# Evaluating the Robustness of Click Models to Policy Distributional Shift

ROMAIN DEFFAYET, Naver Labs Europe, France and University of Amsterdam, The Netherlands
JEAN-MICHEL RENDERS, Naver Labs Europe, France
MAARTEN DE RIJKE, University of Amsterdam, The Netherlands

Many click models have been proposed to interpret logs of natural interactions with search engines and extract unbiased information for evaluation or learning. The experimental set-up used to evaluate them typically involves measuring two metrics, namely the test perplexity for click prediction and nDCG for relevance estimation. In both cases, the data used for training and testing is assumed to be collected using the same ranking policy. We question this assumption.

Important downstream tasks based on click models involve evaluating a different policy than the training policy, i.e., click models need to operate under *policy distributional shift*. We show that click models are sensitive to it. This can severely hinder their performance on the targeted task: conventional evaluation metrics cannot guarantee that a click model will perform equally well under distributional shift.

In order to more reliably predict click model performance under policy distributional shift, we propose a new evaluation protocol. It allows us to compare the relative robustness of six types of click models under various shifts, training configurations and downstream tasks. We obtain insights into the factors that worsen the sensitivity to policy distributional shift, and formulate guidelines to mitigate the risks of deploying policies based on click models.

CCS Concepts: • **Information systems** → **Query log analysis**.

Additional Key Words and Phrases: Click models, Offline evaluation, Web search, Distributional shift

**ACM Reference Format:**
Romain Deffayet, Jean-Michel Renders, and Maarten de Rijke. 2022. Evaluating the Robustness of Click Models to Policy Distributional Shift. *ACM Transactions on Information Systems* 1, 1 (October 2022), 28 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

## 1 INTRODUCTION

Search engines rank items according to their relevance to users, given the query they enter as well as the user and search context. To do so, many learning-to-rank (L2R) approaches leverage click logs, due to their abundance and the realistic settings they result from [7, 23]. However, clicks and skips are not direct signals of relevance. They emerge from interactions of users with the search system, meaning that the data is biased by the policy in place in the search system during data collection, often called the *logging policy* [18]. Different sources of intrinsic bias induced by the logging policy have been identified, such as position bias [22] (the position of documents in the search engine result page (SERP) influences their click likelihood) or trust bias [49] (users are likely to trust the system to return appropriate results and to click on top-returned documents regardless of their actual relevance).

Authors' addresses: Romain Deffayet, Naver Labs Europe, France and University of Amsterdam, The Netherlands, romain.deffayet@naverlabs.com; Jean-Michel Renders, Naver Labs Europe, France, jean-michel.renders@naverlabs.com; Maarten de Rijke, University of Amsterdam, The Netherlands, m.derijke@uva.nl.


1046-8188/2022/10-ART
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Therefore, in order to improve L2R models, it is crucial to gain a better understanding of user behavior on search engines by using *click models* [11]. They model user behavior by learning to generate realistic click sequences, while maintaining an interpretable structure that makes it possible to identify and disentangle the underlying causal factors of user behavior (relevance, examination, . . ., often in the form of latent variables of the model). This constrained structure allows one to extract a more accurate, unbiased relevance signal from the logs, and to derive models that do not replicate the biases seen in the logged data. The typical debiasing workflow with click models consists of (i) collecting biased click logs from the search engine, (ii) training a click model on this dataset, and (iii) using the resulting (hopefully) unbiased model to improve the policy of the search engine.

### 1.1 Evaluating click models

Click models are often evaluated on two tasks [15]: (i) *click prediction*, typically measured by the perplexity (PPL) [14] obtained when predicting clicks on a separate test set, and (ii) *relevance estimation*, typically measured by the normalized discounted cumulative gain (nDCG) [20] of rankings produced by the ordering of relevance scores recovered by the click model compared to those based on relevance grades given by human annotators. The click prediction metrics are usually computed on a test set collected using the same ranking policy as the training set, thus presenting the same intrinsic biases. Consequently, this type of evaluation is able to quantify the goodness of fit of a click model to the distribution of the logged data, but cannot guarantee that this data has been effectively debiased by the click model. On the contrary, at first glance one could hypothesise that strong performance in the relevance estimation task ensures effective debiasing. After all, a high nDCG score should be an indication that the causal identification of underlying factors (relevance, examination, etc.) has been successful, which would mean that the resulting click model is unbiased with respect to the logging policy.

The central question that motivates this paper is: *can the current evaluation practice for click models ensure that a click model is robust to changes in the ranking policy?* To make this question more precise, we introduce the notion of policy distributional shift.

### 1.2 Policy distributional shift and click models

*Policy distributional shift* (PDS) occurs when we ask the click model to predict clicks on rankings produced by a policy different from the logging policy, a task known as *off-policy evaluation* (OPE) [47]. PDS is a type of covariate shift, as it modifies the input distribution of the click model at test time. Under PDS, spurious correlations replicated from the data can have a detrimental impact on the performance. Therefore, robustness to PDS requires successful causal identification of the underlying factors, i.e., effective debiasing of the logged data.

Policy distributional shift matters. To illustrate its real-world impact, we list five common downstream tasks for click models and group them into three categories:

**Group ①: Latent variable extraction.**

- Label debiasing: extracting unbiased relevance scores from the model either to directly derive a policy or to serve as labels or features in a supervised L2R approach [6, 48];
- Counterfactual L2R: extracting propensity scores to reweight individual samples in a L2R loss [35, 48].

**Group ②: Off-policy evaluation.**

- Click-through rate (CTR) prediction: deriving CTR estimates of specific rankings for optimizing ad placement and pricing [9, 33].

**Group ③: Inverse model optimisation.**

- Fair ranking: extracting both relevance and exposure scores to measure and control the utility and fairness of fair re-ranking algorithms [44];
- Offline Bandits and RL: using the model as an interaction predictor for training bandits or RL policies [17, 51].

Tasks in Group ① do not require OPE, so they are not subject to underperformance due to PDS. However, we show in Section 4 that existing offline evaluation protocols cannot guarantee that the relevance or propensity scores have been effectively debiased, which defeats the purpose of click models for such applications.

CTR prediction (②) consists in performing explicit OPE; as a consequence, it is sensitive to policy distributional shift. We show in Section 6.1 that existing offline evaluation protocols are not able to detect such sensitivity, and we investigate how click models compare to each other in this setting in Section 6.2.

The problem of PDS is especially critical for tasks in Group ③, where we seek to recover a high-performance policy through an optimisation process. Such tasks require (implicitly or explicitly) numerous instances of OPE to select the best-performing policy. This leads to a phenomenon known as the *optimiser's curse* where inaccuracies of the model are exploited by the optimisation algorithm, potentially leading to failure at inference time [21, 45]. We compare click models w.r.t. the performance of the policies recovered in such tasks in Section 6.3.

### 1.3 Research goal and findings

Our research goal is to assess whether the current evaluation practice for click models is able to detect a lack of robustness to policy distributional shift, i.e., whether the current offline metrics are good indicators of the performance of click models on downstream tasks involving PDS (②, ③). To address this question, we proceed as follows. We assess the robustness of four different *types* of click models [3, 6, 12, 14], ranging from older models based on probabilistic graphs to newer neural models, which all encode a different structural assumption in their architecture. To make the comparison fair, we instantiate them under a unified implementation based on modern tools such as neural networks and stochastic gradient descent. In other words, we wish to compare the robustness of the key assumptions encoded in each model, rather than specific implementation details. Moreover, to enrich the experiments, we add two additional click models corresponding to two additional structural assumptions.

The experimental setup we adopt to address our research goal involves an evaluation protocol simulating OPE and online deployment that allows us to assess the robustness of click models under policy distributional shift in a way that is as close as possible to practical use cases of click models. Specifically, we simulate the deployment of click models for two tasks: CTR prediction ② and Offline Bandits ③, respectively, in Sections 6.2 and 6.3. For both experiments, we introduce a wide range of semi-synthetic environments in which relevance labels are derived from a real-world dataset but the simulated user behavior and the ranking policy are controlled by us. *Our experiments show that click models exhibit largely different and sometimes unexpected behaviors when tested outside of their training distribution, and that downstream policies are affected by this lack of robustness.*

The performance inside the simulator strongly depends on its design, and is thus not meant to be a reliable indicator of online performance, but it can provide insights into the inner working of click models and enable us to discriminate poorly robust ones. *It allows us to gain higher confidence that the ranking policies that we wish to derive, depending on the chosen downstream task, will behave as*

*expected once deployed into the real system*: it is a way to mitigate the risks of deploying L2R models online. Moreover, evaluating click models in a wide range of simulated environments decreases the sensitivity to simulator design and allows us to identify trends in the results.

### 1.4 Contributions

Our main contributions are the following:

- We identify the problem of policy distributional shift (PDS) in the context of click models and their evaluation;
- We show in a simulated environment that existing evaluation protocols do not guarantee good robustness of click models to PDS; and
- We propose an evaluation protocol for assessing the robustness to PDS of click models and compare various click models in a range of semi-synthetic environments.

Related work is discussed in Section 2. In Section 3, we introduce the concepts and tools necessary to our analysis, notably the existing offline evaluation protocol and the models we study throughout the paper. Section 4 provides a preliminary experiment on real-world datasets which indicates that the existing evaluation protocol suffers from important shortcomings. We therefore describe our augmented evaluation protocol and our experimental setup in Section 5. Finally, we instantiate this protocol in Section 6: we first provide counter-examples where the current evaluation protocol is unable to guarantee robustness to PDS (Section 6.1), and then perform a full comparative evaluation of click models in a wide range of environments for downstream tasks from Group ② in Section 6.2 and Group ③ in Section 6.3. An online appendix containing our code with reproduction instructions can be found at github.com/naver/dist_shift_click_models.

## 2 RELATED WORK

### 2.1 Off-Policy training and evaluation

Offline training and evaluation of search and recommendation systems has been extensively addressed in the literature [1, 27, 46], because performing online testing with real users is very costly. Among the issues to be addressed when evaluating search and recommendation systems offline, mitigating the bias induced by the policy used for data collection in learning-to-rank (L2R) has been tackled in numerous previous studies [27, 32, 35, 47], usually under the umbrella terms *Off-Policy Evaluation* or *Counterfactual L2R*.

In these approaches, the experimental setup is usually as follows: we aim to learn from click logs that have been generated by a logging policy. Since this logging policy is usually not uniformly random, certain documents are more likely to be clicked than they would be under a different policy. Therefore, a correction is applied to de-bias the observed clicks [24]. In domain-agnostic off-policy evaluation [32, 47], the correction corresponds to the probability of the document being placed at the rank at which we found it, by assuming a certain structure for the logging policy. These approaches are domain-agnostic in the sense that they do not leverage the specific properties of user behavior on search and recommendation sytems. Conversely, in inverse propensity scoring (IPS) [24, 35], one assumes a certain user click model and reweighs each observed document by its probability of being examined by the user under the logging policy. Vardasbi et al. [49] and Oosterhuis and de Rijke [36] also model trust bias in order to improve the relevance estimates found by the counterfactual estimator. While certain studies compute the off-policy corrections with online swapping interventions [24] or with eye-tracking experiments [23], in this study we focus on the fully observational training of user click models, directly on the logs, as used in [35, 49, 51].

*Regardless of the method used for propensity estimation, one must ensure the validity of the off-policy correction.* To do so, most of the existing click model and counterfactual L2R literature evaluates the

de-biasing capabilities of learned models by using relevance labels provided by human annotators (either on real-world or semi-synthetic datasets) [3, 12, 24, 35, 49]. However, as we explained in the introduction, this setting does not cover all possible use cases of click models and most importantly does not evaluate what would happen under policy distributional shift, i.e., in many practical scenarios (see task groups ② and ③ in the introduction). Swaminathan et al. [47] and McInerney et al. [32] evaluate the performance of their respective estimator on a simulated CTR prediction task with a few different policies, which ensures a certain degree of robustness to distributional shift for this particular task. However, ensuring that type of robustness is especially critical in applications such as fairness-constrained utility maximisation [44] or RL-based L2R [25, 51], in which the optimisation process will exploit any inaccuracies of the click model if it serves its objective. Besides, their counterfactual estimators are domain-agnostic and do not leverage the specific behavior of users on search and recommendation services. *To the best of our knowledge, no other work has evaluated how counterfactual L2R estimators perform when they are applied on policies different from the logging policy, i.e., under policy distributional shift.* Conversely, the offline reinforcement learning (RL) literature has extensively tackled policy distributional shift [19, 26, 38], including model robustness in model-based RL [2, 50], which can be seen as a similar topic to ours, but these methods are both structure and domain-agnostic because they do not leverage the specific properties of (1) policies returning rankings and (2) user behavior on search and recommendation systems.

