# Online Learning to Rank: Absolute vs. Relative

Yiwei Chen
University College London
yiwei.chen.13@ucl.ac.uk

Katja Hofmann
Microsoft Research
katja.hofmann@microsoft.com

## ABSTRACT

*Online learning to rank* holds great promise for learning personalized search result rankings. First algorithms have been proposed, namely *absolute* feedback approaches, based on contextual bandits learning; and *relative* feedback approaches, based on gradient methods and *inferred preferences* between complete result rankings. Both types of approaches have shown promise, but they have not previously been compared to each other. It is therefore unclear which type of approach is the most suitable for which online learning to rank problems. In this work we present the first empirical comparison of absolute and relative online learning to rank approaches.

## Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

## Keywords

Information retrieval; Online learning; Learning to rank

## 1. INTRODUCTION & RELATED WORK

Learning from user interactions is becoming increasingly important in Web information retrieval (IR), as it enables information systems to provide personalized results. For example, search engines could learn preferences for retrieved documents, and recommender systems could adapt to users' tastes. Exploiting user interactions to improve the performance of search systems has been studied from many perspectives. However, it is challenging, as user interactions are typically biased and noisy [6].

Recently, *bandit algorithms* have been explored as a basis for learning from user interactions in a principled way [3]. Particularly promising are *contextual* bandit algorithms, which can integrate information about the document, query, or user context in the form of context features [4]. They learn a parameterized function of these context features, which allows them to generalize learned solutions to e.g., previously unseen query-document pairs. Current contextual bandit algorithms learn from *absolute* interpretations of user feedback, to optimize, e.g., click-through rate (CTR). An alternative approach has been developed on the basis of interpreting user feedback as *relative* preferences between rankings [2]. The resulting signal has been successfully used as a basis for stochastic gradient techniques [1]. Both types of approaches have shown promising results, but their relative performance has not been examined.

This work presents the first empirical comparison between absolute and relative online learning to rank approaches for IR. It addresses the following questions, designed to improve our understanding of the relative performance and of these approaches. **Q1**: How do absolute and relative approaches compare in terms of online performance on standard IR learning to rank tasks? **Q2**: How are both types of approaches affected by noise in user interactions? **Q3**: How do they perform in settings that **(a)** require generalization across queries, and **(b)** do not require such generalization? Our answers to these questions show that different approaches should be used for different learning to rank settings. This has important implications for practical applications, and for the future development of more effective online learning to rank approaches.

## 2. METHODS

We focus on two online learning approaches that exemplify learning from *absolute* and *relative* feedback. Both assume context information is observed in the form of feature representations of query-document pairs, and learn linear ranking models from user interactions. As characteristic for the bandit learning setting, the learner only observes feedback on actions (e.g. documents) it has presented to the user, resulting in a partial feedback setting [3]. The key to effective learning in this setting is to balance exploration of potential new solutions with exploitation of solutions learned so far.

**Absolute approach.** We present Lin-$\epsilon$, an $\epsilon$-greedy version of LinUCB [4]. LinUCB learns linear combinations of ranking features to optimize absolute metrics, e.g., CTR. Our Lin-$\epsilon$ approach learns models of the same form, and uses the same model updates as LinUCB, but uses the simpler $\epsilon$-greedy strategy, a standard exploration scheme for online learning approaches that has been found to perform well and robustly in practice [3]. Lin-$\epsilon$ is outlined in Algorithm 1. In each round, it observes the context features and estimates rewards for each action based on the current ranking models. $generate\_list(\epsilon, \kappa)$ then fills a list of length $\kappa$ slot-by-slot, each with a $1-\epsilon$ probability to pick the document with the next-highest reward estimate, and an $\epsilon$ probability to pick one uniformly at random. Following LinUCB, we experiment with two variants. Lin-$\epsilon$(disjoint) learns distinct models for each document. Lin-$\epsilon$(hybrid) uses a joint component to generalize across documents and queries.

