Online Expectation-Maximization for Click Models

Ilya Markov
University of Amsterdam
Amsterdam, The Netherlands
i.markov@uva.nl

Alexey Borisov
Yandex
Moscow, Russia
University of Amsterdam
Amsterdam, The Netherlands
alborisov@yandex-team.ru

Maarten de Rijke
University of Amsterdam
Amsterdam, The Netherlands
derijke@uva.nl

ABSTRACT

Click models allow us to interpret user click behavior in search interactions and to remove various types of bias from user clicks. Existing studies on click models consider a static scenario where user click behavior does not change over time. We show empirically that click models deteriorate over time if retraining is avoided. We then adapt online expectation-maximization (EM) techniques to efficiently incorporate new click/skip observations into a trained click model. Our instantiation of *Online EM* for click models is orders of magnitude more efficient than retraining the model from scratch using standard EM, while loosing little in quality. To deal with outdated click information, we propose a variant of online EM called *EM with Forgetting*, which surpasses the performance of complete retraining while being as efficient as Online EM.

CCS CONCEPTS

•Information systems →Web search engines; Information retrieval; Retrieval models and ranking;

1 INTRODUCTION

Click models have been developed to interpret user click behavior in search and to convert biased clicks on search results into unbiased estimates of the results' relevance [3]. Click models are used in various information retrieval tasks, such as ranking [1, 4], evaluation [2, 12] and user simulation [11, 16].

Existing studies of click models assume that user click behavior does not change over time. In this static scenario, click models are trained using a historical click log and evaluated on a limited test set of "future" clicks. This static assumption rarely holds in practice [14]. User click behavior changes over time and it is, therefore, important to keep click models up to date. This problem has been identified previously [9, 10], but has not yet been studied extensively. We recognize two challenges in keeping click models up to date. First, how to efficiently incorporate newly observed information into a trained click model? Second, how to remove outdated information from a trained click model?

CIKM'17, November 6–10, 2017, Singapore.

ISBN 978-1-4503-4918-5/17/11…$15.00
DOI: http://dx.doi.org/10.1145/3132847.3133053

We consider the most widely used and effective inference technique for click models, i.e., expectation-maximization, and study it in an online scenario. We propose two methods to adapt EM to online settings: Online EM, which efficiently incorporates new click information into click models, and EM with Forgetting, which deals with outdated clicks. We show experimentally that Online EM is orders of magnitude more efficient than complete retraining, while loosing little in the quality of updated click models; EM with Forgetting can surpass the effectiveness of complete retraining.

2 BACKGROUND AND RELATED WORK

Most click models operate with binary random variables and corresponding probabilities, e.g., how attractive a document is given a query, what the probability of examining a certain position on a SERP is, etc. These probabilities are the parameters of click models and they need to be estimated from observed clicks and skips.

The parameters of click models are usually calculated using either maximum likelihood estimation (MLE) or expectation-maximization (EM) [3]. MLE parameter estimation is used when a click model does not have hidden random variables. MLE does a single pass over a click log and calculates the model parameters directly based on the observed clicks. Thus, click models that use MLE are fast to train and straightforward to update in online settings. However, these models have been shown to perform worse than models that require EM for parameter estimation [6].

EM parameter estimation is used when a click model has hidden variables. EM estimates the model parameters iteratively. At each iteration, the current values of the parameters are fixed and new values are calculated based on user clicks and current values. At the end of each iteration, the values of the parameters are updated. Click models that use EM are much slower to train than ones that use MLE, but display better performance [6]. However, click models that use EM cannot be directly updated in online settings. When new user clicks become available, such models have to be retrained completely using all clicks observed so far. We address this problem and solve it by adopting online EM algorithms [13, 15]. To the best of our knowledge, no previous work has dealt with outdated clicks.