Dai et al. [13] consider a form of intrinsic distributional shift which occurs when the model is asked to generate a consistent click sequence while having been trained using conditional click probabilities on ground truth clicks, but they did not consider the policy distributional shift induced by the change of policy for evaluation. In the work that is perhaps the closest in essence to ours, Huang et al. [17] introduce a protocol for evaluating policies produced by models trained inside a de-biased simulator, but they address the topic of single-item recommendation, which does not require click models because the sources of bias are different from biases occurring in L2R.

### 2.2 Click models

As mentioned in the previous section, we focus on the evaluation of counterfactual estimators fitted from observed logged data, i.e. click models. In early work on click models [6, 12, 14, 29], authors encode assumptions about user behavior into a probabilistic graphical model (PGM) framework in order to separate the influence of the result page's presentation from the influence of the document's intrinsic relevance.

For example, the examination hypothesis, introduced with the position-based model (PBM) [12], postulates that a document must be examined and perceived as relevant in order to be clicked. Therefore, in this model, one must identify relevance and examination parameters through a method of parameter estimation such as expectation-maximisation or stochastic gradient descent. The cascade model [12] additionally states that users browse the page in a top-down fashion, and that the click probability at a given rank depends on document relevance at lower ranks. Although some of these assumptions can easily be challenged in many modern search engines, they allow a certain level of generalisation in many practical settings [11].

More recently, neural click models have emerged [3, 4, 8, 13, 52] because they enable a better representation of the relevant variables (query, document, vertical type, etc) and recent optimisation methods are efficient for training large-scale click models from abundant data. But some of these models [8, 13] do not offer a straightforward solution for extracting intrinsic relevance scores, because they encode a notion of contextual relevance, i.e., dependent on rank, previous clicks and SERPs and vertical type.

The comparison of these different click models with respect to their robustness to policy distribu-

tional shift is the main focus of this paper. To allow for a fair comparison between them and filter out differences originating from representational and algorithmic design choices, we adopt the same way of representing entities such as query, context and document and the same type of optimisation algorithms, while keeping the structural assumptions of the original methods. In particular, the "older" methods [6, 12, 14] are instantiated with the representational and optimisation tools used by newer methods [3, 4, 8, 13], i.e., deep neural networks with embedding-based input encodings and gradient-based optimisation algorithms.

## 3 EXPERIMENTAL SETUP

This section describes the experimental setup we adopt for the remainder of the paper. Section 3.1 introduces the necessary notations and recalls how click models are traditionally evaluated, and in Section 3.2 we define the stack of click models we will use and compare throughout the experiments. We detail our evaluation protocol for assessing click models' robustness to policy distributional shift in Section 5, after a preliminary experiment highlighting the shortcomings of the existing evaluation protocol (Section 4).

### 3.1 Problem statement and existing protocol

We consider users engaging with a web search system displaying SERPs, i.e., fixed-size ranked lists of documents $\mathcal{S} = (d_1, \dots, d_R)$, as a response to a query $q$ that the user entered. The user can then click on zero, one or more documents, which is represented by the sequence of binary outcomes $(c_1, \dots, c_R)$. Therefore, logs in the dataset $\mathcal{D}$ contain SERP-wide interactions $\mathcal{I}_\mathcal{S} = (\tau_1, \dots, \tau_R)$ with $\tau_r = (q, d_r, r, c_r)$, where $1 \leqslant r \leqslant R$. For simplicity of notation, we will note $\mathbf{1}_{(q,d,r)}$ the function $\tau \mapsto 1$ if $\tau[0] = q, \tau[1] = d, \tau[2] = r$ and 0 otherwise for any $\tau \in \mathcal{D}$.

In order to better convey our argument, we do not consider session-based click models and we strip all datasets of additional context such as vertical type, device, etc.

We consider click models trained by maximum likelihood estimation on the task of predicting clicks based on query and presented SERP. The models' performance is evaluated on two tasks:

- *Click prediction on a separate test set* (randomly or chronologically split), which is measured by the conditional perplexity (PPL) at each rank and the average conditional perplexity:

$$\begin{aligned} \text{PPL@}r &= 2^{\frac{1}{|\mathcal{D}|}\sum_{I\in\mathcal{D}} c_r^I \log_2 \tilde{p}_r^I + (1-c_r^I)\log_2\left(1-\tilde{p}_r^I\right)} \\ \text{PPL} &= \frac{1}{R}\sum_{r=1}^{R} \text{PPL@}r, \end{aligned} \tag{1}$$

  where $\tilde{p}_r^I$ is the conditional click probability according to the model.
- *Estimation of the relevance* $\text{rel}(q, d)$ *against labels provided by human annotators*, which is measured by the normalized discounted cumulative gain (nDCG) at several truncation levels.

Throughout the paper, we perform a statistical significance analysis by training each model using 10 random seeds, where the seed controls model initialisation as well as the order of input documents to rank in the relevance estimation task. Confidence bounds use Student's $t$-distribution and statistical tests are Welch's $t$-tests. Both use a confidence level of 0.95.

### 3.2 Click model definitions

In this section, we define several click models that will be used in the remainder of the paper. To make the discussion easier in Section 4, we group these models into three families: (i) naive baselines based on number of clicks and impressions, (ii) as-is click models that we select from the existing literature, and (iii) modified click models, which are adapted from the literature to fit our

requirements, that we introduce to enrich the experiments. As we mentioned in the introduction, we are interested in assessing the robustness of different structural assumptions to policy distributional shift, rather than the effectiveness of specific implementation choices. To this end, we test several *types* of click models which differ by their key structural assumptions, and we implement all models in the same way, i.e., with modern neural methods, to ensure a fair and up-to-date comparison, as we describe in Section 3.2.2.

*3.2.1 Naive baselines.* For each document-query pair $(q, d)$, we define the number of clicks at rank $r$ as $N^c_{q,d,r} = \sum_{\tau \in \mathcal{D}} c^\tau \, \mathbf{1}_{(q^\tau=q, d^\tau=d, r^\tau=r)}$ and aggregated over all ranks as $N^c_{q,d} = \sum_r N^c_{q,d,r}$, where $\tau = (q^\tau, d^\tau, r^\tau, c^\tau)$, as defined in the previous section. Similarly, we define the number of impressions at rank $r$ as $N^i_{q,d,r} = \sum_{\tau \in \mathcal{D}} \mathbf{1}_{(q^\tau=q, d^\tau=d, r^\tau=r)}$ and aggregated over all ranks as $N^i_{q,d} = \sum_r N^i_{q,d,r}$. Using these quantities, we define the following models:

**dCTR:** document-based click through rate model [12]: It is one of the simplest click models, assuming that every document in the list is examined equally, independently of its rank. Consequently, its estimate of CTR is identical to its estimate of relevance probability:

$$P(C_d = 1|q) = \text{rel}(q, d) = \frac{N^c_{q,d}}{N^i_{q,d}}. \tag{2}$$

where $C_d$ is a random variable indicating whether the user has clicked document $d$.

**drCTR:** rank-weighted dCTR: It is an improved version of the dCTR model, relaxing its rank-independence assumption. It basically consists of a combination of two simple models: the dCTR model and the rCTR model [11], the latter making a document-independence assumption (i.e. the click probability only depends on the rank of the document).

$$\begin{aligned} P(C_{d_r} = 1|q) &= \frac{N^c_{q,d,r}}{N^i_{q,d,r}}, \\ \text{rel}(q, d) &= \sum_r \frac{1}{r\text{CTR}(r)} \times \frac{N^c_{q,d,r}}{N^i_{q,d,r}}, \end{aligned} \tag{3}$$

with $r\text{CTR}(r) = \left(\sum_{\tau \in \mathcal{D}} c_\tau \, \mathbf{1}_{(r_\tau=r)}\right) / \left(\sum_{\tau \in \mathcal{D}} \mathbf{1}_{(r_\tau=r)}\right)$.

**TopPop:** popularity-based model, based on the number of clicks:

$$\text{rel}(q, d) = N^c_{q,d}. \tag{4}$$

Of course, the relevance estimate of this model is strongly influenced by the logging policy, because the latter will determine the number of impressions of a document and, therefore, the number of opportunities to be clicked. The following two models further exacerbate this bias, by giving more importance to the number of impressions of a document. They are not intended to be used as realistic click models, but as a way of detecting and quantifying the potentially detrimental effect of such bias in our experiments.

**TopPopObs:** popularity-based model, based on the number of clicks and impressions:

$$\text{rel}(q, d) = N^c_{q,d} \times N^i_{q,d}. \tag{5}$$

**RankTopObs:** rank-weighted popularity-based model, based only on the number of impressions:

$$\text{rel}(q, d) = \sum_r r\text{CTR}(r) N^i_{q,d,r}. \tag{6}$$

Note that TopPop, TopPopObs and RankTopObs are not well-defined click models, as they cannot be used for click prediction. Additionally, for dCTR and drCTR, we add Bayesian smoothing with a

Dirichlet prior with parameter $\alpha = 1$ when predicting clicks. This allows us to avoid arbitrarily high perplexities on rarely seen items.

*3.2.2 As-is click models.* We select from the existing literature four widely used *types* of click models that make different modeling assumptions. Here, we first list and describe the key structural assumptions of the models we include in our experiments, and then explain in more detail the selection and standardisation process, with the help of two exclusion criteria.

We select the following four types of model from the literature :

**PBM:** position-based model [12]. It makes the examination hypothesis by stating that a user clicks document $d$ if and only if that document is attractive to the user and has been examined by them: $C_d = 1 \Leftrightarrow A_d = 1$ and $E_d = 1$. Moreover, it encodes this assumption such that attractiveness $\alpha_{q,d}$ only depends on the query-document pair and examination $\gamma_r$ on the rank:

$$\begin{aligned} P(C_d = 1) &= P(A_d = 1) \times P(E_d = 1) \\ P(A_d = 1) &= \alpha_{q,d} \\ P(E_d = 1) &= P(E_r = 1) = \gamma_r. \end{aligned} \tag{7}$$

It is further assumed that the document relevance coincides with its attractiveness:

$$\text{rel}(q, d) = \alpha_{q,d}. \tag{8}$$

**UBM:** user browsing model [14]. UBM makes the same assumptions except that the examination probabilities also depend on the rank of the latest clicked document:

$$\begin{aligned} P(C_d = 1) &= P(A_d = 1) \times P(E_d = 1) \\ P(A_d = 1) &= \alpha_{q,d} \\ P(E_d = 1) &= P(E_r = 1 \mid C_1 = c_1, \ldots, C_{r-1} = c_{r-1}) = \gamma_{r,r'}, \end{aligned} \tag{9}$$

where $\gamma_{r,r'}$ is the probability that the document at rank $r$ is examined given that the latest clicked document was located at rank $r'$.

**DBN:** dynamic Bayesian network model [6]. Comparatively to previous models, it adds the concept of satisfaction $S_d$ that can happen with probability $\sigma_{q,d}$ after a click; note that this satisfaction probability depends on the particular query-document pair. The model assumes that the user examines the page in a top-down fashion with discount factor $\gamma$ until they are satisfied:

$$\begin{aligned} P(C_d = 1) &= P(A_d = 1) \times P(E_d = 1) \\ P(A_d = 1) &= \alpha_{q,d} \\ P(E_1 = 1) &= 1 \\ P(E_r = 1 \mid E_{r-1} = 0) &= 0 \\ P(E_r = 1 \mid S_{r-1} = 1) &= 0 \\ P(E_r = 1 \mid E_{r-1} = 1, S_{r-1} = 0) &= \gamma \\ P(S_r = 1 \mid C_r = 1) &= \sigma_{q,d}. \end{aligned} \tag{10}$$

The relevance probability is then estimated by:

$$\text{rel}(q, d) = \alpha_{q,d} \times \sigma_{q,d}. \tag{11}$$

**NCM:** neural click model [3]. NCM does not encode the examination hypothesis but instead considers click prediction as a sequence-to-sequence problem, where the SERP is modeled as a top-down sequence:

$$P(C_{d,r_d} = 1) = f(q, d_1, \ldots, d_{r_d}, c_1, \ldots, c_{r_d-1}), \tag{12}$$

where $f$ is parameterised as a long short-term memory (LSTM) in the original implementation [3], with ad-hoc representations (embeddings) for queries, documents and clicks. Because NCM does not implement the examination hypothesis, it does not explicitly infers a latent variable which can be interpreted as the relevance probability. Instead, thanks to the top-down browsing assumption, Borisov et al. [3] suggest that a relevance score can be inferred by placing the considered document at the first position of the ranking:

$$\text{rel}(q, d) = f(q, d). \tag{13}$$

*Selection process.* The selection of click models from the literature was made according to two exclusion criteria :

- **EC1 : The click model should not leverage context other than position and previous clicks on the same SERP**, such as vertical type, timestamp, previous SERPs, etc. This allows us to compare models on a fair basis, and we leave the effect of context features on the robustness of click models for future work. This criterion prevents the study of context-related biases but still allows the mitigation of the main identified sources of bias such as position bias [22] which constitute the motivation for using click models.
- **EC2 : The click model should offer a way to compute intrinsic, de-contextualized relevance scores**, i.e., relevance scores depending solely on the query-document pair. This is required in order to compute nDCG, and to leverage relevance scores in downstream tasks such as label debiasing.

