Correcting for Recency Bias in Job Recommendation

Ruey-Cheng Chen
RMIT University
Melbourne, Australia

Qingyao Ai
School of Computing, University of Utah
Salk Lake City, UT

Gaya Jayasinghe
GO1
Melbourne, Australia

W. Bruce Croft
RMIT University
Melbourne, Australia

ABSTRACT

Users are known to interact more with fresh content in certain temporally associated domains such as news search or job seeking, leading to an uneven distribution of interactions over items of different degrees of freshness. Data collected under such an "aging effect" is usually used unconditionally on all sort of recommendation tasks, and as a result more recently published content may be over-represented during model training and evaluation. In this study, we characterize this temporal influence as a *recency bias*, and present an analysis in the domain of job recommendation. We show that, by correcting for recency bias using an unbiased learning to rank approach, one can improve the quality of recommendation significantly over a recent neural collaborative filtering model on RecSys Challenge 2017 data.

ACM Reference Format:
Ruey-Cheng Chen, Qingyao Ai, Gaya Jayasinghe, and W. Bruce Croft. 2019. Correcting for Recency Bias in Job Recommendation. In *The 28th ACM International Conference on Information and Knowledge Management (CIKM '19), November 3–7, 2019, Beijing, China.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3357384.3358131

1 INTRODUCTION

In news search, users tend to interact more with recently posted articles [17] than stories in the backlog. Similar influences are seen across different domains where new contents are to be pushed to users periodically. This preference towards fresh content, which is orthogonal to the more accustomed topical relevance, can usually be captured by directly learning from implicit user interactions [8]. However, raw user interactions collected under this temporal influence can also exhibit a bias in favor of more recently published content. Various approaches have been proposed [5, 11, 13] to incorporate recency into ranking, but adjustment for such biases in the data has not been explored.

In this paper, we present a case study of the *recency bias* in the domain of job recommendation. Job recommendation [10, 12] is the task of recommending job advertisements (job ads) to the potential candidates (users). Job ads have a shorter lifespan than documents in ad hoc retrieval. New job positions are advertised on a daily basis, and depending on the area a position can be filled in a matter of few weeks. Users compete for the advertised jobs, and hence prefer to apply for job ads as soon as they are discovered. Our analysis shows that recent job ads garner more clicks and have a higher clickthrough rate compared to older job ads. Therefore, interactions recorded at the system side may suffer from this uneven distribution of content freshness, leading to an inherent recency bias in the data.

We conjecture that co-interaction patterns that do not involve recent items will have lower weights despite the fact that they might be indicative of relevance. Consequently, relevant job connections across longer periods of time are less likely to be discovered. In job recommendation, job seekers are driven by their long-term career preferences so recommendations consistent to their long-term behavior are more favorable [10].

In this paper, we argue that addressing recency bias has a positive impact for job recommendation. As an example, we present an approach based on unbiased learning to rank to remove the influence of recency bias when training a recommender system. Our experiments conducted on RecSys Challenge 2017 Data shows that correcting for recency bias can lead to improved recommendation effectiveness of a recent neural collaborative filtering model in both early NDCG and Hit Rate.

2 RELATED WORK

Recency Ranking. The idea of recency ranking emerged from the context of web search [6]. The mainstream methodology is largely "Cranfield" based: Editorial judgments are solicited from experts, and relevance labels are demoted manually by 1 or 2 grades according to how quickly the perceived document relevance decreases. To incorporate recency into ranking, a popular approach is to directly promote recent documents in the ranking [5, 11, 13].

Our work can be seen as an extension to this line of research. We take a different stand in this paper on how recency signals should be best treated prior to model training, and argue that using untreated signals may be suboptimal on similar tasks. Many recommender systems rely on using clickthrough data as the main input source. Clicks are indicative of the underlying document relevance but just like any other type of data they are subject to various forms of biases [15, 16]. In prior work it has been shown that such signals can be used to learn effective ranking functions when biases in data are properly addressed [15, 20]. Examples of such work can be seen in Ai et al. [2], where an unbiased ranker is learned using click data which are missing not at random due to presentation bias.

