the click-through rate for new ads. In: Proceedings of the 16th international conference on World Wide Web, pp. 521–530. ACM (2007)
 35. Tagami, Y., Ono, S., Yamamoto, K., Tsukamoto, K., Tajima, A.: Ctr prediction for contextual advertising: Learning-to-rank approach. In: Proceedings of the Seventh International Workshop on Data Mining for Online Advertising, p. 4. ACM (2013)
 36. Wang, X., Bendersky, M., Metzler, D., Najork, M.: Learning to rank with selection bias in personal search. In: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 115–124 (2016)
 37. Wang, X., Golbandi, N., Bendersky, M., Metzler, D., Najork, M.: Position bias estimation for unbiased learning to rank in personal search. In: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, pp. 610–618. ACM (2018)
 38. Yi, X., Hong, L., Zhong, E., Liu, N.N., Rajan, S.: Beyond clicks: dwell time for personalization. In: Proceedings of the 8th ACM Conference on Recommender Systems, pp. 113–120. ACM (2014)
 39. Yue, Y., Patel, R., Roehrig, H.: Beyond position bias: examining result attractiveness as a source of presentation bias in clickthrough data. In: Proceedings of the 19th International Conference on World Wide Web, pp. 1011–1018. ACM (2010)
 40. Zhai, A., et al.: Visual discovery at pinterest. In: Proceedings of the 26th International Conference on World Wide Web Companion, pp. 515–524. International World Wide Web Conferences Steering Committee (2017)
 41. Zhou, G., et al.: Deep interest network for click-through rate prediction. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1059–1068. ACM (2018)
 42. Zhuang, H., et al.: Cross-positional attention for debiasing clicks. In: Proceedings of the Web Conference 2021, pp. 788–797 (2021)