As a result of **EC1**, we exclude context-aware models such as [CSM, 4], [MCM, 52], [CACM, 8], [AICM, 13] and [GraphCM, 28]. However, we include in Section 3.2.3 a variant of [CACM, 8] stripped of additional context, called CACM$^{\ominus}$. When stripped of such context, AICM and MCM reduce respectively to [NCM, 3] and [DBN, 6], that we already include in our experiments, and GraphCM reduces to CACM$^{\ominus}$. It should also be noted that CTR prediction models [10, 16], which are widely used in recommendation settings, incorporate no concept of latent relevance and do not explicitly consider the ranks of the recommended items. Moreover, these models exploit the "collaborative" nature of the problem, namely that a single item is typically seen by numerous users and users interact with a lot of common items. In our stylised setting, the queries of the datasets are associated most of the time with disjoint sets of documents and the only available information consists of their id's, their ranks and the clicks. Consequently, there is no collaborative nature we could capture and rely on. In other words, these model reduce to the key assumptions of [dCTR, 12] (included in the previous section) when stripped of additional features.

To satisfy **EC2**, we also had to modify the relevance model of our stripped variant of CACM, as we will see in Section 3.2.3, because the relevance variable depends on rank and previous clicks in the original paper. For the same reason, we had to exclude CSM because even when stripped of the use of timestamp, its bi-GRU architecture does not allow the computation of de-contextualized relevance scores.

*Standardisation process.* In order to ensure a fair comparison, we implement the included models using the same representation of query and documents. More specifically, for models following the examination hypothesis (PBM, UBM, DBN), we separately encode query and documents into embeddings of size 64, which are then combined using a multi-layer perceptron (MLP). In particular, this means that, instead of considering query-document attractiveness ($\alpha_{q,d}$) and satisfaction probability ($\sigma_{q,d}$) directly as model parameters typically identified by an Expectation-Maximisation algorithm, these variables are modeled as the output of an MLP whose inputs are trainable document and query embeddings. For NCM, we directly pass the concatenation of the embeddings of query

and document and a binary variable representing the click value at previous rank, rather than the distributed representation introduced in the original paper. We also replace the original long short-term memory (LSTM) with a gated recurrent unit (GRU), as in our experiments it obtained the same performance more quickly. These implementation choices allow us to isolate the contribution of the structural assumptions encoded in these models and can be considered standard with respect to recent click model literature [4, 8, 13]. All models are trained by minimizing the cross-entropy between the predicted click distribution and the actual one, using stochastic gradient descent. Further training and implementation details of the click models can be found in Appendix A.

*3.2.3 Modified click models.* We adapt structural assumptions from the literature to our requirements, and therefore add two new types of click models encoding two different structural assumptions:

**CACM$^{\ominus}$:** Minimalistic, context-free variant of the context-aware click model [CACM, 8]. This click model relies on the examination hypothesis by having on one side attractiveness probabilities modeled with an MLP from query/document pairs, similarly to PBM/UBM/DBN, and on the other side examination probabilities computed similarly as in the original CACM paper [8]: the examination score is the output of a GRU whose input at each rank is the concatenation of a position embedding and an embedding for the previous click. There are two main differences with the original CACM :

(1) CACM$^{\ominus}$ is context-free, i.e., it does not leverage information from past interactions within the same session and the type of vertical. This is to ensure fair comparisons, folowing our exclusion criterion **EC1**.
(2) The attractiveness probabilities are agnostic to the rank of the document as well as previous clicks, which allows us to compute non-contextual relevance scores. This is in line with our exclusion criterion **EC2**.

Even though we drop certain specificities of the original work, our variant keeps the key structural assumptions of CACM. Also, note that CACM$^{\ominus}$ differs from NCM as it implements the examination hypothesis. We have:

$$\begin{aligned} P(C_d = 1) &= P(A_d = 1) \times P(E_d = 1) \\ P(A_d = 1) &= \alpha_{q,d} \\ P(E_d = 1) &= \mathrm{GRU}((c_1, p_1), \ldots, (c_{r_d-1}, p_{r_d-1})) \\ \mathrm{rel}(q, d) &= \alpha_{q,d}. \end{aligned} \tag{14}$$

where $c_k$ and $p_k$ designate embeddings for click and position respectively, while $\alpha_{q,d}$ is derived as the output of an MLP whose inputs are trainable document and query embeddings.

**ARM:** auto-regressive click model. Similarly to CACM$^{\ominus}$, ARM only differs from PBM, UBM and DBN by the way it models examination: the examination probability is the output of a logistic regression $f_\omega$ on previous clicks, with one parameter for each absolute rank. We can write:

$$\begin{aligned} P(C_d = 1) &= P(A_d = 1) \times P(E_d = 1) \\ P(A_d = 1) &= \alpha_{q,d} \\ P(E_d = 1) &= f_\omega(c_1, \ldots, c_{r_d-1}) \\ \mathrm{rel}(q, d) &= \alpha_{q,d}, \end{aligned} \tag{15}$$

with $f_\omega(c_1, \ldots, c_k) = \omega_0 + \sum_{j=1} \omega_j c_j$ for all $j \leqslant k$ and $\alpha_{q,d}$ derived as the output of an MLP whose inputs are trainable document and query embeddings. Even though ARM is similar to CACM$^{\ominus}$ in its architecture, the logistic regression requires significantly less parameters to predict the examination probability, and non-clicked documents do not influence the

examination. We also introduce a non-causal version of ARM (called ARM NC) where instead of passing only the previous clicks to the logistic regression, we pass all previous clicks as well as the current one, i.e., $P(E_d = 1) = f_\omega(c_1, \ldots, c_{r_d})$. This variant involves a degenerate training tasks: learning to predict a click which has been passed as input ! Yet, we include it in our experiments in order to highlight the shortcomings of the existing evaluation protocol.

Note once again that our contribution is not to introduce new click models or to improve existing ones, but to test the robustness of several structural assumptions in the context of policy distributional shift. To do so, we adopt a setting which can be considered standard with respect to the recent click model literature [3, 4, 8, 13], and we add two new click models (CACM$^{\ominus}$, ARM) in order to draw more reliable conclusions during our experiments.

## 4 NAÏVE BASELINES BEAT ADVANCED MODELS AT RELEVANCE ESTIMATION

In this section, we perform a preliminary experiment on real datasets using the widely used experimental setup described in Section 3.1, i.e., by computing two offline metrics: perplexity and nDCG. We first give experimental details in Section 4.1, and present the results in Section 4.2.

### 4.1 Data and evaluation protocol

We evaluate click models on two real-world datasets: the Yandex Relevance Prediction dataset [42] and the CLARA dataset, which comprises logs from Naver, a major South Korean search engine,[1] and relevance labels provided by human annotators. For both datasets, we follow the same processing workflow: (i) we first break down sessions into separate SERPs as we do not wish to use the additional information contained in a session beyond the current page; then (ii) we restrict the dataset to queries that have been annotated; finally (iii) we discard all pages without clicks. After pre-processing, the Yandex dataset contains 255,467 unique documents and 4,991 unique queries and the CLARA dataset contains 1,345,880 documents and 1,507 uniques queries. Both datasets have a cutoff rank of 10.

For perplexity computation, we use a chronological split where the test and validation set both represent 1/30th of the full Yandex dataset and 1/20th of the full CLARA dataset. For the nDCG computation, we remove documents which do not appear in the dataset as well as queries whose remaining documents all have equal relevance, as they would output a nDCG of 1 regardless of the quality of the click model.

We report our results in Tables 1 and 2 and 95% confidence bounds from the t-distribution can be found in Appendix D. Note that when we indicate statistical significance or absence of it, we do not correct for multiple comparisons because we are not looking for at least one test to be positive, but for baselines beating all existing click models on all nDCG metrics, i.e., for all tests to be positive at the same time. Indeed, the informal null hypothesis corresponding to *this experiment* could be defined as "*No naïve baseline beats all click models in terms of nDCG at every truncation level*". Therefore, adding comparisons to the experiment (more click models, more truncation levels) would only *increase* the likelihood of at least one test to be negative and decrease the likelihood of our hypothesis being rejected.

We also report in Appendix E the results measured by two other metrics, namely area under the ROC curve for click prediction and recall for relevance estimation.

[1] https://www.naver.com/

Table 1. Results on the CLARA dataset. ① are naive baselines, ② as-is click models, and ③ modified click models; see Section 3.2. The best performing model in average is reported in bold and $^{=}$ indicates a result is not significantly worse than the best performing model. Additionally, $^{\dagger}$ before a model's nDCG from ② or ③ indicates it is not significantly worse than ARM NC's. ↓ : lower is better; ↑ : higher is better. Full confidence bounds can be found in Appendix D.

| | | **Perplexity ↓** | | | | | **nDCG ↑** | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | **Click model** | Avg. | @1 | @2 | @5 | @10 | @1 | @3 | @5 | @10 |
| ① | RankTopObs | – | – | – | – | – | **0.7911** | **0.8360** | **0.8695** | **0.9167** |
| | TopPopObs | – | – | – | – | – | 0.7797 | 0.8263 | 0.8652 | 0.9137 |
| | TopPop | – | – | – | – | – | 0.7782 | 0.8285 | 0.8646 | 0.9136 |
| | dCTR | 1.1413 | 1.4000 | 1.2502 | 1.1333 | 1.0515 | 0.7445 | 0.7881 | 0.8323 | 0.8954 |
| | drCTR | 1.1381 | 1.3932 | 1.2413 | 1.1311 | 1.0500 | 0.7227 | 0.7762 | 0.8224 | 0.8872 |
| ② | PBM | 1.1445 | $^{=}$1.3955 | 1.2523 | 1.1368 | 1.0546 | 0.6801 | 0.7396 | 0.7993 | 0.8736 |
| | UBM | 1.1410 | **1.3906** | 1.2431 | 1.1343 | 1.0519 | 0.6825 | 0.7411 | 0.8002 | 0.8744 |
| | DBN | 1.1386 | $^{=}$1.3936 | $^{=}$1.2282 | 1.1283 | 1.0523 | 0.6665 | 0.7355 | 0.7967 | 0.8715 |
| | NCM | **1.1371** | 1.3994 | **1.2274** | **1.1254** | **1.0497** | 0.6554 | 0.7311 | 0.7942 | 0.8688 |
| ③ | CACM$^{\ominus}$ | 1.1417 | $^{=}$1.3962 | 1.2437 | 1.1337 | 1.0519 | 0.6808 | 0.7406 | 0.8001 | 0.8740 |
| | ARM | 1.1438 | 1.3994 | 1.2478 | 1.1350 | 1.0566 | $^{\dagger}$0.6861 | $^{\dagger}$0.7442 | $^{\dagger}$0.8027 | $^{\dagger}$0.8760 |
| | ARM NC | – | – | – | – | – | 0.6930 | 0.7456 | 0.8035 | 0.8768 |

Table 2. Results on the Yandex dataset. Same conventions as in Table 1.

| | | **Perplexity ↓** | | | | | **nDCG ↑** | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | **Click model** | Avg. | @1 | @2 | @5 | @10 | @1 | @3 | @5 | @10 |
| ① | RankTopObs | – | – | – | – | – | 0.7138 | 0.7003 | 0.7291 | 0.8034 |
| | TopPopObs | – | – | – | – | – | 0.7206 | 0.7023 | 0.7319 | 0.8056 |
| | TopPop | – | – | – | – | – | 0.7225 | 0.7110 | 0.7374 | 0.8097 |
| | dCTR | 1.3212 | 1.6054 | 1.5581 | 1.2901 | 1.1790 | **0.7246** | **0.7165** | **0.7456** | **0.8155** |
| | drCTR | 1.3146 | **1.5784** | 1.5439 | 1.2895 | 1.1764 | 0.6273 | 0.6531 | 0.6989 | 0.7795 |
| ② | PBM | 1.3177 | 1.5970 | 1.5488 | 1.2884 | 1.1781 | 0.5977 | $^{\dagger}$0.6243 | $^{\dagger}$0.6734 | $^{\dagger}$0.7621 |
| | UBM | 1.2823 | 1.5918 | 1.5366 | 1.2463 | 1.1247 | $^{\dagger}$0.6017 | $^{\dagger}$0.6255 | $^{\dagger}$0.6734 | $^{\dagger}$0.7631 |
| | DBN | 1.2803 | 1.5861 | 1.5200 | 1.2463 | 1.1271 | 0.5785 | 0.6096 | 0.6610 | 0.7529 |
| | NCM | **1.2717** | 1.5842 | **1.5141** | **1.2361** | **1.1152** | 0.5606 | 0.5987 | 0.6518 | 0.7462 |
| ③ | CACM$^{\ominus}$ | 1.2789 | 1.5969 | 1.5443 | 1.2405 | 1.1165 | 0.5648 | 0.6008 | 0.6546 | 0.7485 |
| | ARM | 1.3147 | 1.6142 | 1.5755 | 1.2719 | 1.1807 | 0.5993 | $^{\dagger}$0.6258 | $^{\dagger}$0.6730 | $^{\dagger}$0.7616 |
| | ARM NC | – | – | – | – | – | 0.6086 | 0.6279 | 0.6756 | 0.7634 |

### 4.2 Results

In Tables 1 and 2, we report the performance of the click models we described in Section 3.2. NCM achieves the lowest perplexity at all ranks but the first one on both datasets, while dCTR and RankTopObs achieve the highest nDCG on respectively Yandex and CLARA. Overall, models in Group ①, i.e., naïve baselines, beat well-formed models from Groups ② and ③ on the relevance estimation task.