**Relative approach.** We use the state-of-the-art method for online learning to rank from relative feedback, Candidate Preselection (CPS) [1], outlined in Algorithm 2. CPS learns from relative ranker comparisons obtained through *interleaving*. To optimally use observed samples it uses observations collected in previous rounds to select a promising candidate ranker for the next round. CPS learns in rounds, too. After observing context features, a promising can-


*WWW 2015 Companion,* May 18–22, 2015, Florence, Italy.
ACM 978-1-4503-3473-0/15/05.
http://dx.doi.org/10.1145/2740908.2742718.

didate ranker is selected. $generate\_list(\cdot)$ returns a ranked result list given the weight vector. The result lists for the current and the candidate weight vectors are combined, presented to the user, and interactions are projected back to the result lists to infer a user preference. The current ranker is then updated accordingly.

**Input** : exploration rate $\epsilon$, result list length $\kappa$, initial model
**for** query $q_t (t = 0 \dots T)$ **do**
  Estimate rewards given observed context
  $\mathbf{l} \leftarrow generate\_list(\epsilon, \kappa)$
  Present $\mathbf{l}$ to user and observe absolute rewards
  Update models, following [4]
**end**

**Algorithm 1:** Lin-$\epsilon$: $\epsilon$-greedy strategy with linear models.

**Input** : CPS parameters $\theta$, result list length $\kappa$, initial weights $\mathbf{w}_0$
**for** query $q_t (t = 0 \dots T)$ **do**
  Generate ranker pool and select best candidate $\mathbf{w'}_t$
  $\mathbf{l}_1, \mathbf{l}_2 \leftarrow generate\_list([\mathbf{w}_t, \mathbf{w'}_t], \kappa)$
  Infer ranker preference
  Update model: $\mathbf{w}_{t+1} \leftarrow (\mathbf{w}_t \text{ wins ? } \mathbf{w}_t : \mathbf{w'}_t)$
**end**

**Algorithm 2:** Candidate preselection (CPS), following [1].

# 3. EXPERIMENTS

Our experiments are designed to address questions **Q1**-**Q3** (Section 1). They are based on the open source evaluation framework Lerot [7], a standard evaluation setup for online learning to rank methods that uses annotated query-document data and models of user interactions (which reflect, e.g., click noise and bias). Following [1], we experiment at three levels of noise and bias: *perfect* (no noise or bias), *navigational* (little noise, high bias) and *informational* (high noise, medium bias). Our main metric is *online performance* – the discounted cumulative reward of the results shown to the simulated user (in terms of (a) NDCG@10 and (b) CTR@1) [7]. We use offline performance for additional analysis.

Given this setup, we conduct two sets of experiments, as follows. **Experiment 1 (general)** addresses **Q1** and **Q2**. It examines learning performance across queries and requires generalization across queries. Data: NP2003 (named page finding) LETOR 3.0 data [5]. **Experiment 2 (focused)** addresses **Q2** and **Q3**, by examining learning for specific repeated queries. Data: 10 queries sampled at random from the TD2003 (topic distillation) LETOR 3.0 data [5].

Each experiment compares three approaches: (1) disjoint Lin-$\epsilon$, (2) hybrid Lin-$\epsilon$ (both Algorithm 1), (3) CPS (Algorithm 2), all with standard parameters as reported in [4] and [1].

# 4. RESULTS

The results of our experiments are shown in Table 1 and Table 2. We see that CPS performs best on the general task and Lin-$\epsilon$ (disjoint or hybrid) does better in the focused task (**Q1**). This is consistent with our expectations regarding the need for generalization. In the general task, Lin-$\epsilon$(disjoint) shows significantly lower performance on all user models and metrics. It performs much better in the focused task, which also indicates that the lack of generalization results in the need of sufficient feedback (**Q3**). Comparing across different user models, CPS shows the best robustness to noise (**Q2**). We plot the offline performance under the navigational user model from the first experiment in Figure 1, which shows that Lin-$\epsilon$(hybrid) reaches the highest performance after enough interactions (training, it requires approximately 10 times more samples than CPS), but again does not generalize well to completely new queries (test).

# 5. CONCLUSION

We have presented a first empirical comparison of absolute and relative online learning to rank for IR approaches. We found that, while absolute approaches can be more effective when reliable feedback can be inferred, and in cases with few queries and documents (e.g., standing queries, recommendation), relative approaches are more robust to noisy feedback and can deal with larger document spaces. An urgent direction for future work is to extend current linear learning approaches with online learning to rank algorithms that can effectively learn more complex models.