3 ONLINE EXPECTATION-MAXIMIZATION

Online scenario and EM with Retraining. Various applications of click models (e.g., ranking, model-based evaluation, etc.) operate with click models trained using a historical click log. We assume that we have a click model M that is trained on n past observations of clicks/skips. As time passes, new clicks/skips are observed that can be used to update the trained model M. We would like to update M after observing m new clicks/skips, where $m \geq 1$. The task of

online parameter estimation is, thus, to consider $n+m$ observations for training (or updating) the click model M.

Completely re-training the model parameters using EM inference has a computational complexity of $O([n+m]\cdot K)$, where K is the number of iterations. If we perform T consecutive updates of size m, then the complexity becomes $O([n+m\cdot T]\cdot K)$ for the T-th update. Thus, T consecutive updates, each considering m new observations, have a complexity of $O([n\cdot T+m\cdot\sum_{i=1}^{T} i]\cdot K) = O([n\cdot T+m\cdot T^2]\cdot K)$, which is quadratic in the number of updates. In the rest of the paper, we refer to this approach as *EM with Retraining*.

Online EM. Online (or incremental) EM reduces the computational cost of EM with Retraining by updating the model parameters using only a part of the available data [13, 15]. Following this idea, we adapt EM inference for click models so as to update the models' parameters in online settings.

Most click models (and particularly those that use EM) consist of Bernoulli-distributed random variables $X \sim Bernoulli(\theta)$, where θ is a parameter corresponding to a random variable X. For example, most click models have an attractiveness random variable $A_{qd} \sim Bernoulli(\alpha_{qd})$ that is equal to 1 if, and only if, document d is attractive to a user given query q. In a number of click models, such as UBM, DBN, and CCM, this random variable is not observed and so its corresponding parameter α_{qd} has to be estimated using EM inference.

In general, the parameter θ of a Bernoulli-distributed random variable X is computed as the number of query sessions where $X=1$, divided by the number of query sessions where X is defined:

$$\theta = \frac{1}{|\mathcal{S}_X|}\sum_{s\in\mathcal{S}_X} P(X^{(s)}=1 \mid \mathbf{C}^{(s)}), \tag{1}$$

where $\mathcal{S}_X$ is the set of query sessions where random variable X is defined, $X^{(s)}$ is the value of X in a particular query session s and $\mathbf{C}^{(s)}$ is the vector of clicks and skips observed in session s.

However, if a random variable X is not observed (e.g., A_{qd} is not observed in UBM, DBN and CCM), the probability $P(X^{(s)}=1 \mid \mathbf{C}^{(s)})$ cannot be computed directly from clicks $\mathbf{C}^{(s)}$. Instead, this probability should be estimated based both on the observed clicks/skips $\mathbf{C}^{(s)}$ and the current values of all the parameters of a click model (we will denote these values as $\mathbf{\Psi}^{(k)}$, where k is the iteration counter). Then, the parameter θ should be updated iteratively as follows:

$$\theta^{(k+1)} = \frac{1}{|\mathcal{S}_X|}\sum_{s\in\mathcal{S}_X} P(X^{(s)}=1 \mid \mathbf{C}^{(s)}, \mathbf{\Psi}^{(k)}). \tag{2}$$

In an online scenario, we have a click model M that is trained on n past observations of clicks/skips. In other words, all the parameters of M are already estimated, i.e., the sum in Eq. 2 is calculated for all θ's. Let P_X denote this sum. Then, the learned parameter, denoted as $\theta^{(curr)}$, can be written as $\theta^{(curr)} = P_X/|\mathcal{S}_X|$. When we observe a new query session s with clicks $\mathbf{C}^{(s)}$, we need to update the current value $\theta^{(curr)}$ taking into account both the already calculated quantity P_X and the new observations $\mathbf{C}^{(s)}$. Iteratively re-estimating all the parameters of the model M for every newly observed query session is prohibitively slow in practice. Instead, we adapt the online EM inference method [13, 15] and update the value of the click model parameters as follows:

$$\theta^{(new)} = \frac{P_X + P(X^{(s)}=1 \mid \mathbf{C}^{(s)}, \mathbf{\Psi}^{(k)})}{|\mathcal{S}_X|+1}, \tag{3}$$

where the probability added to P_X is the same as in Eq. 2.