As mentioned in the introduction, even though measuring perplexity on a test set collected by

the same policy as the training set is not able to guarantee robustness to policy distributional shift, one could hypothesise that the evaluation protocol of the relevance estimation task (using nDCG) is a good proxy for ensuring effective debiasing and therefore robustness to distributional shift.

However, the high nDCG of Group ① in both tables seems to contradict this hypothesis: *naïve baselines beat all click models in terms of nDCG, while we expect most of these baselines to be strongly biased by the logging policy* (especially TopPop, TopPopObs, Weighted TopObs and dCTR). Indeed, they incorporate no mechanism for correcting common biases of the logged data such as position bias or trust bias. Perhaps an even more surprising result is that the non-causal ARM (ARM NC), despite using a degenerate training task, is able to beat most or all existing click models in terms of nDCG on both datasets.

To explain these unsettling results, we hypothesise that when the logging policy is good, the nDCG-based evaluation protocol cannot distinguish between biased models and well-debiasing models, and thus cannot ensure robustness to policy distributional shift. To illustrate this intuition, let us assume the existence of perfect relevance annotations and an optimal logging policy with respect to these annotations. A click model such as dCTR incorporating no bias mitigation mechanism will overestimate the relevance of the most exposed documents and underestimate the relevance of the least exposed documents. But since the logging policy is optimal, the most relevant documents are also the most exposed, so the ordering of relevance scores learned by the biased click model does not differ from the optimal ordering, leading to an nDCG of 1. Yet, this dCTR model is strongly biased and would not be able to accurately predict the CTR of a different policy: for example, if we consider the reverse policy ranking documents by *increasing* order of relevance, dCTR would give the same CTR estimate as for the optimal policy, while this reverse policy would clearly lead to a lower CTR due to position bias. This counter-example shows that *achieving high nDCG is no guarantee for effective debiasing, and consequently nor for out-of-distribution robustness.* As an aside, biased models can be favored even more if the logging policy uses features that are meaningful for relevance prediction but not observable by the click model, as it is often the case in industrial settings.

Considering, again, the downstream tasks listed in the introduction, this lack of guarantee on debiasing performance and robustness to a change of policy is clearly a critical issue for the required Off-Policy Evaluation in task Groups ② and ③. But it is also problematic in tasks in Group ① because we may obtain narrow, conservative, strongly biased policies as a result of the training process, while we expect the use of click models to provide debiased and potentially diverse policies. As we hinted in the discussion above, nDCG is not a reliable indicator of click model debiasing when the logging policy itself outputs high-nDCG rankings, because one cannot distinguish high nDCG from having successfully debiased the click data and high nDCG from having replicated biases in the click data. However, one might expect that, in practical use cases, the rankings extracted from the click model should be at least as good as those of the logging policy, and that if we observe an improvement in nDCG over the logging policy, it could only be attributed to effective debiasing. On the contrary, we argue that:

(1) in many industrial settings, the logging policy can be expected to obtain higher nDCG than the click models we may train from it. Indeed, commercial search engines usually perform well because they use many additional features, while click models may not attain such performance alone. Instead, the relevance scores extracted from click models may be used as one feature of a larger learning-to-rank model and therefore be useful even without a direct improvement in nDCG over the logging policy;
(2) even if there is an improvement in nDCG over the logging policy, we cannot quantify how much of it can be attributed to effective debiasing. Indeed, nDCG being aggregated over all queries, it is possible that click models are affected by the issue we highlight above on certain

queries (e.g., head or tail), even though their aggregated nDCG is higher than the logging policy's. This renders improvements in nDCG unreliable.

### 4.3 Upshot

We have shown that models that we expect to be strongly biased (i.e., naive baselines implementing no bias correction) achieve high nDCG scores. It suggests that the current evaluation protocol based on relevance labels from human annotators is not able to single out biased models from well de-biased models. This would be a critical issue, as click models require correctly de-biasing the observed logs in order to perform well on downstream tasks involving policy distributional shift. We therefore formulate the hypothesis that nDCG in the current offline evaluation protocol is not a good indicator of robustness to distributional shift (hypothesis $\mathcal{H}$).

## 5 AN AUGMENTED EVALUATION PROTOCOL

The surprising results of the previous section motivate us to design an augmented evaluation protocol in a simulated environment, in order to verify hypothesis $\mathcal{H}$, to highlight the shortcomings of the nDCG-based evaluation protocol, and to allow researchers and practitioners to mitigate the risks induced by distributional shift.

### 5.1 New evaluation criteria

*5.1.1 Robustness of click prediction.* To evaluate the robustness to policy distributional shift of the click prediction capabilities of click models, we can measure the perplexity of the model on a dataset generated using a different ranking policy, i.e., a different distribution of rankings. We call this metric the *out-of-distribution* (ood) *perplexity*, as opposed to the usual *in-distribution* (ind) *perplexity*. If the perplexity of a model significantly increases on a new policy, we can conclude that causal identification during training partly failed, leading to high sensitivity to policy distributional shift. This protocol thus evaluates the downstream performance of click models on the off-policy evaluation (OPE) task (② in the introduction). Actually, the absolute value of perplexity is affected by other factors than click model's performance: it also depends on the dataset's click distribution, which itself depends on the choice of ranking policy, meaning we cannot directly compare ind-perplexity and ood-perplexity. Therefore, in Section 6.2, we only look at a *normalised perplexity* bounded by the respective performance of the best and the worst click model:

$$n\mathrm{PPL}(\mathrm{CM}_i) = 0.2 + \log\left(1 + \frac{\mathrm{PPL}(\mathrm{CM}_i) - \min_k \mathrm{PPL}(\mathrm{CM}_k)}{\max_k \mathrm{PPL}(\mathrm{CM}_k) - \min_k \mathrm{PPL}(\mathrm{CM}_k)}\right), \tag{16}$$

where $\mathrm{PPL}(\mathrm{CM}_i)$ is the perplexity obtained by the $i$-th click model. The use of the logarithm and the additive constant in this formula is simply for ease of visualisation in Figure 1, in order to spot small absolute differences. Note that using normalised perplexity means that all models in the experiment are compared relatively to each other. $n$PPL is bounded by 0.2 for the best model out-of-distribution and $0.2 + \log(2)$ for the worst. More importantly, *a click model $CM_i$ will be considered more robust than its counterparts if its ood normalised perplexity is lower than its ind normalised perplexity, i.e.,* $n\mathrm{PPL}^{ood}(\mathrm{CM}_i) < n\mathrm{PPL}^{ind}(\mathrm{CM}_i)$.

*5.1.2 Robustness of subsequent policies.* Click models are used to derive an unbiased ranking policy in four of the five tasks we identified in the introduction. In label debiasing for L2R, the policy is obtained by directly ordering documents by decreasing relevance, by a distillation process where an L2R algorithm uses relevance scores as training targets, or by using the unbiased labels as features of an L2R model. In counterfactual L2R, the propensities are extracted from the model, in order to reweight the training targets of an L2R algorithm. In the fair ranking task, relevance

and exposure scores are derived to find a policy maximizing a notion of utility while satisfying fairness constraints. Finally, in offline bandits and reinforcement learning, the click model is used as a click predictor for training agents seeking to maximise the expected reward, typically the expected number of clicks. It is therefore crucial to assess the quality of the policies that we aim to derive, depending on the choice of downstream task. To do so, we study the expected number of clicks of two downstream policies for each click model:

- The *Top-Down policy*, which consists in ranking documents by decreasing relevance scores according to the probability ranking principle [40]. This is the policy that we aim to recover in the label debiasing task.
- The *Max-Reward policy*, i.e., the policy maximizing the expected number of clicks according to the trained click model. It is the policy we wish to recover in offline bandits. As mentioned in the introduction, this downstream task requires numerous implicit or explicit instances of OPE, which is already evaluated by the first criterion, but it is also affected by the optimiser's curse [45]: the maximisation process is likely to select rankings that are grossly overestimated by the click model.

Note that in a non-Top-Down environment, i.e., when exposure does not always decrease with the rank, the Max-Reward policy has the potential to lead to more clicks than the Top-Down policy. A click model whose Max-Reward policy incurs a lower CTR than its Top-Down policy would therefore be interepreted as a poorly robust model.

For certain click models (UBM, NCM, ARM and CACM$^{\ominus}$), finding the Max-Reward policy by brute force can become intractable, especially if the cut-off rank of the desired policy is large or if SERP-specific context such as vertical type or GUI presentation is added. In our experiments, we randomly sample 8! rankings from the 10! possible rankings and find the best one according to the click model by brute force. We chose this sampling-based method over guided methods such as Beam Search because it is not biased towards certain types of solution; it is notably well-known that Beam Search favors near-Top-Down solutions [30].

### 5.2 Simulator design

The evaluation protocol involving the two criteria presented in the previous section is operationalised in a simulator. Although no simulation can guarantee good online performance, it allows us to mitigate the risks of deploying the model by testing the robustness of click models in a wide range of settings before deployment.

A suitable simulator needs to include the following components:

- An *internal click model*, which emulates the click behavior of users when confronted to a SERP;
- *Ranking policies*, which present SERPs to the simulated users, in response to a query; and
- *Relevance ground truth*, in order to compute the click probabilities as well as the nDCG for the relevance estimation task.

*5.2.1 Relevance ground truth.* For the relevance ground truth, we use real-world data from the Microsoft Learning to Rank Datasets [39], allowing us to get relevance labels on a scale of 0 to 4. In practice, we restrict the dataset to 1000 random queries which all have at least 10 documents of not-all-equal relevance.

*5.2.2 Ranking policies.* From the dataset, we are also able to get two ranking policies: BM25 [41], which we directly extract from the features, and a LambdaMART [5] policy we train from all available features. We then rescale the scores given by these policies to be between 0 and 1. We also derive a near-optimal policy ($\epsilon$-oracle) by adding Gaussian perturbations of variance 0.15 to the

rescaled ground truth relevance labels $(2^{\text{rel}_{q,d}} - 1)/15$. Finally, we derive a stochastic variant of all policies by sampling from a Plackett-Luce model [31, 37], in order to allow causal identification by the click models when the policy is used for training. In practice, the sampling is performed using the Gumbel sampling trick [34] with a temperature specific to each policy: $T = 0.1$ for $\epsilon$-oracle and lambdamart and $T = 0.03$ for BM25. The resulting policies are therefore stochastic but rather low-entropy.

*5.2.3 Query distribution.* In order to mimic a realistic query frequency distribution, we fit a power-law model on the query distribution of the CLARA dataset. We find that the $k$-th most frequent query appears with probability $p \propto (\alpha - 1)k^{-\alpha}$ with $\alpha = 1.12$.

*5.2.4 Internal click model.* Based on the relevance labels $\text{rel}_{q,d}$, we design three internal click models:

- DBN [6] with the attractiveness, satisfaction and continuation parameters taken respectively as $\alpha_{q,d} = 0.95 \times (2^{\text{rel}_{q,d}} - 1)/15$, $\sigma_{q,d} = 0.9 \times (2^{\text{rel}_{q,d}} - 1)/15$ and $\gamma = 0.9$.
- A "Complex Click Model" (CoCM), which is a mixture of click models that does not follow either the examination hypothesis or the cascade hypothesis. Therefore, all click models that we evaluate in our experiments suffer from click model mismatch, i.e., their structure cannot accurately model CoCM's distribution. The complete definition of CoCM can be found in Appendix B. Note that the policy placing the most relevant documents at the top is not necessarily optimal with this model as the examination is not top-down.
- CoCM mismatch: A variant of CoCM with a stronger click model mismatch (see Appendix B).

## 6 EVALUATING ROBUSTNESS TO POLICY DISTRIBUTIONAL SHIFT IN A SIMULATOR

The experiment described in Section 6.1 below provides counter-examples that confirm hypothesis $\mathcal{H}$ formulated in Section 4 and justify our evaluation protocol. Then, we instantiate this protocol and perform a comparison of six click models in Sections 6.2 and 6.3, corresponding respectively to the simulated deployments of click models for the tasks of CTR prediction (②) and Offline Bandits (③) .