CIKM '19, November 3–7, 2019, Beijing, China

ACM ISBN 978-1-4503-6976-3/19/11…$15.00
https://doi.org/10.1145/3357384.3358131

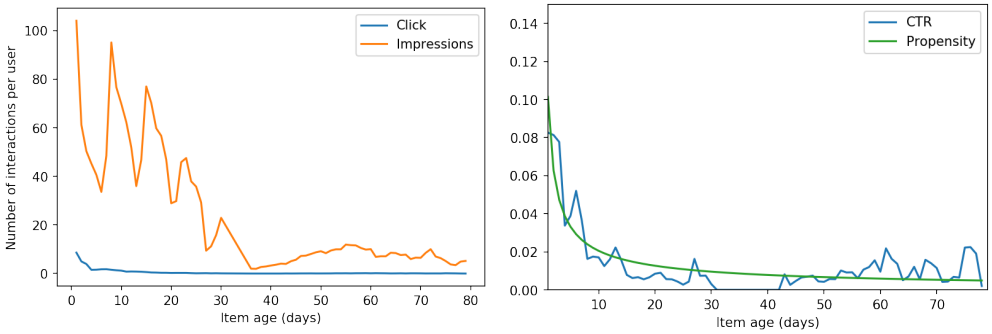

Figure 1: Number of interactions (left) and clickthrough rate (right) plotted against item age.

Counterfactual Learning. Recent advances on counterfactual learning algorithms have attracted a lot of attention in the studies of unbiased learning to rank. Proposed by Wang et al. [18] and Joachims et al. [9], the idea of counterfactual learning is to conduct unbiased learning to rank by weighting individual training instances with the respective inverse propensity weights. Previous studies have shown that presentation bias (e.g., users tend to click more on the top of a page than the bottom) can be effectively removed by weighting user clicks with an examination probability on each result position estimated from online randomization experiments [9, 18] or dual learning algorithm [2]. Inspired by the fact that users exhibit similar preference on more recent job posts, we hypothesize that recency could be another type of selection bias and may be treated using similar methodology to remove the unwanted effect from existing recommender systems.

3 RECENCY BIAS: A CASE STUDY

We present an analysis in Figure 1 to demonstrate how item recency might influence users' behavior. The analysis is based on job recommendation data from the RecSys Challenge 2017 dataset [1], which will be used as a running example in this paper. The dataset consists of interaction records about job seekers (users) interacting with job advertisements (items). The type of recorded interaction can be roughly divided into three categories: (i) *impression*, meaning that the job ad has been shown to the user in a recommendation module; (ii) *clickthrough*, meaning that the user clicks on the job ad to see the detailed content; (iii) high-level user actions such as the user putting in an application or bookmarking the job ad. Further details about the data are given in Section 3.2.

Figure 1 shows that, by referring to an item with its age, there is a clear trend that more recently published items attract more clickthrough and user interactions. Number of impressions decreases as the item becomes less recent (leftmost plot), and some pattern of periodicity (i.e. the sawtooth-shaped part) can be seen from the impression curve, demonstrating the "weekday" effect.[1] As opposed to impressions, clickthrough appears to be mildly decreasing roughly five days after the publishing event. The rightmost plot however shows a clear pattern that clickthrough rate is seemingly exponentially decreasing as the job item ages. Job ads can become "too old" to be noticed by most job seekers in about a week, despite the fact that the job positions might remain open for another couples of weeks. So unlucky recruiters have to work around this issue by reposting these ads, a situation that can be avoided if the system manages to counter or reduce the recency bias.

[1]New job ads are generally posted in bulk on a weekly basis (e.g. on Mondays), leading to peaked user activities early in the week. This craze would gradually cool down, exhibiting a diminishing returns in the logged datastream.

3.1 Unbiased Learning to Rank

Leveraging *implicit* user feedback is a common approach for collaborative filtering [8]. In job recommendation, such feedback is usually recorded as a tuple (u, x, y, t) meaning that user u performs some action y on item x at time t. Consider that y is a binary variable indicating whether a positive interaction (i.e. clickthrough) has taken place. Following prior work on unbiased learning to rank [2, 15, 18], the relationship between recorded interactions and the underlying document relevance is defined as:

$$y = o_{uxt}\, r_{ux} \qquad \text{for } u, x, t, \tag{1}$$

where o_{uxt} and r_{ux} are both binary indicators, with the first denoting if item x is "observed" by user u at time t, and the second denoting if item x is deemed relevant by user u. In line with the formulation in prior work [15, 18], relevance variable r_{ux} does not depend on time t, which reflects the prior belief that topical relevance does not change over time. This can be further formalized in a probabilistic sense for any given (u, x) pair as follows:

$$\begin{aligned} \Pr(y = 1|t) &= \Pr(o_{uxt} = 1|t)\ \Pr(r_{ux} = 1|t) \\ &\propto \Pr(o_{uxt} = 1|t) \end{aligned} \tag{2}$$

This allows one to estimate recency propensity directly from the observed clickthrough rate data, which departs from approaches taken in prior work where online result randomization is employed to estimate the positional bias [9, 18].