Eq. 3 allows us to update the model parameters instantly, moving from the current values $\theta^{(curr)}$ to the new values $\theta^{(new)}$ as soon as new clicks/skips $\mathbf{C}^{(s)}$ are observed. For this reason, we call this approach *Online EM*.

The complexity of Online EM is linear with respect to the number of observations, i.e., it is $O(m)$ for m new clicks/skips no matter how many updates are performed. T consecutive updates, each considering m new observations, have a complexity of $O(m\cdot T)$, which is linear with respect to the number of updates. Moreover, the complexity of this approach does not depend on the number of past observations n (which could have a significant impact on the execution time in case $n \gg m$) and the number of iterations K.

EM with Forgetting. Online EM addresses the problem of how to efficiently update click models in an online scenario by considering newly observed clicks/skips. The second challenge here is how to deal with past clicks, which may become outdated as time passes. In this section we adopt the idea of "forgetting" (or "aging") [8, 13] and propose the *EM with Forgetting* method that discounts past observations depending on their age. In particular, during every update of the model parameters, a certain percentage of past observations, denoted η here, can be "forgotten" as follows:

$$\theta^{(new)} = \frac{P_X\cdot(1-\eta) + P(X^{(s)}=1 \mid \mathbf{C}^{(s)}, \mathbf{\Psi}^{(k)})}{|\mathcal{S}_X|\cdot(1-\eta)+1}. \tag{4}$$

The forgetting ratio η has the following interpretation. Assume that after observing m new clicks/skips and performing m updates for a particular query-document pair, we need to forget x% of past observation. Then $(1-\eta)^m = 1-x$, so $\eta = 1-\sqrt[m]{1-x}$. E.g., if we would like to forget 50% of past observations after 20 updates, then $\eta \approx 0.034$. To forget 10% after 10 updates, we need to set $\eta \approx 0.01$.

The computational complexity of this approach is the same as for Online EM, i.e., $O(m)$ for a single update of size m and $O(m\cdot T)$ for T consecutive updates.

4 EXPERIMENTAL SETUP

Dataset. There exist several publicly available datasets with click logs [3]. The one suitable for our task is the Yandex personalized web search challenge dataset (PWSC),[1] because it contains both the click/skip and time information. The timestamps provided are at the level of days. We use the first two weeks of the dataset, consisting of 33,310,079 query sessions, as historical data and the next 13 days, consisting of 31,862,774 query sessions, to simulate the incoming click stream.

Methodology. To evaluate our proposed online EM methods, we use the following experimental protocol:

(1) Consider a click model M, trained on days $1,\ldots,x$ of the click log. Initially, $x=14$.
(2) Evaluate M using the click log of day $x+1$.
(3) Update M using data collected on day $x+1$.
(4) Increment x and repeat steps 1–3 until all days of the dataset have been considered, i.e., $x+1=27$.

Since the online scenario for click models has not been studied yet, there are no standard baselines to compare against for online click modeling. To get an understanding of the relative efficiency and

[1] https://www.kaggle.com/c/yandex-personalized-web-search-challenge

Table 1: Performance of click models with EM inference trained on the first two weeks of the click log and evaluated on the next 13 days. $^{\Delta}$ denotes the best click model with statistically significant differences compared to other models.

Model	Log-likelihood	Perplexity
UBM	-0.2204^{Δ}	1.2825^{Δ}
DBN	−0.3523	1.3135
CCM	−0.3534	1.3176

effectiveness of our online EM updating strategies, we compare them to two extreme cases:

(i) **Static**, where a click model that has been trained once is kept unchanged, i.e., step 3 of the above protocol is not performed.

(ii) **EM with Retraining**, where a click model is re-trained from scratch every day, using historical and newly observed click-through data, i.e., in step 3 of the above protocol instead of updating the model, we retrain it using all data from days $\{1, \ldots, x+1\}$.