### 6.1 Observable metrics do not guarantee robustness

In this section, we provide counter-examples where the ood-perplexity cannot be inferred from either ind-perplexity or nDCG. In Tables 3 and 4, we study the effect of policy distributional shift on several click models, under respectively DBN and CoCM as internal click models. For this particular experiment, we design ranking policies so as to create a strong policy distributional shift. To do so, we first sample 10 documents per query using relevance-stratified sampling. Then the training rankings are obtained by sampling in a top-down fashion from a Plackett-Luce model derived from the true relevances of these documents. The ind-perplexity is computed on a randomly-split separate test set. The policy used for ood-perplexity computation consists in ranking the same 10 documents by *increasing* order of relevance scores. Consequently, the training policy contains spurious correlations (e.g., position bias) that do not hold under the ood-testing policy, and it is near-optimal, which may lead to the behavior observed in Section 4 according to our hypothesis.

dCTR and ARM NC turn out exhibiting very poor ood-PPL, especially with a near-optimal logging policy (in Table 3), despite achieving high nDCG. This collapse under the test policy shows that they were unable to learn meaningful relationships, and are instead biased by the logging policy. More importantly, PBM, UBM, DBN, NCM and CACM$^{\ominus}$, which were designed to be unbiased with respect to the logging policy, show varying levels of robustness, and it does not seem possible to

Table 3. Effect of distributional shift with DBN as internal click model (same conventions as in Table 1). DBN Oracle is a DBN model with hardcoded optimal parameters, and therefore shows a lower bound of ind and ood perplexity.

| **Click model** | ind PPL ↓ | nDCG@3 ↑ | ood PPL ↓ |
|---|---|---|---|
| DBN Oracle | 1.2856 (+- 0.0000) | 1.0 (+- 0.0000) | 1.3002 (+- 0.0000) |
| DBN | 1.3230 (+- 0.0005) | 0.7784 (+- 0.0136) | 1.3355 (+- 0.0016) |
| dCTR | 1.3428 (+- 0.0000) | 0.9219 (+- 0.0015) | 1.4683 (+- 0.0000) |
| PBM | 1.3336 (+- 0.0005) | 0.8482 (+- 0.0033) | 1.3561 (+- 0.0023) |
| UBM | 1.3271 (+- 0.0005) | 0.8580 (+- 0.0000) | **1.3413 (+- 0.0011)** |
| NCM | **1.3248 (+- 0.0002)** | 0.7851 (+- 0.0124) | 1.3501 (+- 0.0027) |
| CACM$^{\ominus}$ | 1.3270 (+- 0.0005) | 0.8119 (+- 0.0116) | $^{=}$1.3414 (+- 0.0010) |
| ARM NC[2] | (1.2429 (+- 0.0003)) | **0.9315 (+- 0.0021)** | 1.6100 (+- 0.0074) |

Table 4. Effect of distributional shift with CoCM as internal click model (same conventions as in Table 1). CoCM Oracle is a CoCM model with hardcoded optimal parameters, and therefore shows a lower bound of ind and ood perplexity.

| **Click model** | ind PPL ↓ | nDCG@3 ↑ | ood PPL ↓ |
|---|---|---|---|
| CoCM Oracle | 1.2670 (+- 0.0000) | 1.0 (+- 0.0000) | 1.2611 (+- 0.0000) |
| dCTR | 1.3025 (+- 0.0000) | 0.8476 (+- 0.0018) | 1.3485 (+- 0.0000) |
| PBM | 1.3036 (+- 0.0003) | 0.7475 (+- 0.0135) | 1.3030 (+- 0.0011) |
| UBM | 1.2945 (+- 0.0004) | 0.7677 (+- 0.0137) | 1.2949 (+- 0.0007) |
| DBN | 1.2973 (+- 0.0003) | 0.6674 (+- 0.0111) | 1.3040 (+- 0.0006) |
| NCM | 1.2948 (+- 0.0003) | 0.6723 (+- 0.0105) | 1.3067 (+- 0.0009) |
| CACM$^{\ominus}$ | **1.2928 (+- 0.0002)** | 0.7019 (+- 0.0151) | **1.2934 (+- 0.0011)** |
| ARM NC[2] | (1.1891 (+- 0.0003)) | **0.8765 (+- 0.0029)** | 1.7012 (+- 0.0108) |

accurately predict their ood-perplexity from ind-PPL and nDCG. *Consequently, we cannot rely on results in either ind-PPL or nDCG to make statements about the success of click models at debiasing the logged data and their robustness to policy distributional shift.*

Our hypothesis regarding why dCTR and ARM NC can achieve such a high nDCG while showing poor performance out-of-distribution is that nDCG indistinguishably rewards biased and well-debiasing model (see Section 4). dCTR, incorporating no bias mitigation mechanism, and amplifies the biases present in the logged data. Regarding ARM NC, it seems surprising that a model using such a degenerate training task is even able to get a high nDCG. But it did not collapse to a trivial solution during training because it does not have a parameter specifically assigned to the current click (parameters are associated to absolute ranks). The functional structure of the examination branch makes it impossible to correctly estimate the examination probabilities and this incentivises the relevance branch to directly predict clicks, in a way reminiscent of what the dCTR model is doing. This behaviour consequently leads to strongly biased relevance scores.

It is also worth noting that the ood-PPL of oracle models can be higher or lower than their ind-PPL, even though these models perfectly match the internal click model. This is not surprising because, as explained in Section 5, the policy used to generate the data determines the expected click

[2]Note that the perplexity of ARM NC cannot be compared to other click models as it takes the click label as input, but we include it in Tables 3 and 4 to show that despite having access to the ground truth at test time, the correlations learned by this model during training were so spurious that its ood-perplexity is extremely high.

probabilities at each rank, and, therefore, how hard the click prediction task is. As an illustrating example, under DBN, the training policy is near optimal: top documents are very likely to be clicked and bottom documents are almost never clicked, leading to an easy task. But under the test policy, all ranks have either low relevance or low examination probabilities, making the prediction task harder. This is the effect we want to alleviate in our protocol for evaluating the robustness of click prediction by considering normalised instead of absolute perplexity.

*6.1.1 Upshot.* This experiment showed that the observable offline metrics commonly used in click model evaluation cannot guarantee robustness to distributional shift. Our evaluation protocol described in Section 5 allows us to observe the effect of distributional shift on simulated deployments for common downstream tasks.

### 6.2 Robustness of click prediction

This section serves three purposes: (i) we instantiate our evaluation protocol for mitigating the risk of a click model being affected by policy distributional shift so that it can be easily reproduced, (ii) we compare the robustness of several click models on the click prediction task, and (iii) we highlight how policy distributional shift can affect click models differently depending on training and test configurations.

Figure 1 illustrates the robustness of each click model compared to the other models in the experiment. In this graph, the perplexity is normalised according to our protocol (see Section 5). Therefore, when the blue line (ood-perplexity) is inside the red dashed line (ind-perplexity), this means that the model being considered is comparatively more robust than the others, and vice-versa.

PBM usually comes with poor ind-perplexity, but its robustness is far greater than most other models under a near-optimal policy like PL-oracle and strong distributional shift (when tested on BM25 or a random policy), making it a competitive choice for ood-click prediction in this case, despite its poor ind-performance. Note that this setting is common in practice as logging policies from commercial search engines usually have high performance. UBM is also quite robust overall, especially under a near-optimal policy and strong distributional shift. However, when trained on sub-optimal policies and mild click model mismatch, it does not match the robustness of some other models. Unsurprisingly, DBN almost always has the best performance both in-distribution and out-of-distribution when a DBN is also used as internal click model. But under click model mismatch, its robustness is very poor as ind-perplexity is a very unreliable indicator of ood-perplexity and it is constantly worse than other models when trained on a PL-oracle policy. NCM is also particularly brittle to distributional shift under better policies. However, its relatively better representativeness makes it a competitive choice under strong click mismatch. ARM is comparatively more robust than DBN or NCM under PL-oracle, but its ood-performance is quite unreliable and its ind-performance is usually too poor to make it competitive. Finally, CACM$^{\ominus}$ usually retains most of its very good ind-performance when tested out-of-distribution, which also makes it a good candidate for the CTR prediction task.

*6.2.1 Upshot.* In summary, each click model shows strengths and weaknesses in different settings, but PBM, UBM, and CACM$^{\ominus}$ are generally more robust than ARM, DBN and NCM, especially under strong policy distributional shift and a near-optimal logging policy, which are common real-world conditions. We therefore consider the former three as safer choices for the CTR prediction task ②.

### 6.3 Robustness of subsequent policies

Here, we instantiate the second criterion of our proposed evaluation protocol: measuring the CTR of policies produced by the click model. As explained in Section 5, we derive two policies for each click model and simulator configuration: Top-Down and Max-Reward. Under CoCM and

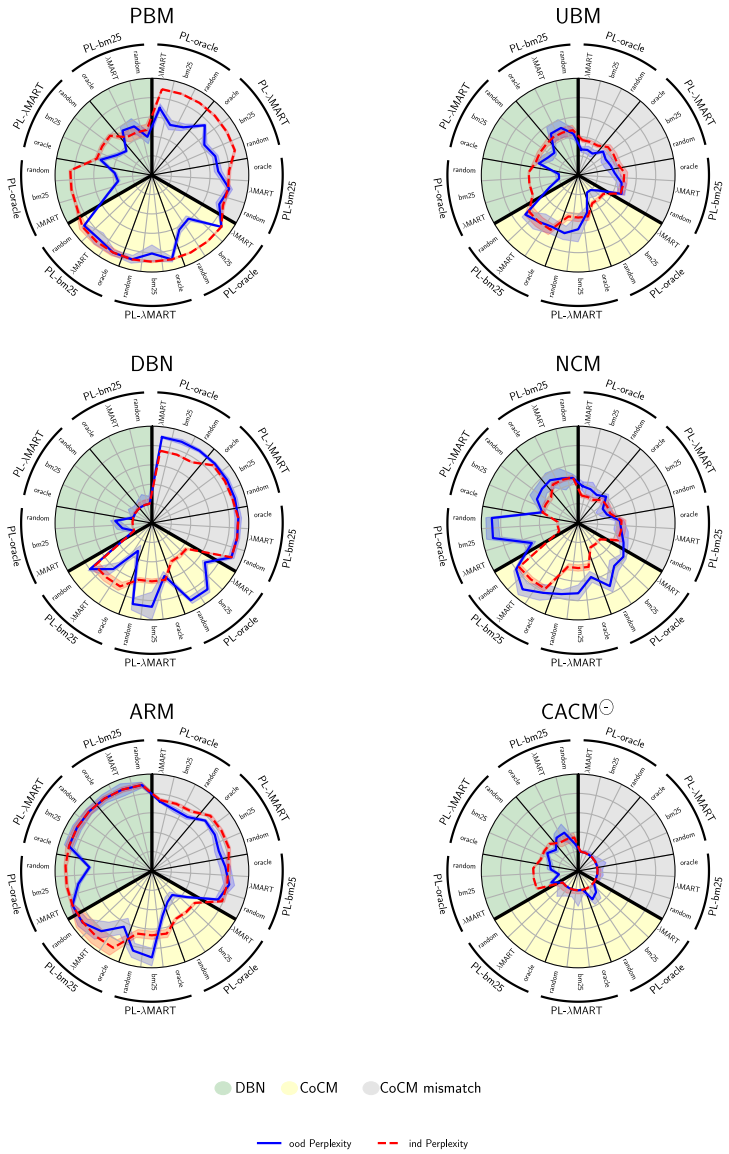


Fig. 1. Spider charts showing the level of robustness in a wide range of simulated environments. Perplexities represented on each graph are log-normalised perplexities where the best model is close to the center and the worst model is close to the border (see Section 5 for the complete formula of log-normalised perplexity). The background colors represent the type of internal click model, the outer circle of labels indicates the training policy and the inner circle indicates the test policy for ood-perplexity computation. Confidence bounds appear in shaded areas.

CoCM mismatch, the user does not always examine the page in a top-down fashion. Therefore, Max-Reward has the potential to lead to higher click-through rate than Top-Down. Conversely, spurious correlations in the data and high uncertainty on rarely seen SERP configurations may lead to high expected CTR according to the model but poor performance when facing the true internal

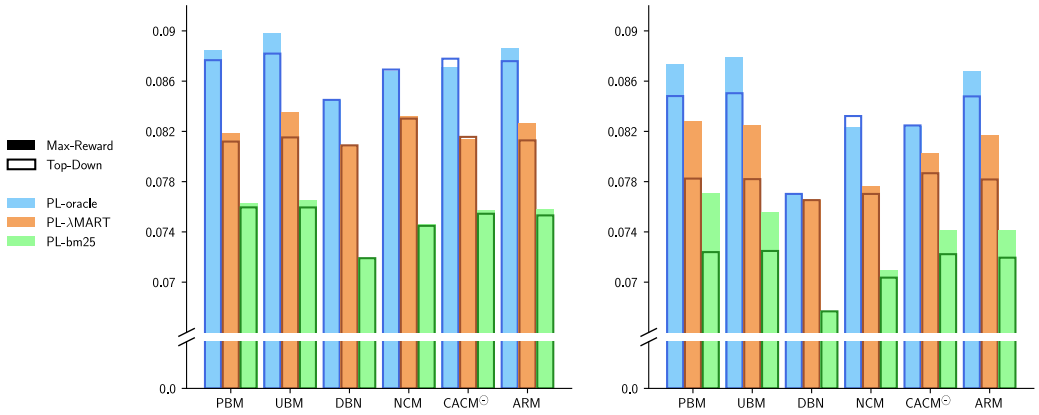


Fig. 2. Click-through rate obtained by the Top-Down and Max-Reward policies of click models. A figure including confidence bounds is provided in Appendix C. Left: CoCM as internal click model. Right: CoCM mismatch as internal click model.

click model.