We then define an unbiased loss using inverse propensity weighting (IPW) [9, 18], to be used in our neural collaborative filtering system. [7]. Binary cross-entropy (BCE) is commonly used in such neural collaborative filtering models as the loss function, but unfortunately, constructing a theoretically principled unbiased version over this pointwise formulation is difficult [2, 9, 18]. Thus, in this paper, we propose an empirical method to adopt the idea of IPW and formulate a new BCE-based pointwise loss function as the following:

$$-\sum_{(x,y,t)\in\mathcal{D}} \left[\frac{y}{\hat{p}_{xt}} \log \sigma(f(x)) + \frac{1-y}{\hat{p}_{xt}} \log(1 - \sigma(f(x))) \right]. \tag{3}$$

Here, $f(x)$ denotes the model score and $\hat{p}_{xt}$ is the recency propensity of item x at time t, namely $\Pr(o_{uxt} = 1|t)$ in Equation (2).

To the best of our knowledge, we are the first to propose such an IPW loss in training neural collaborative filtering models. Although the proposed loss function is not theoretically guaranteed to remove the bias completely [9], it follows the basic motivation of IPW and reduces the weights of training examples that are likely to be affected by the biased user behaviors. In our experiments, this formulation has been proved to be robust and effective at alleviating the recency bias in NeuMF.

3.2 Experiments

The experiment was conducted on the RecSys Challenge 2017 dataset [1]. As the original test data (close set) is not made available to general public, we split the training set into three parts (train/valid/test) for all our experiments.

Textual content materials are not used in this study as our focus is on the user-item interactions. The interaction data is available as a list of interaction records, each of the form: user ID, item ID, interaction type, and timestamp. The challenge data offers 5 levels of interactions, from 0 to 5: impression, click, bookmark, apply, delete, and recruiter action [1]. We re-coded the data by assigning label 0 to impression/delete, 1 to click and all other actions. The original training stream was then broken down into three consecutive periods using time-based splitting: The latest 7 days are deemed as the test period, the 7 days preceding the test period as the validation period, and all other as the training period.

Note that job ads in the training/validation set are removed from the test set. So performance metrics are affected only by test-period items that the user actually interacted with. When making recommendations, the system takes all items into account — the protocol is enforced to prevent unnecessary information leak (i.e. "TrainItems" strategy [14]). Considering on average a job ad is taken down in 3 weeks, we chose a 7-day test window to investigate the effect of recommending active job ads that might be slightly dated (within 2 weeks) but not to the extent of being called expired. Cold-start users were removed from our experiments as its treatment is beyond the scope of this work. Specifically, a filter was set up to retain only users that had seen at least 20 impressions in each of the three periods.[2] As a result, we ended up having a moderate sized subset of the original data, with 2024 users and the respective 99724 items on file. In total, there are 2723K interactions in the training period, 477K in the validation period, and 103K in the test period.

As the raw data contains true impressions which is close to a production setting, the evaluation is set up as a simple reranking task. Job recommender systems are first trained and optimized on the training and validation sets, and then evaluated with their reranking performance on the test set. Note that the raw data contains multiple interactions towards the same item (e.g. job ad Y is shown to user X five times before it generating two clicks leading to one final apply), so for the purpose of item reranking only the "most positive" label of an item is used.

Propensity Estimation. We approximate the creation time of an item by its first observed user interaction across all three sets. A new attribute for each interaction "item age" (in days) can be calculated accordingly. As recency bias in our formulation is a type of selection bias imposed by the users, it is not clear how conventional approach such as randomization or click models can be used to estimate recency propensity. Instead, based on Equation (2), we used the following simplistic propensity model in our study:

$$\text{click}(A(x,t)) \sim \alpha\, A(x,t)^{-\beta} + \gamma, \qquad \hat{p}_{xt} = \text{click}(A(x,t)) \tag{4}$$

where $\alpha, \beta, \gamma \geq 0$ are model weights, and $A(x,t)$ the age of item x in days at time t. The propensity is estimated by first fitting a model $\text{click}(\cdot)$ to the clickthrough rate data as in Figure 1 (rightmost plot) using a least square method. For robustness, we used a trust region reflective method with an exact solver to fit the model [4].

