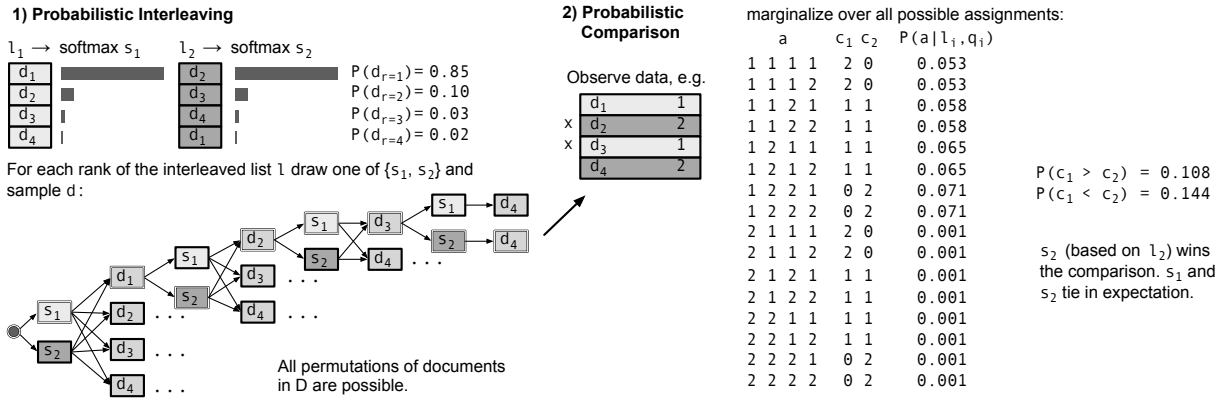

Figure 5: Example of an interleaved comparison using our *probabilistic marginalization* method with softmax functions.

rank:

$$P(L|A,Q) = \prod_{r=1}^{|L|} P(L[r] \mid A[r], L[1,r-1], Q), \qquad (6)$$

where $A[r] \in \{1,2\}$ specifies which softmax function contributed $L[r]$, the document at rank r, and $L[1,r-1]$ is the list of all documents between ranks 1 and $r-1$.

Figure 5 illustrates the resulting probabilistic method for comparing rankers based on implicit feedback. For an incoming query, the system generates an interleaved document list, as described in §4.1. After observing clicks, the system computes the probability of each possible outcome. All possible assignments are generated, and the probability of each is computed using Eq. 5. The example shows that, while the softmax functions tie in expectation (because our method is unbiased), s_2 has a higher probability of winning, because it has a higher probability of generating the clicked documents than s_1 (because the method is sensitive to such differences in rankings).

Generating all possible assignments and computing their probabilities for a given list is computationally expensive. However, assignments and probabilities need to be calculated only up to the lowest observed click. In the worst case, the lowest click is at the lowest rank of the interleaved list, which is a small constant, e.g., 10.[2] In practice, we perform computations using logs and sums instead of probabilities and products to avoid buffer underflow.

5. EXPERIMENTS

We present three sets of experiments, designed to answer the question: *To what extent do the analytical differences between the comparison methods translate into performance differences?*

In our first set of experiments, we address this question on a large scale. We compare a large number of ranker pairs to examine how the analytical differences between comparison methods affect the accuracy of ranker comparisons. In the second experiment, we use these first results to select a small subset of ranker pairs with which to study convergence characteristics in more detail. In the third, we use the same subset of rankers to examine the effect of noise in click data.

All experiments employ the same experimental setup that simulates user interactions. This setup is described in the next section (§5.1). After this, we specify our data (§5.2) and runs (§5.3).

[2]To further reduce computation, probabilities for documents below a certain rank can be set to a small constant, trading a small loss in sensitivity for gains in speed.

5.1 Experimental setup

Our experiments are based on a recently developed experimental setup that simulates user interactions based on datasets with explicit relevance judgments and user click models [12]. This simulator fulfills the assumption made by all comparison methods that more relevant documents are more likely to be clicked. Simulating user interactions makes it possible to study the comparison methods at different levels of noise in user clicks. In addition, the experiments can be repeated for many comparisons.

The simulation takes as input a dataset with queries and explicit relevance judgments and simulates user interactions as follows. It simulates a user submitting a query by randomly sampling from the set of available queries (with replacement).[3] On the system side, the interleaved comparison method observes the sampled query and produces an interleaved result list that is presented to the user (sent back to the simulator). User clicks are then simulated using a click model and the relevance judgments provided with the dataset.

The click model simulates user interactions according to the Dependent Click Model (DCM) [8, 9], an extension of the cascade model [6]. According to this model, users traverse result lists from top to bottom. For each document they encounter, they decide whether the document representation is promising enough to warrant a click (e.g., based on the title and document snippets). If, after clicking on a document, the users' information need is satisfied (likely if the document was relevant), then they stop browsing the result list. Otherwise, they continue examining the result list (likely if the document was not relevant).

We implement the click model following [12], with the difference that we define click and stop probabilities based on graded relevance assessments instead of binary ones. Simulated users examine the result list starting at the top. For each document they encounter, they decide whether to click or not, with probability $P(c|R(d))$, i.e., the probability of a click given the relevance level of the examined document $R(d)$. While the relevance level is not known to the user before clicking, we assume that its presentation in the result list is informative enough to make relevant documents more likely to be clicked. The probability that a user is satisfied with the obtained results and stops after clicking a document with relevance level $R(d)$ is modeled by $P(s|R(d))$.

We instantiate the click model in two ways (for 5 relevance lev-

[3]In the absence of information about the original distribution of the queries in the data set, we sample uniformly. An advantage of doing so is that frequent queries cannot dominate performance. Different query distributions may affect convergence speed, but not the qualitative differences between the comparison methods we evaluate.