In Figure 2, we report the observed click-through rate of the Top-Down and the Max-Reward policy extracted from six click models, using CoCM and CoCM mismatch as internal models. Confirming our intuition, the Max-Reward policy can perform better than the Top-Down policy, especially with CoCM mismatch as internal model (right plot). However, certain models sometimes achieve worse CTR than with Top-Down, demonstrating a critical lack of robustness to policy distributional shift.

By looking more closely at the relative performance of Top-Down and Max-Reward in Figure 2, we observe that complex and expressive models like NCM and CACM$^{\ominus}$ are poorly robust since they achieve no-better or worse performance with Max-Reward than with Top-Down in certain configurations. On the contrary, simpler models with fewer parameters, like PBM and UBM, can lead to highly rewarding policies and robustly increase their performance by applying Max-Reward instead of Top-Down.

We hypothesise that by relying on simpler structural assumptions and having fewer but more frequently used parameters to be trained, these models are less likely to suffer from spurious correlations as well as high uncertainty. It is striking to see that models with poor ind click prediction performance can produce highly rewarding policies.

Moreover, CACM$^{\ominus}$ is clearly more affected than ARM by the maximisation of the expected CTR, despite having shown better performance and robustness than ARM on the CTR prediction task (see Section 6.2). This suggests that tasks involving maximisation of the expected CTR, such as Fair Ranking and Offline Bandits, are more demanding regarding robustness to policy distributional shift. The expressivity of CACM$^{\ominus}$ and NCM makes them better at fitting the click distribution, but it also critically exposes them to the optimiser's curse, i.e., they are likely to grossly overestimate the expected CTR of at least one ranking which is going to be selected by the maximisation process.

*6.3.1 Upshot.* In summary, this experiment shows (i) that distributional shift critically impacts the policies recovered by poorly robust models in certain downstream tasks, e.g., Fair Ranking and Offline Bandits, and (ii) that simple models should not be overlooked as they can produce highly rewarding policies and be robust under distributional shift.

## 7 DISCUSSION

In the previous section, we have instantiated our proposed evaluation protocol with a selection of six types of click models in order to analyse how well-known models perform in practical scenarios that can be encountered in multiple downstream tasks, which have in common requiring out-of-distribution predictions. Our results allow us to identify general trends, e.g., that fairness and bandits tasks require stricter robustness than CTR prediction tasks or that simpler models are usually more robust out-of-distribution. In this section, we discuss how this protocol can be leveraged by practitioners in a real-world deployment scenario. Note that our protocol allows us to compare different click models with respect to each other. It can be used to do that in two different ways:

A first use case would involve a practitioner wishing to quickly assess a new candidate click model before taking the risk of deploying policies based on it for online evaluation. The instantiation on MSLR data that we describe in our experiments can be used out of the box. In order to make this process easier, we provide code and result files that can be readily used when testing a new candidate click model on MSLR annotations and features, without re-training models included in our experiments.

A second possibility is to adapt the protocol described in Section 5 to a given search engine. Indeed, it relies on semi-synthetic simulators that can be derived from graded relevance annotations (to define click probabilities) and document features (to build realistic logging policies). Because of this semi-synthetic setup, the scenario can be made to fit any real-world search engine, with only the internal user click model left to be controlled by the practitioner. In this setup, comparing candidate click models across a wide range of internal click models is key in order to assess how their robustness is affected by click model mismatch.

Also, in both settings, our experimental setup can be enriched with available context features to fit the scenario at hand. Even though our protocol cannot replace online evaluation, it constitutes a way to reduce the cost of deploying new learning-to-rank models by mitigating the risk of under-performance once deployed.

## 8 CONCLUSION

In this work, we have highlighted the limitations of the traditional offline evaluation protocol for click models, specifically that it fails to detect a lack of robustness of click models to policy distributional shift. To do so, we have implemented several types of click models encoding different structural assumptions of user behavior, and have augmented the evaluation of these models by using simulations that aim to mimic real-world deployment for different downstream tasks involving policy distributional shift.

### 8.1 Main findings

Our experiments highlight the existence of a critical issue with click models: the existing offline evaluation protocol cannot guarantee effective debiasing and robustness to distributional shift. We show that it can cause click models to underperform on the target downstream task because of poor out-of-distribution policy evaluation capabilities, whether it is when predicting the CTR of unknown policies or when training policies based on the click model's parameters.

Three major take-aways emerge from our experiments:

- They show that certain training configurations (strong click model mismatch, near-optimal training policies) are more likely to be negatively affected by distributional shift than others,
- They allow us to identify risky models (DBN, NCM) as well as relatively safer ones (PBM, UBM), and

- They provide practitioners with a way of mitigating the risks of deploying policies based on candidate click models by detecting the lack of robustness before deployment.

### 8.2 Broader implications

Our findings indicate that counterfactual models and estimators must be carefully evaluated in order to make them trustworthy for practical use in downstream tasks and that, despite being convenient, offline metrics can miss important robustness issues in certain settings.

On a more actionable note, our work suggests that getting more diverse test sets, i.e., from different logging policies, should be considered whenever possible. Moreover, simulations can play a role in an offline evaluation protocol by measuring otherwise unobservable metrics, as long as we evaluate on a wide range of simulations so as to mitigate the influence of simulator design. Developing high-quality, learnable simulators matching the dynamics of real-world deployment could also futher mitigate the risks associated with it.

### 8.3 Limitations

We implemented click models in a standardised, context-free fashion in order to fairly compare their resilience to policy distributional shift, but many improvements of these models leveraging context features have been proposed in recent years [4, 8, 13, 52]. We expect the general trends identified in this work to generalise to these context-aware models, but the precise effect of data enrichment with abundant side information and historical behavior remains unaddressed.

### 8.4 Future work

Generalising predictions out-of-distribution is a hard problem exhibiting no theoretical guarantees without further assumptions [43]. In this work we showed that click models, which aim to lift the in-distribution requirement by encoding such assumptions in their architecture, cannot be simply evaluated with traditional offline metrics and human relevance annotations. Future work should therefore investigate under which assumptions we can derive theoretical results for the generalisation capabilities of click models, and how their offline results relate to their online performance when such assumptions are satisfied.

As we have seen in the experiments, the most robust models also tend to be the simplest, whose in-distribution performance is generally subpar compared to more advanced models. Therefore, future work should also investigate strategies to counter the effect of distributional shift in order to attain similar levels of robustness with more complex click models, such as training on multiple logging policies, enforcing invariances or penalizing the epistemic uncertainty.

## SUPPLEMENTARY MATERIALS

To facilitate reproducibility of the results in this paper, we share the code along with guidelines for reproduction at github.com/naver/dist_shift_click_models.

## ACKNOWLEDGMENTS

We would like to thank Thibaut Thonet, Jin Huang and Philipp Hager for their relevant comments as well as Pooya Khandel for help with the Yandex dataset and Till Kletti for insightful discussions.

## REFERENCES

[1] Qingyao Ai, Tao Yang, Huazheng Wang, and Jiaxin Mao. 2021. Unbiased Learning to Rank: Online or Offline? *ACM Trans. Inf. Syst.* 39, 2, Article 21 (Feb. 2021).
[2] Arthur Argenson and Gabriel Dulac-Arnold. 2021. Model-Based Offline Planning. arXiv:2008.05556 [cs.LG]
[3] Alexey Borisov, Ilya Markov, Maarten de Rijke, and Pavel Serdyukov. 2016. A Neural Click Model for Web Search. In

*Proceedings of the 25th International Conference on World Wide Web* (Montréal, Québec, Canada) *(WWW '16)*. IW3C2, Geneva, Switzerland, 531–541.

[4] Alexey Borisov, Martijn Wardenaar, Ilya Markov, and Maarten de Rijke. 2018. A Click Sequence Model for Web Search. In *The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval* (Ann Arbor, MI, USA) *(SIGIR '18)*. ACM, New York, NY, USA, 45–54.

[5] Christopher J. C. Burges. 2010. *From RankNet to LambdaRank to LambdaMART: An Overview*. Technical Report. Microsoft Research. http://research.microsoft.com/en-us/um/people/cburges/tech_reports/MSR-TR-2010-82.pdf

[6] Olivier Chapelle and Ya Zhang. 2009. A Dynamic Bayesian Network Click Model for Web Search Ranking. In *Proceedings of the 18th International Conference on World Wide Web* (Madrid, Spain) *(WWW '09)*. ACM, New York, NY, USA, 1–10. https://doi.org/10.1145/1526709.1526711

[7] Jia Chen, Jiaxin Mao, Yiqun Liu, Min Zhang, and Shaoping Ma. 2019. TianGong-ST: A New Dataset with Large-Scale Refined Real-World Web Search Sessions. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management* (Beijing, China) *(CIKM '19)*. ACM, New York, NY, USA, 2485–2488. https://doi.org/10.1145/3357384.3358158

[8] Jia Chen, Jiaxin Mao, Yiqun Liu, Min Zhang, and Shaoping Ma. 2020. A Context-Aware Click Model for Web Search. In *Proceedings of the 13th International Conference on Web Search and Data Mining* (Houston, TX, USA) *(WSDM '20)*. ACM, New York, NY, USA, 88–96. https://doi.org/10.1145/3336191.3371819

[9] Ye Chen and Tak W Yan. 2012. Position-normalized click prediction in search advertising. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, New York, NY, USA, 795–803.

[10] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. 2016. Wide & Deep Learning for Recommender Systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems* (Boston, MA, USA) *(DLRS 2016)*. Association for Computing Machinery, New York, NY, USA, 7–10. https://doi.org/10.1145/2988450.2988454

[11] Aleksandr Chuklin, Ilya Markov, and Maarten de Rijke. 2015. *Click Models for Web Search*. Morgan & Claypool.

[12] Nick Craswell, Onno Zoeter, Michael Taylor, and Bill Ramsey. 2008. An Experimental Comparison of Click Position-Bias Models. In *Proceedings of the 2008 International Conference on Web Search and Data Mining* (Palo Alto, CA, USA) *(WSDM '08)*. ACM, New York, NY, USA, 87–94. https://doi.org/10.1145/1341531.1341545

[13] Xinyi Dai, Jianghao Lin, Weinan Zhang, Shuai Li, Weiwen Liu, Ruiming Tang, Xiuqiang He, Jianye Hao, Jun Wang, and Yong Yu. 2021. An Adversarial Imitation Click Model for Information Retrieval. In *Proceedings of the Web Conference 2021* (Ljubljana, Slovenia) *(WWW '21)*. ACM, New York, NY, USA, 1809–1820. https://doi.org/10.1145/3442381.3449913

[14] Georges E. Dupret and Benjamin Piwowarski. 2008. A User Browsing Model to Predict Search Engine Click Data from Past Observations.. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Singapore, Singapore) *(SIGIR '08)*. ACM, New York, NY, USA, 331–338. https://doi.org/10.1145/1390334.1390392

[15] Artem Grotov, Aleksandr Chuklin, Ilya Markov, Luka Stout, Finde Xumara, and Maarten de Rijke. 2015. A Comparative Study of Click Models for Web Search. In *CLEF 2015*. Springer, Cham, Switzerland, 78–90.

[16] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A Factorization-Machine Based Neural Network for CTR Prediction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence* (Melbourne, Australia) *(IJCAI'17)*. AAAI Press, 1725–1731.

[17] Jin Huang, Harrie Oosterhuis, Maarten de Rijke, and Herke van Hoof. 2020. Keeping Dataset Biases out of the Simulation: A Debiased Simulator for Reinforcement Learning Based Recommender Systems. In *Fourteenth ACM Conference on Recommender Systems* (Virtual Event, Brazil) *(RecSys '20)*. ACM, New York, NY, USA, 190–199. https://doi.org/10.1145/3383313.3412252

[18] Rolf Jagerman, Ilya Markov, and Maarten de Rijke. 2019. When People Change their Mind: Off-policy Evaluation in Non-stationary Recommendation Environments. In *WSDM 2019: 12th International Conference on Web Search and Data Mining*. ACM, 447–455.