**Table 1: Online performance, *general* experiment for perfect (per), navigational (nav) and informational (inf) user models. Best scores are shown in bold. Statistically significant differences with CPS are indicated by $^{\triangle}/^{\triangledown}(p = 0.05)$, $^{\blacktriangle}/^{\blacktriangledown}(p = 0.01)$.**

| | NDCG@10 | | | CTR@1 | | |
|---|---|---|---|---|---|---|
| | per | nav | inf | per | nav | inf |
| *CPS* | **106.64** | **104.88** | **97.83** | **62.22** | 62.05 | **57.78** |
| *Lin-$\epsilon$(disjoint)* | 6.40$^{\blacktriangledown}$ | 5.16$^{\blacktriangledown}$ | 2.74$^{\blacktriangledown}$ | 5.60$^{\blacktriangledown}$ | 4.14$^{\blacktriangledown}$ | 1.51$^{\blacktriangledown}$ |
| *Lin-$\epsilon$(hybrid)* | 71.11$^{\blacktriangledown}$ | 88.64$^{\blacktriangledown}$ | 33.58$^{\blacktriangledown}$ | 55.73$^{\triangledown}$ | **66.69** | 20.01$^{\blacktriangledown}$ |

**Table 2: Online performance, *focused* experiment.**

| | NDCG@10 | | | CTR@1 | | |
|---|---|---|---|---|---|---|
| | per | nav | inf | per | nav | inf |
| *CPS* | 70.71 | 61.45 | **47.34** | 91.01 | 85.85 | 63.74 |
| *Lin-$\epsilon$(disjoint)* | **120.22**$^{\blacktriangle}$ | 50.85 | 14.67$^{\blacktriangledown}$ | **165.14**$^{\blacktriangle}$ | **128.21**$^{\blacktriangle}$ | 26.12$^{\blacktriangledown}$ |
| *Lin-$\epsilon$(hybrid)* | 90.54$^{\blacktriangle}$ | **63.07** | 47.13 | 98.52 | 118.69$^{\blacktriangle}$ | **96.94**$^{\blacktriangle}$ |

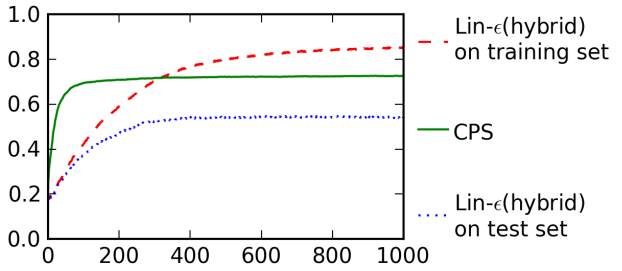


**Figure 1: Offline performance (nav), *focused* experiment.**

**Acknowledgements.** We would like to thank Jun Wang and Emine Yilmaz for supporting this work.

# References

[1] K. Hofmann, A. Schuth, S. Whiteson, and M. de Rijke. Reusing historical interaction data for faster online learning to rank for IR. In *WSDM '13*, pages 549–558, 2013.

[2] T. Joachims. Evaluating retrieval performance using clickthrough data. *Text Mining*, pages 79–96, 2003.

[3] J. Langford and T. Zhang. The epoch-greedy algorithm for multi-armed bandits with side information. In *Advances in neural information processing systems*, pages 817–824, 2008.

[4] L. Li, W. Chu, J. Langford, and R. E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *WWW '10*, pages 661–670, 2010.

[5] T. Qin, T.-Y. Liu, J. Xu, and H. Li. Letor: A benchmark collection for research on learning to rank for information retrieval. *Information Retrieval*, 13(4):346–374, 2010.

[6] F. Radlinski, M. Kurup, and T. Joachims. How does clickthrough data reflect retrieval quality? In *CIKM '08*, pages 43–52, 2008.

[7] A. Schuth, K. Hofmann, S. Whiteson, and M. de Rijke. Lerot: An online learning to rank framework. In *LivingLab '13*, pages 23–26, 2013.