Metrics. We are interested in assessing two aspects of our proposed strategies for updating click models: efficiency and effectiveness. We assess efficiency by the time it takes to update click models. And we measure effectiveness of our click model updating strategies using standard metrics, namely log-likelihood and perplexity [3, 5, 7]. Statistical significance of observed differences is determined using a paired Student t-test at the 0.01 level.

When measuring perplexity and log-likelihood, we use all available data to train and test click models. In this case, training is performed on a MapReduce cluster. To measure execution time, we use a part of the dataset that can be processed on a single machine with 2.00GHz CPU and 64Gb RAM (because we cannot control the cluster load and distribution of our jobs). In particular, we use the first 100K query sessions of each day to measure the time it takes to train and update click models.

Click models. To assess the impact of online EM on click models, we need to select one or more click models that require EM for their inference. DBN [1], UBM [5], and CCM [7] are the standard click models of this kind [3]. Table 1 shows performance of the above models when trained on the first two weeks of the PWSC dataset and evaluated on the next 13 days.

We see from Table 1 that UBM significantly outperforms both DBN and CCM, a finding that is in line with other results reported on the Yandex relevance prediction challenge dataset [3, 6].[2] For this reason, we use the UBM model in our experiments.

5 RESULTS AND DISCUSSION

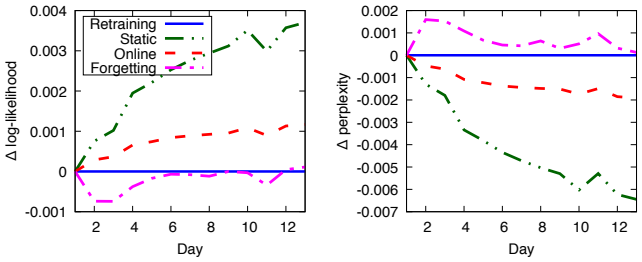
Efficiency. In Section 3 we show that Online EM and EM with Forgetting have lower computational complexity than EM with Retraining. Here, we show that the gains from the reduced complexity are also practically important. To do that, we measure the time it takes for Online EM, EM with Forgetting and EM with Retraining to update the trained UBM model once a day using days 15–27 of the PWSC dataset. (As explained in the previous section, we use the first 100K query sessions of each day.)

[2]See also https://academy.yandex.ru/events/data_analysis/relpred2011/

Table 2: Execution time in minutes of EM with Retraining, Online EM and EM with Forgetting when updating UBM on days 15–27 of the PWSC dataset. Time is measured on a subset of the click log (first 100K query sessions of each day).

Update strategy	Min (day 15)	Max (day 27)	Average	Total
EM with Retraining	28.1	49.6	39.7	516.1
Online EM	0.3	0.3	0.3	3.9
EM with Forgetting	0.3	0.3	0.3	3.9

Table 2 lists the execution times of EM with Retraining, Online EM, and EM with Forgetting. Online EM and EM with Forgetting are 130 times more efficient than EM with Retraining. This improvement can be explained by the fact that EM with Retraining does K passes over $n+m$ observations, while Online EM (and EM with Forgetting) does only 1 pass over m observations. Also, Online EM (and EM with Forgetting) requires constant time to incorporate m observations (18 seconds), while the time for EM with Retraining grows from 28 minutes (for day 15) to 50 minutes (for day 27).



(a) Log-likelihood (lower is better) (b) Perplexity (higher is better)

Figure 1: Difference in log-likelihood and perplexity between EM with Retraining (oracle) and other methods. EM with Forgetting uses $\eta = 0.001$.

Effectiveness of Online EM. Next, we compare the effectiveness of Online EM, which incorporates newly observed click information on the fly, to the Static approach, which ignores newly observed clicks, and to EM with Retraining, which trains click models from scratch every day using historical data and newly observed clicks. In particular, we aim to answer the following questions: (i) Does Online EM achieve higher effectiveness compared to the Static approach? (ii) Does Online EM achieve a comparable effectiveness to the one of EM with Retraining? To answer these questions, we measure the log-likelihood and perplexity of the UBM model after daily updates using days 15–27 of the PWSC dataset.