Table 2: Instantiations of the click model used in our experiments.

perfect model					
$R(d)$	0	1	2	3	4
$P(c\|R(d))$	0.00	0.20	0.40	0.80	1.00
$P(s\|R(d))$	0.00	0.00	0.00	0.00	0.00
realistic model					
$R(d)$	0	1	2	3	4
$P(c\|R(d))$	0.05	0.10	0.20	0.40	0.80
$P(s\|R(d))$	0.00	0.20	0.40	0.60	0.80

els) to study the behavior of comparison methods under *perfect* and *realistic* conditions. The perfect click model is used to examine the behavior of comparison methods in an ideal setting. The stop probabilities are 0 for all relevance levels, i.e., the user continues to examine documents regardless of the relevance of previously encountered documents. Therefore, all documents in the top 10 of each result list are examined. The click probabilities are defined such that highly relevant documents ($R(d) = 4$) are always clicked, and that non-relevant documents ($R(d) = 0$) are never clicked. Click probabilities for relevance levels between these extremes decrease monotonically. The realistic click model was designed to approximate noisier click behavior. Here, highly relevant documents are most likely to be clicked ($P(c|R(d) = 4) = 0.8$), and click probabilities decrease for lower relevance levels. Similarly, stop probabilities for this model are highest after seeing a highly relevant document and decrease for lower probabilities. Besides the different levels of noise, the two models differ in the overall expected number of clicks, which is lower for the realistic click model because the click probabilities are consistently lower and the stop probabilities are higher. Table 2 lists the probabilities defined for the two models.

The click models instantiated here are appropriate for our evaluation of comparison methods, as they satisfy the assumptions made by all these methods. In preliminary experiments, we found that the level of noise introduced by the realistic click model is of the same order of magnitude as observed in previous work [21]. In addition, we observed qualitatively similar results for all other instantiations we tested. Finer details, such as click behavior that takes relations between documents into account are not captured. Therefore, extending our evaluation to include real-life experiments is important future work.

5.2 Data

All experiments make use of the MSLR-WEB30k Microsoft learning to rank (MSLR) data set.[4] This data set consists of approximately $30,000$ queries with 136 precomputed document features and relevance judgments. Relevance judgments are provided on a 5-point scale. Unless stated otherwise, our experiments use the training set of fold 1. This set contains $18,919$ queries, with an average of 9.6 judged documents per query.

To generate rankers to compare in our experiments, we make use of the features provided for the documents of the data set. Specifically, we take each feature to be a ranker, and formulate the task of the comparison method as identifying the ranking features that are expected to give the best performance. In our first set of experiments, we exhaustively compare all distinct ranker pairs that can be generated in this way. For the second and third sets, we select a small subset that appears most suitable for more detailed analysis.

[4]Downloaded from http://research.microsoft.com/en-us/projects/mslr/default.aspx.

The experiments are designed to evaluate the comparison methods' ability to identify the better of two rankers based on (simulated) user clicks. To set the ground truth for which ranker is better, we use NDCG based on the explicit relevance judgments provided with the data set, following previous work [21]. We use the complete set of documents provided with the data set to determine NDCG for the ground truth, i.e., no cut-off is used. Note that comparisons between rankers using implicit feedback depend not only on the true difference in NDCG between two rankers but also on the magnitude of the difference per query and the number of affected queries.

5.3 Runs

All our experiments evaluate the same four interleaved comparison methods. As baselines, we use three existing methods (cf., §2):

- **balanced interleave**, as described in [23].
- **team draft**, as introduced in [23]; as in the original, clicks on top results are ignored when the documents are placed in identical order by both rankers.
- **document constraint**, introduced in [10], as described in §2.

In addition, we evaluate the experimental condition:

- **marginalized probabilities**: our model using the probabilistic interleave process (cf. §4.1) and the probabilistic comparisons defined in Eq. 4 (cf. §4.3).

Below we describe the three sets of experiments designed to assess interleaved comparison methods in terms of accuracy, convergence behavior, and robustness to noise.

Accuracy. The first experiment assesses the accuracy of interleaved comparison methods when evaluated on a large set of ranker pairs. We construct a ranker for each of the 136 individual features provided with the MSLR data set. We then evaluate the interleaved comparison methods on all $9,180$ possible distinct pairs that can be obtained from these rankers. We evaluate the methods on this large set of ranker pairs using the perfect click model and a fixed subset of the data consisting of $1,000$ queries. We then compare the outcome predicted by the interleaved comparison methods to the true NDCG of the rankers, computed on the same set of queries. We report on *accuracy* after $1,000$ queries: the percentage of pairs for which a comparison method correctly identified the better ranker.

Convergence. Based on the findings of the first experiment, we select a small subset of ranker pairs with which to investigate the methods' behavior in more detail. These experiments were designed to follow the setup used in [21] as closely as possible, with the difference that we use simulated interactions with the perfect click model instead of real-life search engine traffic, as discussed in §5.1. For each pair of rankers, each comparison method was run on randomly sampled query sets of size 1 to $10,000$; queries for which an interleaved list did not obtain any clicks were ignored. Each experiment was repeated $1,000$ times. We evaluate each method by computing, for each sampled query set size, the fraction of these $1,000$ repetitions for which it detected a preference for the correct ranker. We compute binomial confidence intervals on this fraction using Wilson score intervals at the 95% level [3].

Noise. The third set of experiments assesses the robustness of interleaved comparison methods to noise in click feedback. The setup is identical to the previous one except that the *realistic* click model is used instead.