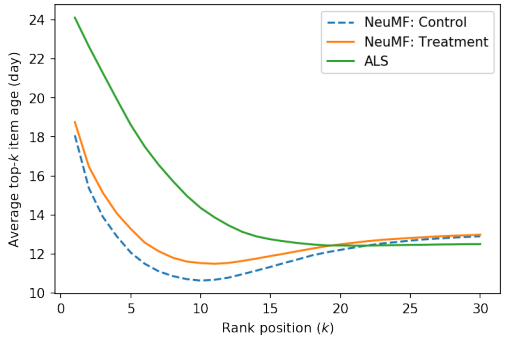

Figure 2: The average item age (day) of the top-k recommendation results on different rank positions.

Control / Treatment Models. Our experiment involves one control model and one treatment model. The control model is trained on ordinary binary cross entropy, and the treatment model is trained on the IPW BCE loss (3). Note that both models are of the same capacity, under the same configuration.

We use the NeuMF model [7] to serve as the base system. The NeuMF model is considered to be one of the most representative methods in recommendation on implicit feedback data [3, 19]. It combines two sub-networks, both built on top of user/item embeddings, with the first network implementing a generalized matrix factorization approach and the second leveraging feed-forward network to approximate the target labels.

To strengthen the baseline, we used true impressions as negative instances rather than randomly drawn samples, and employed layer-wise weight decay and mini-batch training over ranked lists. Our final configuration of the NeuMF model was set to using 4 hidden layers, with 64/32/16/8 nodes respectively, 256 dimensions for both user/item embeddings, and a batchsize of 10 ranked lists. The model is optimized using 100 epochs and early stopped if no improvement is seen in 20 consecutive epochs. For the weight decay at each layer, we set the regularization factor to 0.01.

For comparison, a standard collaborative filtering algorithm ALS [8] is also included as a reference. While some tuning was done over a simplistic confidence model $C_{ux} = 1 + \alpha y$ taking into account the original interaction level, we found that uniform confidence (i.e., $\alpha = 0$) gives the best result. The resultant ALS model was set to using 100 hidden dimensions and a regularization factor of 0.01.

[2]Note that requiring less impressions in the filter (such as 5 impressions) did not change the conclusion except for the lengthened training/inference time.

Table 1: Correcting for recency bias leads to improved early NDCG and HitRate for job recommender systems. */ denote significant differences over ALS, and †/‡ over the NeuMF control group for $p < 0.05$ and $p < 0.01$ respectively.**

	NDCG@5	NDCG@10	HitRate@5	HitRate@10
ALS	0.4793	0.5801	0.8454	0.9427
NeuMF: Control	0.5055**	0.5929*	0.8903**	0.9363
NeuMF: Treatment	**0.5383****‡ (+6.5%)**	**0.6210****‡ (+4.7%)**	**0.9027****† (+1.4%)**	**0.9496**‡ (+1.4%)**

3.3 Results

The experimental results are given in Table 1. Both the control and the treatment group NeuMF models are found more effective than the ALS model, and the differences on NDCG@5, NDCG@10, and HitRate@5 are significant. Our results in Table 1 show that switching to the IPW BCE loss function (3) can lead to significantly increased performance on NDCG and Hit Rate for the neural collaborative filtering model. The effect on early NDCG is more pronounced: the treatment leads to a significant increase of 6.5% in NDCG@5 and 4.7% in NDCG@10 respectively, relative to the control model. HitRate also saw significant increases but the gain is relatively mild (around 1.4%) compared to NDCG.

Further, we investigated the influence of the treatment model on top-k recommended items. Figure 2 shows the average item ages on each rank position for the results produced by the control and treatment models. As we can see, the treatment model successfully recommends more older job items to the user, pushing back top-10 average item age by 1 day compared to the control model. This result suggests that the proposed approach indeed reduces the bias of recency in the treatment model, and in turn improves recommendation effectiveness.

Also from the same plot, we note that ALS tends to recommend older items up front and behaves quite differently from NeuMF. For smaller cutoff such as $k \leq 10$, the items recommended by ALS are on average between 4 to 6 days older than those recommended by either NeuMF model. Since item timestamps are not used in model training for both ALS and NeuMF (except for the treatment), one possible explanation for this phenomenon is that ALS does not fit the data well due to its limited model capacity (even fully optimized), which makes it less likely to observe the same recency bias in ALS as it is in NeuMF and the clickthrough data. Further investigation is needed to formally validate the hypothesis.