We consider EM with Retraining as an oracle and plot the performance of the Static and Online EM approaches with respect to this oracle (Fig. 1). Note that the perplexity deltas are negative, because lower values of perplexity indicate higher effectiveness. In addition, we report the average values of log-likelihood and perplexity in Table 3 (top half).

Fig. 1 shows that EM with Retraining, which trains click models from scratch every day, achieves the best effectiveness. Online EM significantly improves the quality of click models over the Static

Table 3: Average effectiveness of different update strategies. All update strategies are significantly better than the Static approach. $^{\triangle}/^{\triangledown}$ denote statistically significant differences with respect to EM with Retraining; $^{\blacktriangle}/^{\blacktriangledown}$ denote statistically significant differences with respect to Online EM.

Update strategy	Log-likelihood	Perplexity
EM with Retraining	−0.2180	1.2783
Static	$-0.2204^{\triangledown}$	$1.2825^{\triangledown}$
Online EM	$-0.2188^{\triangledown}$	$1.2796^{\triangledown}$
EM with Forgetting, $\eta = 0.001$	$-0.2178^{\blacktriangle}$	$1.2777^{\triangle\blacktriangle}$
EM with Forgetting, $\eta = 0.005$	$-0.2180^{\blacktriangle}$	$1.2775^{\triangle\blacktriangle}$
EM with Forgetting, $\eta = 0.01$	$-0.2183^{\triangledown\blacktriangle}$	$1.2775^{\triangle\blacktriangle}$

approach, because it utilizes additional information, i.e., newly observed clicks/skips, to update the model parameters. However, the effectiveness of Online EM does not reach the effectiveness of EM with Retraining, while being close to it (see Table 3).

Note that, as opposed to Online EM, EM with Retraining first collects a number of click/skip observations and then uses all available information to update the model parameters. This way it uses more information than Online EM and, thus, achieves higher effectiveness but at substantially higher computational costs. In particular, EM with Retraining cannot be used to continuously update click models in online settings.

Effectiveness of EM with Forgetting. Here, we evaluate the effectiveness of EM with Forgetting, which deals with outdated click information. We aim to answer the following question: Does EM with Forgetting, which discounts past click/skip observations, improve the effectiveness of Online EM, which does not do any discounting? To answer this question, we measure the log-likelihood and perplexity of UBM that is updated using EM with Forgetting during days 15–27 of the PWSC dataset.

We use forgetting rates η of 0.001, 0.005 and 0.01 in our experiments. Lower values of η did not result in any changes in effectiveness, while larger values did degrade the performance. The average absolute values of log-likelihood and perplexity are presented in Table 3 (bottom half), where all differences between EM with Forgetting and Online EM are statistically significant. The relative performance of EM with Forgetting ($\eta = 0.001$) compared to EM with Retraining is presented in Fig. 1.

First, we see that EM with Forgetting significantly improves the effectiveness of Online EM. This confirms our intuition that click information becomes outdated over time and past observations need to be discounted. Second, EM with Forgetting for $\eta = 0.001$ and $\eta = 0.005$ reaches the same effectiveness as EM with Retraining in terms of log-likelihood and significantly outperforms the latter in terms of perplexity for all forgetting rates. Hence, EM with Forgetting is at least as effective as EM with Retraining, while being orders of magnitude more efficient. This result suggests that in online scenarios EM with Forgetting should be preferred over both Online EM and EM with Retraining.

6 CONCLUSIONS

We have studied click models in an online scenario, where a search engine deals with a stream of click/skip observations. We have shown that the effectiveness of once trained click models degrades over time and, thus, these models must be constantly updated to keep up with the click stream.