[19] Michael Janner, Qiyang Li, and Sergey Levine. 2021. Reinforcement Learning as One Big Sequence Modeling Problem. arXiv:2106.02039 [cs.LG]

[20] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated Gain-Based Evaluation of IR Techniques. *ACM Trans. Inf. Syst.* 20, 4 (Oct. 2002), 422–446. https://doi.org/10.1145/582415.582418

[21] Olivier Jeunen and Bart Goethals. 2021. Pessimistic Reward Models for Off-Policy Learning in Recommendation. In *Fifteenth ACM Conference on Recommender Systems* (Amsterdam, Netherlands) *(RecSys '21)*. Association for Computing Machinery, New York, NY, USA, 63–74. https://doi.org/10.1145/3460231.3474247

[22] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. 2005. Accurately Interpreting Clickthrough Data as Implicit Feedback. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Salvador, Brazil) *(SIGIR '05)*. ACM, New York, NY, USA, 154–161.

https://doi.org/10.1145/1076034.1076063
[23] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, Filip Radlinski, and Geri Gay. 2007. Evaluating the Accuracy of Implicit Feedback from Clicks and Query Reformulations in Web Search. *ACM Trans. Inf. Syst.* 25, 2 (April 2007), 7–es. https://doi.org/10.1145/1229179.1229181
[24] Thorsten Joachims, Adith Swaminathan, and Tobias Schnabel. 2017. Unbiased Learning-to-Rank with Biased Feedback. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining* (Cambridge, United Kingdom) *(WSDM '17)*. Association for Computing Machinery, New York, NY, USA, 781–789. https://doi.org/10.1145/3018661.3018699
[25] Branislav Kveton, Csaba Szepesvári, Zheng Wen, and Azin Ashkan. 2015. Cascading Bandits: Learning to Rank in the Cascade Model. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37* (Lille, France) *(ICML'15)*. JMLR.org, 767–776.
[26] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. 2020. Offline Reinforcement Learning: Tutorial, Review, and Perspectives on Open Problems. arXiv:2005.01643 [cs.LG]
[27] Shuai Li, Yasin Abbasi-Yadkori, Branislav Kveton, S. Muthukrishnan, Vishwa Vinay, and Zheng Wen. 2018. Offline Evaluation of Ranking Policies with Click Models. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (London, United Kingdom) *(KDD '18)*. ACM, New York, NY, USA, 1685–1694. https://doi.org/10.1145/3219819.3220028
[28] Jianghao Lin, Weiwen Liu, Xinyi Dai, Weinan Zhang, Shuai Li, Ruiming Tang, Xiuqiang He, Jianye Hao, and Yong Yu. 2021. A Graph-Enhanced Click Model for Web Search. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, Canada) *(SIGIR '21)*. Association for Computing Machinery, New York, NY, USA, 1259–1268. https://doi.org/10.1145/3404835.3462895
[29] Yiqun Liu, Xiaohui Xie, Chao Wang, Jian-Yun Nie, Min Zhang, and Shaoping Ma. 2016. Time-Aware Click Model. *ACM Trans. Inf. Syst.* 35, 3, Article 16 (dec 2016), 24 pages. https://doi.org/10.1145/2988230
[30] Bruce Lowerre. 1976. The HARPY speech recognition system. *The Journal of the Acoustical Society of America* 60, S1 (1976).
[31] Duncan Luce. 1959. *Individual Choice Behavior: A Theoretical Analysis*. Courier Corporation.
[32] James McInerney, Brian Brost, Praveen Chandar, Rishabh Mehrotra, and Benjamin Carterette. 2020. Counterfactual Evaluation of Slate Recommendations with Sequential Reward Interactions. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, New York, NY, USA, 1779–1788.
[33] H Brendan McMahan, Gary Holt, David Sculley, Michael Young, Dietmar Ebner, Julian Grady, Lan Nie, Todd Phillips, Eugene Davydov, Daniel Golovin, et al. 2013. Ad Click Prediction: A View from the Trenches. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, New York, NY, USA, 1222–1230.
[34] Harrie Oosterhuis. 2021. *Computationally Efficient Optimization of Plackett-Luce Ranking Models for Relevance and Fairness*. Association for Computing Machinery, New York, NY, USA, 1023–1032. https://doi.org/10.1145/3404835.3462830
[35] Harrie Oosterhuis and Maarten de Rijke. 2020. *Policy-Aware Unbiased Learning to Rank for Top-k Rankings*. Association for Computing Machinery, New York, NY, USA, 489–498. https://doi.org/10.1145/3397271.3401102
[36] Harrie Oosterhuis and Maarten de Rijke. 2021. Unifying Online and Counterfactual Learning to Rank: A Novel Counterfactual Estimator That Effectively Utilizes Online Interventions. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining* (Virtual Event, Israel) *(WSDM '21)*. ACM, New York, NY, USA, 463–471. https://doi.org/10.1145/3437963.3441794
[37] Robin Plackett. 1975. The Analysis of Permutations. *Journal of the Royal Statistical Society* 24, 2 (1975). https://doi.org/10.2307/2346567
[38] Doina Precup, Richard S. Sutton, and Satinder P. Singh. 2000. Eligibility Traces for Off-Policy Policy Evaluation. In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML '00)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 759–766.
[39] Tao Qin and Tie-Yan Liu. 2013. Introducing LETOR 4.0 Datasets. abs/1306.2597.
[40] Stephen E. Robertson. 1977. The Probability Ranking Principle in IR. *Journal of Documentation* 33, 4 (1977), 294–304.
[41] Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. 1994. Okapi at TREC-3. In *Proceedings of The Third Text REtrieval Conference, TREC (NIST Special Publication, Vol. 500-225)*. National Institute of Standards and Technology (NIST), Gaithersburg, MD, USA, 109–126.
[42] Pavel Serdyukov, Nick Craswell, and Georges Dupret. 2012. WSCD 2012: Workshop on Web Search Click Data 2012. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining* (Seattle, Washington, USA) *(WSDM '12)*. ACM, New York, NY, USA, 771–772. https://doi.org/10.1145/2124295.2124396
[43] Zheyan Shen, Jiashuo Liu, Yue He, Xingxuan Zhang, Renzhe Xu, Han Yu, and Peng Cui. 2021. Towards Out-Of-Distribution Generalization: A Survey. https://doi.org/10.48550/ARXIV.2108.13624
[44] Ashudeep Singh and Thorsten Joachims. 2018. Fairness of Exposure in Rankings. In *Proceedings of the 24th ACM*

*SIGKDD International Conference on Knowledge Discovery and Data Mining* (London, United Kingdom) *(KDD '18)*. ACM, New York, NY, USA, 2219–2228. https://doi.org/10.1145/3219819.3220088

[45] James Smith and Robert Winkler. 2006. The Optimizer's Curse: Skepticism and Postdecision Surprise in Decision Analysis. *Management Science* 52 (2006).

[46] Harald Steck. 2010. Training and Testing of Recommender Systems on Data Missing Not at Random. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, New York, USA.

[47] Adith Swaminathan, Akshay Krishnamurthy, Alekh Agarwal, Miroslav Dudík, John Langford, Damien Jose, and Imed Zitouni. 2017. Off-Policy Evaluation for Slate Recommendation. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 3635–3645.

[48] Ali Vardasbi, Maarten de Rijke, and Ilya Markov. 2020. Cascade Model-Based Propensity Estimation for Counterfactual Learning to Rank. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, New York, NY, USA, 2089–2092.

[49] Ali Vardasbi, Harrie Oosterhuis, and Maarten de Rijke. 2020. When Inverse Propensity Scoring Does Not Work: Affine Corrections for Unbiased Learning to Rank. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*. ACM, New York, NY, USA, 1475–1484.

[50] Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. 2020. MOPO: Model-based Offline Policy Optimization. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates.

[51] Junqi Zhang, Yiqun Liu, Jiaxin Mao, Weizhi Ma, Jiazheng Xu, Shaoping Ma, and Qi Tian. 2022. User Behavior Simulation for Search Result Re-Ranking. *ACM Trans. Inf. Syst.* (jan 2022). https://doi.org/10.1145/3511469 Just Accepted.

[52] Yukun Zheng, Jiaxin Mao, Yiqun Liu, Cheng Luo, Min Zhang, and Shaoping Ma. 2019. Constructing Click Model for Mobile Search with Viewport Time. *ACM Trans. Inf. Syst.* 37, 4 (sep 2019), 34 pages. https://doi.org/10.1145/3360486

## A TRAINING AND IMPLEMENTATION DETAILS

All click models are implemented in PyTorch with PyTorch-Lightning. They are trained using the Adam Optimiser with the ReduceLROnPlateau scheduler with a factor of 0.5 and patience of 2. We use Early Stopping with patience of 3 to stop the training when the validation loss does not improve and restore the best checkpoint for evaluation on the test set. We trained these models on a single nVIDIA V100 GPU and no models required more than 10 minutes to be trained on the simulated datasets.

## B DEFINITION OF COCM

We consider three modes: the user browses the page in a top-down fashion ($\triangledown$), in a bottom-up fashion ($\triangle$), or does not look at the page at all and clicks on documents without examining them ($\bigcirc$). In our experiments, we take the respective probabilities of each mode to be $(0.6, 0.3, 0.1)$ for CoCM and $(0.2, 0.7, 0.3)$ for CoCM mismatch.

- In the top-down mode, the click probability of document $d$ at rank $k$ depends on its attractiveness $\alpha_{q,d}$, whether the preceding document has been clicked $c_{k-1}$, and the attractiveness of the next document $\alpha_{q,d_{k+1}}$. With $A_k$, $A_{k+1}$, $C_{k-1}$ and $E_k|C_{<k}$ jointly independent, we have:

$$\begin{aligned} P(C_k = 1 \mid q, d_k, d_{k+1}, c_{k-1} = 1, \triangledown) &= 0 \\ P(C_k = 1 \mid q, d_k, d_{k+1}, c_{k-1} = 0, c_{<k-1}, \triangledown) &= \alpha_{q,d_k} \times (1 - \alpha_{q,d_{k+1}}/2) \times \\ &\quad P(E_k = 1 \mid c_{k-1} = 0, c_{<k-1}) \\ P(E_k = 1 \mid c_{k-1} = 1, c_{<k-1}) &= (1 - \sigma) \times \gamma \times P(E_{k-1} = 1 \mid c_{<k-1}) \\ P(E_k = 1 \mid c_{k-1} = 0, c_{<k-1}) &= \gamma \times P(E_{k-1} = 1 \mid c_{<k-1}). \end{aligned} \tag{17}$$

  We take $\alpha_{q,d} = (2^{\text{rel}(q,d)} - 1)/15$, $\sigma = 0.7$ and $\gamma = 0.9$. Note that even the top-down mode does not follow the cascade hypothesis as the click probability depends on the relevance of the following document.
- The order of next and previous documents and clicks is simply reversed in the bottum-up

mode.

- In the no-look mode, the probability of the document at rank $k$ being clicked is $P(C_k = 1 \mid \bigcirc) = \epsilon_k$ with $\epsilon_k = 0.2 \times 0.9^k$.

## C FIGURE 2 WITH CONFIDENCE BOUNDS

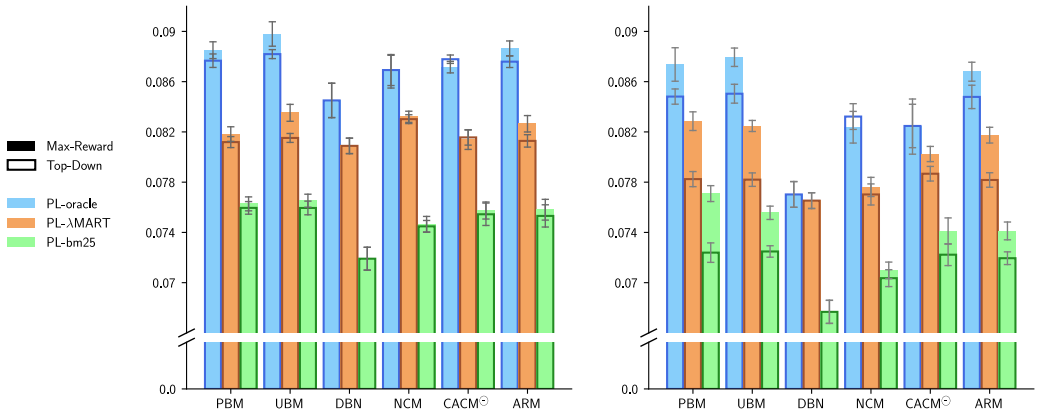


Fig. 3. Click-through rate obtained by the Top-Down and Max-Reward policies of click models. Left: CoCM as internal click model. Right: CoCM mismatch as internal click model.