4 CONCLUSION

In various search and recommendation tasks, recency has come to be an essential factor in modeling item relevance and it usually requires special treatment in downstream applications. In this paper we provide a different angle into the resolution of this issue: Treating recency as a kind of user-side selection bias. Through this study we develop an unbiased learning to rank approach for correcting the recency bias, and propose a new inverse propensity weighted loss function. Our treatment model is shown to significantly improve early NDCG and Hit Rate on the RecSys Challenge 2017 data, which indicates that removing recency bias could be beneficial for job recommendation. Despite the potential, our results are currently limited to the use of neural collaborative filtering method and the scope of the offline evaluation data. It is also not clear how this bias correction approach might influence user satisfaction. In future work, we aim to address these issues in a broader domain under an online A/B experimental setting.

5 ACKNOWLEDGEMENTS

The authors thank Damiano Spina for the access to data.

REFERENCES

[1] Fabian Abel, Yashar Deldjoo, Mehdi Elahi, and Daniel Kohlsdorf. 2017. Recsys challenge 2017: Offline and online evaluation. In *Proceedings of RecSys '17*. ACM, 372–373.
[2] Qingyao Ai, Keping Bi, Cheng Luo, Jiafeng Guo, and W Bruce Croft. 2018. Unbiased Learning to Rank with Unbiased Propensity Estimation. In *Proceedings of SIGIR '18*. ACM, 385–394.
[3] Ting Bai, Ji-Rong Wen, Jun Zhang, and Wayne Xin Zhao. 2017. A neural collaborative filtering model with interaction-based neighborhood. In *Proceedings of CIKM '17*. ACM, 1979–1982.
[4] Mary Ann Branch, Thomas F Coleman, and Yuying Li. 1999. A subspace, interior, and conjugate gradient method for large-scale bound-constrained minimization problems. *SIAM Journal on Scientific Computing* 21, 1 (1999), 1–23.
[5] Na Dai and Brian D Davison. 2010. Freshness matters: in flowers, food, and web authority. In *Proceedings of SIGIR '10*. ACM, 114–121.
[6] Anlei Dong, Yi Chang, Zhaohui Zheng, Gilad Mishne, Jing Bai, Ruiqiang Zhang, Karolina Buchner, Ciya Liao, and Fernando Diaz. 2010. Towards recency ranking in web search. In *Proceedings of WSDM '10*. ACM, 11–20.
[7] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of WWW '17*. International World Wide Web Conferences Steering Committee, 173–182.
[8] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *Proceedings of ICDM '08*. IEEE, 263–272.
[9] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2017. Unbiased learning-to-rank with biased feedback. In *Proceedings of WSDM '17*. ACM, 781–789.
[10] Krishnaram Kenthapadi, Benjamin Le, and Ganesh Venkataraman. 2017. Personalized job recommendation system at linkedin: Practical challenges and lessons learned. In *Proceedings of RecSys '17*. ACM, 346–347.
[11] Xiaoyan Li and W Bruce Croft. 2003. Time-based language models. In *Proceedings of CIKM '03*. ACM, 469–475.
[12] Ioannis Paparrizos, B Barla Cambazoglu, and Aristides Gionis. 2011. Machine learned job recommendation. In *Proceedings of RecSys '11*. ACM, 325–328.
[13] Maria-Hendrike Peetz and Maarten De Rijke. 2013. Cognitive temporal document priors. In *Proceedings of ECIR '13*. Springer, 318–330.
[14] Alan Said and Alejandro Bellogín. 2014. Comparative recommender system evaluation: benchmarking recommendation frameworks. In *Proceedings of RecSys '14*. ACM, 129–136.
[15] Tobias Schnabel, Adith Swaminathan, Ashudeep Singh, Navin Chandak, and Thorsten Joachims. 2016. Recommendations as treatments: Debiasing learning and evaluation. *Proceedings of ICML '16*.
[16] Harald Steck. 2013. Evaluation of recommendations: rating-prediction and ranking. In *Proceedings of RecSys '13*. ACM, 213–220.
[17] Hongning Wang, Anlei Dong, Lihong Li, Yi Chang, and Evgeniy Gabrilovich. 2012. Joint relevance and freshness learning from clickthroughs for news search. In *Proceedings of WWW '12*. ACM, 579–588.
[18] Xuanhui Wang, Michael Bendersky, Donald Metzler, and Marc Najork. 2016. Learning to rank with selection bias in personal search. In *Proceedings of SIGIR '16*. ACM, 115–124.
[19] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *Proceedings of SIGIR '19*. ACM, 165–174.
[20] Longqi Yang, Yin Cui, Yuan Xuan, Chenyang Wang, Serge Belongie, and Deborah Estrin. 2018. Unbiased offline recommender evaluation for missing-not-at-random implicit feedback. In *Proceedings of RecSys '18*. ACM, 279–287.