We have proposed Online EM to efficiently update click models on the fly using readily available EM equations as well as the EM with Forgetting method to deal with outdated click information by discounting past observations depending on their age. We have shown experimentally that the proposed methods are orders of magnitude more efficient than complete retraining and, as opposed to the latter, always keep click models up to date. We have also shown that Online EM and EM with Forgetting have significantly higher effectiveness than the static approach with no updates.

With this work we aim to move click modeling research closer to more real-world settings. As to future work, we plan to study the effect of the forgetting rate on different click model parameters; and to develop methods for dealing with outdated click information for other inference methods (e.g., probit).

Acknowledgments. This research was supported by Ahold Delhaize, Amsterdam Data Science, the Bloomberg Research Grant program, the Criteo Faculty Research Award program, the Dutch national program COMMIT, Elsevier, the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement nr 312827 (VOX-Pol), the Microsoft Research Ph.D. program, the Netherlands Institute for Sound and Vision, the Netherlands Organisation for Scientific Research (NWO) under project nrs 612.001.116, HOR-11-10, CI-14-25, 652.002.001, 612.001.551, 652.001.003, and Yandex. All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

REFERENCES

[1] O. Chapelle and Y. Zhang. A dynamic Bayesian network click model for web search ranking. In *WWW*, pages 1–10, 2009.
[2] A. Chuklin, P. Serdyukov, and M. de Rijke. Click model-based information retrieval metrics. In *SIGIR*, pages 493–502, 2013.
[3] A. Chuklin, I. Markov, and M. de Rijke. *Click Models for Web Search.* Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan & Claypool, 2015.
[4] G. Dupret and C. Liao. A model to estimate intrinsic document relevance from the clickthrough logs of a web search engine. In *WSDM*, pages 181–190, 2010.
[5] G. E. Dupret and B. Piwowarski. A user browsing model to predict search engine click data from past observations. In *SIGIR*, pages 331–338, 2008.
[6] A. Grotov, A. Chuklin, I. Markov, L. Stout, F. Xumara, and M. de Rijke. A comparative study of click models for web search. In *CLEF*, pages 78–90. Springer, 2015.
[7] F. Guo, C. Liu, A. Kannan, T. Minka, M. Taylor, Y.-M. Wang, and C. Faloutsos. Click chain model in web search. In *WWW*, pages 11–20, 2009.
[8] P. Liang and D. Klein. Online EM for unsupervised models. In *NAACL*, pages 611–619, 2009.
[9] C. Liu, F. Guo, and C. Faloutsos. BBM: Bayesian browsing model from petabyte-scale data. In *KDD*, pages 537–546, 2009.
[10] C. Liu, F. Guo, and C. Faloutsos. Bayesian browsing model: Exact inference of document relevance from petabyte-scale data. *ACM Transactions on Knowledge Discovery from Data*, 4(4):Article 19, 2010.
[11] S. Malkevich, I. Markov, E. Michailova, and M. de Rijke. Evaluating and analyzing click simulation in web search. In *ICTIR*, 2017.
[12] I. Markov, E. Kharitonov, V. Nikulin, P. Serdyukov, M. de Rijke, and F. Crestani. Vertical-aware click model-based effectiveness metrics. In *CIKM*, pages 1867–1870, 2014.
[13] R. M. Neal and G. E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in Graphical Models*, pages 355–368. MIT Press, 1999.
[14] K. Radinsky, K. M. Svore, S. T. Dumais, M. Shokouhi, J. Teevan, A. Bocharov, and E. Horvitz. Behavioral dynamics on the web: Learning, modeling, and prediction. *ACM Transactons on Information Systems*, 31(3):Article 16, 2013.
[15] B. Thiesson, C. Meek, and D. Heckerman. Accelerating EM for large databases. *Machine Learning*, 45:279–299, 2001.
[16] Q. Xing, Y. Liu, J.-Y. Nie, M. Zhang, S. Ma, and K. Zhang. Incorporating user preferences into click models. In *CIKM*, pages 1301–1310, 2013.