## D TABLES 1 (LEFT) AND 2 (RIGHT) WITH CONFIDENCE BOUNDS

Table 5. Results on CLARA with 95% confidence bounds.

| Click Model | PPL | PPL@1 | PPL@2 | PPL@5 | PPL@10 | nDCG@1 | nDCG@3 | nDCG@5 | nDCG@10 |
|---|---|---|---|---|---|---|---|---|---|
| RankTopObs | – | – | – | – | – | 0.7138±0.0002 | 0.7003±0.0001 | 0.7291±0.0001 | 0.8034±0.0001 |
| TopPopObs | – | – | – | – | – | 0.7206±0.0009 | 0.7023±0.0009 | 0.7319±0.0005 | 0.8056±0.0005 |
| TopPop | – | – | – | – | – | 0.7225±0.0007 | 0.7110±0.0010 | 0.7374±0.0004 | 0.8097±0.0005 |
| dCTR | 1.3212±0.0000 | 1.6054±0.0000 | 1.5581±0.0000 | 1.2901±0.0000 | 1.1790±0.0000 | **0.7246±0.0010** | **0.7165±0.0006** | **0.7456±0.0005** | **0.8155±0.0003** |
| drCTR | 1.3146±0.0000 | **1.5784±0.0000** | 1.5439±0.0000 | 1.2895±0.0000 | 1.1764±0.0000 | 0.6273±0.0002 | 0.6531±0.0001 | 0.6989±0.0001 | 0.7795±0.0001 |
| PBM | 1.3177±0.0013 | 1.5970±0.0039 | 1.5488±0.0016 | 1.2884±0.0009 | 1.1781±0.0011 | 0.5977±0.0076 | †0.6243±0.0044 | †0.6734±0.0038 | †0.7621±0.0028 |
| UBM | 1.2823±0.0011 | 1.5918±0.0035 | 1.5366±0.0018 | 1.2463±0.0009 | 1.1247±0.0011 | †0.6017±0.0083 | †0.6255±0.0050 | †0.6734±0.0033 | †0.7631±0.0024 |
| DBN | 1.2803±0.0008 | 1.5861±0.0019 | 1.5200±0.0012 | 1.2463±0.0006 | 1.1271±0.0005 | 0.5785±0.0065 | 0.6096±0.0030 | 0.6610±0.0024 | 0.7529±0.0018 |
| NCM | **1.2717±0.0011** | 1.5842±0.0023 | **1.5141±0.0025** | **1.2361±0.0011** | **1.1152±0.0008** | 0.5606±0.0057 | 0.5987±0.0043 | 0.6518±0.0036 | 0.7462±0.0027 |
| CACM$^{\ominus}$ | 1.2789±0.0013 | 1.5969±0.0054 | 1.5443±0.0032 | 1.2405±0.0009 | 1.1165±0.0004 | 0.5648±0.0105 | 0.6008±0.0063 | .6546±0.0048 | 0.7485±0.0036 |
| ARM | 1.3147±0.0023 | 1.6142±0.0069 | 1.5755±0.0033 | 1.2719±0.0012 | 1.1807±0.0042 | 0.5993±0.0048 | †0.6258±0.0030 | †0.6730±0.0022 | †0.7616±0.0018 |
| ARM NC | – | – | – | – | – | 0.6086±0.0084 | 0.6279±0.0040 | 0.6756±0.0050 | 0.7634±0.0033 |

Table 6. Results on Yandex with 95% confidence bounds.

| Click Model | PPL | PPL@1 | PPL@2 | PPL@5 | PPL@10 | nDCG@1 | nDCG@3 | nDCG@5 | nDCG@10 |
|---|---|---|---|---|---|---|---|---|---|
| RankTopObs | – | – | – | – | – | 0.7138±0.0002 | 0.7003±0.0001 | 0.7291±0.0001 | 0.8034±0.0001 |
| TopPopObs | – | – | – | – | – | 0.7206±0.0009 | 0.7023±0.0009 | 0.7319±0.0005 | 0.8056±0.0005 |
| TopPop | – | – | – | – | – | 0.7225±0.0007 | 0.7110±0.0010 | 0.7374±0.0004 | 0.8097±0.0005 |
| dCTR | 1.3212±0.0000 | 1.6054±0.0000 | 1.5581±0.0000 | 1.2901±0.0000 | 1.1790±0.0000 | **0.7246±0.0010** | **0.7165±0.0006** | **0.7456±0.0005** | **0.8155±0.0003** |
| drCTR | 1.3146±0.0000 | **1.5784±0.0000** | 1.5439±0.0000 | 1.2895±0.0000 | 1.1764±0.0000 | 0.6273±0.0002 | 0.6531±0.0001 | 0.6989±0.0001 | 0.7795±0.0001 |
| PBM | 1.3177±0.0013 | 1.5970±0.0039 | 1.5488±0.0016 | 1.2884±0.0009 | 1.1781±0.0011 | 0.5977±0.0076 | †0.6243±0.0044 | †0.6734±0.0038 | †0.7621±0.0028 |
| UBM | 1.2823±0.0011 | 1.5918±0.0035 | 1.5366±0.0018 | 1.2463±0.0009 | 1.1247±0.0011 | †0.6017±0.0083 | †0.6255±0.0050 | †0.6734±0.0033 | †0.7631±0.0024 |
| DBN | 1.2803±0.0008 | 1.5861±0.0019 | 1.5200±0.0012 | 1.2463±0.0006 | 1.1271±0.0005 | 0.5785±0.0065 | 0.6096±0.0030 | 0.6610±0.0024 | 0.7529±0.0018 |
| NCM | **1.2717±0.0011** | 1.5842±0.0023 | **1.5141±0.0025** | **1.2361±0.0011** | **1.1152±0.0008** | 0.5606±0.0057 | 0.5987±0.0043 | 0.6518±0.0036 | 0.7462±0.0027 |
| CACM$^{\ominus}$ | 1.2789±0.0013 | 1.5969±0.0054 | 1.5443±0.0032 | 1.2405±0.0009 | 1.1165±0.0004 | 0.5648±0.0105 | 0.6008±0.0063 | .6546±0.0048 | 0.7485±0.0036 |
| ARM | 1.3147±0.0023 | 1.6142±0.0069 | 1.5755±0.0033 | 1.2719±0.0012 | 1.1807±0.0042 | 0.5993±0.0048 | †0.6258±0.0030 | †0.6730±0.0022 | †0.7616±0.0018 |
| ARM NC | – | – | – | – | – | 0.6086±0.0084 | 0.6279±0.0040 | 0.6756±0.0050 | 0.7634±0.0033 |
| PBM | 1.1445±0.0014 | =1.3984±0.0084 | 1.2523±0.0009 | 1.1368±0.0006 | 1.0546±0.005 | 0.6801±0.0051 | 0.7396±0.0029 | 0.7993±0.0019 | 0.8736±0.0016 |
| UBM | 1.1410±0.0000 | **1.3906±0.0003** | 1.2431±0.0002 | 1.1343±0.0001 | 1.0519±0.0002 | 0.6825±0.0039 | 0.7411±0.0023 | 0.8002±0.0016 | 0.8744±0.0012 |
| DBN | 1.1386±0.0013 | =1.3936±0.0061 | =1.2282±0.0009 | 1.1283±0.0008 | 1.0523±0.0005 | 0.6665±0.0067 | 0.7355±0.0030 | 0.7967±0.0023 | 0.8715±0.0016 |
| NCM | **1.1371±0.0002** | 1.3994±0.0034 | **1.2274±0.0005** | **1.1254±0.0007** | **1.0497±0.0002** | 0.6554±0.0047 | 0.7311±0.0027 | 0.7942±0.0022 | 0.8688±0.0011 |
| CACM$^{\ominus}$ | 1.1417±0.0018 | =1.3962±0.0085 | 1.2437±0.0009 | 1.1337±0.0010 | 1.0519±0.0009 | 0.6808±0.0042 | 0.7406±0.0017 | 0.8001±0.0010 | 0.8740±0.0010 |
| ARM | 1.1438±0.0014 | 1.3994±0.0066 | 1.2478±0.0007 | 1.1350±0.0007 | 1.0566±0.0010 | †0.6861±0.0076 | †0.7442±0.0027 | †0.8027±0.0018 | †0.8760±0.0015 |
| ARM NC | – | – | – | – | – | 0.6930±0.0048 | 0.7456±0.0015 | 0.8035±0.0012 | 0.8768±0.0012 |

## E OFFLINE METRICS ON CLARA AND YANDEX : AUROC AND RECALL

Table 7. Area under the receiving operating characteristic curve (AUROC) and Recall on the CLARA dataset. ① are naive baselines, ② as-is click models, and ③ modified click models; see Section 3.2. The best performing model in average is reported in bold and $^{=}$ indicates a result is not significantly worse than the best performing model. ↓ : lower is better; ↑ : higher is better. AUROC@k is computed only on documents at rank $k$ while Recall@$k$ is computed on documents up to rank $k$, similarly to respectively PPL@$k$ and nDCG@$k$. Also, AUROC "Full" is computed over the whole dataset.

| | | AUROC ↑ | | | | | Recall ↑ | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | **Click model** | Full | @1 | @2 | @5 | @10 | @1 | @3 | @5 | @10 |
| ① | RankTopObs | – | – | – | – | – | $^{=}$0.2521 | **0.5814** | 0.7908 | 0.9830 |
| | TopPopObs | – | – | – | – | – | 0.2502 | 0.5789 | **0.7929** | $^{=}$0.9838 |
| | TopPop | – | – | – | – | – | **0.2525** | 0.5799 | $^{=}$0.7927 | **0.9842** |
| | dCTR | 0.9214 | 0.8970 | 0.7924 | 0.7902 | 0.7888 | 0.2393 | 0.5445 | 0.7600 | 0.9802 |
| | drCTR | 0.9255 | 0.9004 | 0.8028 | 0.8097 | 0.8337 | 0.2345 | 0.5376 | 0.7540 | 0.9790 |
| ② | PBM | 0.9161 | $^{=}$0.9002 | 0.7929 | 0.7998 | 0.8160 | 0.1997 | 0.4958 | 0.7354 | 0.9819 |
| | UBM | $^{=}$0.9255 | $^{=}$0.9027 | $^{=}$0.8100 | $^{=}$0.8210 | $^{=}$0.8531 | 0.2020 | 0.4984 | 0.7358 | 0.9819 |
| | DBN | $^{=}$0.9244 | 0.8961 | **0.8110** | 0.8195 | $^{=}$0.8384 | 0.1891 | 0.4903 | 0.7311 | 0.9828 |
| | NCM | **0.9292** | **0.9037** | $^{=}$0.8098 | **0.8257** | **0.8576** | 0.1873 | 0.4889 | 0.7320 | 0.9819 |
| ③ | CACM$^{\ominus}$ | $^{=}$0.9242 | $^{=}$0.9010 | $^{=}$0.8094 | $^{=}$0.8219 | $^{=}$0.8546 | 0.1899 | 0.4900 | 0.7320 | 0.9819 |
| | ARM | 0.8906 | 0.8815 | 0.7630 | 0.7724 | 0.7702 | 0.1975 | 0.4936 | 0.7337 | 0.9821 |
| | ARM NC | – | – | – | – | – | 0.1995 | 0.4955 | 0.7342 | 0.9817 |

Table 8. AUROC and Recall on the Yandex dataset. Same conventions as in Table 7.

| | | AUROC ↑ | | | | | Recall ↑ | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | **Click model** | Full | @1 | @2 | @5 | @10 | @1 | @3 | @5 | @10 |
| ① | RankTopObs | – | – | – | – | – | 0.7138 | 0.7003 | 0.7291 | 0.8034 |
| | TopPopObs | – | – | – | – | – | 0.2139 | 0.4693 | 0.6572 | 0.9127 |
| | TopPop | – | – | – | – | – | $^{=}$0.2176 | 0.4791 | 0.6643 | 0.9149 |
| | dCTR | 0.8766 | 0.8014 | 0.7547 | 0.7624 | 0.7697 | **0.2185** | **0.4864** | **0.6752** | **0.9207** |
| | drCTR | 0.8819 | **0.8135** | 0.7670 | 0.7614 | 0.7675 | 0.1778 | 0.4496 | 0.6547 | 0.9157 |
| ② | PBM | 0.8798 | 0.8035 | 0.7603 | 0.7622 | 0.7575 | 0.1670 | 0.4284 | 0.6364 | 0.9106 |
| | UBM | 0.9149 | 0.8057 | 0.7749 | 0.8557 | 0.9194 | 0.1692 | 0.4289 | 0.6345 | 0.9108 |
| | DBN | 0.9166 | 0.8068 | $^{=}$0.7932 | 0.8586 | 0.9199 | 0.1588 | 0.4199 | 0.6283 | 0.9076 |
| | NCM | **0.9222** | 0.8090 | **0.7952** | **0.8689** | **0.9264** | 0.1539 | 0.4133 | 0.6233 | 0.9061 |
| ③ | CACM$^{\ominus}$ | 0.9182 | 0.8008 | 0.7675 | 0.8630 | 0.9245 | 0.1559 | 0.4153 | 0.6256 | 0.9067 |
| | ARM | 0.8865 | 0.7988 | 0.7648 | 0.8102 | 0.8111 | 0.1656 | 0.4288 | 0.6348 | 0.9105 |
| | ARM NC | – | – | – | – | – | 0.1681 | 0.4286 | 0.6362 | 0.9108 |