Knowledge and Information Systems (2022) 64:2833–2847
https://doi.org/10.1007/s10115-022-01726-0

REGULAR PAPER

Hybrid online–offline learning to rank using simulated annealing strategy based on dependent click model

Osman Ali Sadek Ibrahim[1] · **Eman M. G. Younis**[2]

Received: 11 April 2022 / Revised: 12 May 2022 / Accepted: 16 July 2022 /
Published online: 8 August 2022

Abstract
Learning to rank (LTR) is the process of constructing a model for ranking documents or objects. It is useful for many applications such as Information retrieval (IR) and recommendation systems. This paper introduces a comparison between Offline and Online (LTR) for IR. It also proposes a novel Offline (1 + 1)-Simulated Annealing Strategy (SAS-Rank) and introduces the first Hybrid Online–Offline LTR techniques using SAS-Rank and ES-Rank with Online Dependent Click Model (DCM). SAS-Rank is a combination of Simulated Annealing method and Evolutionary Strategy. From the obtained experimental results, we can conclude that the Offline LTR techniques outperformed the well-known Online Dependent Click Model (DCM) technique. Moreover, the Hybrid Online–Offline SAS-Click outperformed the predictive ranking results on unseen data in most evaluation fitness metrics using LETOR 4 dataset compared to other approaches. On the other hand, Hybrid ES-Click is a competitive approach with SAS-Click in evolving ranking models for training and validation data. Regarding Offline LTR, the SAS-Rank outperformed the well-known ES-Rank which has been compared in previous studies with fourteen machine learning techniques. This research uses the best available Linear LTR approaches existing in the literature which are offline ES-Rank with Online DCM. The linear LTR approach output is a linear ranking model which can be represented as a vector of feature importance weights. This paper demonstrated the results and findings obtained using the LETOR 4 dataset, and Java Archive Package is provided for facilitating reproducible research.

Keywords Learning to rank · Simulated annealing strategy · Dependent click model · Hybrid online–offline learning

✉ Osman Ali Sadek Ibrahim
osman.ibrahim@mu.edu.eg; osmaneg200@gmail.com

Eman M. G. Younis
eman.younas@mu.edu.eg

[1] Computer Science Department, Minia University, Minya, Egypt

[2] Information Systems Department, Minia University, Minya, Egypt

1 Introduction

Click-through data from search engine logs is a great source of feedback from users. Click-through data have been offered as a replacement resource for relevance labels in a number of studies, including [1–3]. This is because the expenses of creating an IR dataset using the Click-through data paradigm are lower than using the Pooling and Cranfield paradigms. Click-through data, on the other hand, is subject to a number of biasing criteria as a result of user decisions. There are some factors that may affect it such as the following: (1) The user's educational level, intelligence, and familiarity with the IR system, all of which influence the decision of user satisfaction and IR system performance [4]. This leads to click bias. (2) The position of the document in the retrieved ranked list [2, 5, 6], user clicks are known to be biased, noisy, and difficult to interpret. As a result, research has been conducted to simulate actual Click-through data that is free of noise and bias. This research proposes several Click models for removing noise from Click-through data based on various assumptions. In online LTR research, these Click models were used. The click ranking models are created using the relevance labels of the benchmarks to simulate actual user clicks. There are two types of performance indicators available for online Click models. Offline assessment measures are the first category, while online evaluation metrics are the second. MAP, NDCG, P@K, RR@K, ERR@K with/without A/B testing, and Interleave [6–8] are examples of offline and online evaluation metrics. According to Kharitonov [6, 7], in online LTR, the Probabilistic Interleave is more effective than A/B test in representing user interaction within the IR system. The offline and online evaluation measures are based on the training set's relevance labels. Due to the constraints of online evaluation measures, offline evaluation metrics are the most trustworthy metrics for comparing Computational Intelligence (CI) and Machine Learning (ML) techniques [4, 6]. In the Hybrid Online–Offline method, this study shows how ES-Rank and SAS-Rank can be used to optimize online LTR click models utilizing five evaluation metrics (MAP, NDCG, P@10, RR@10, and ERR@10). Firstly, an online LTR Dependent Click Model (DCM) is utilized to initialize ES-Rank and SAS-Rank algorithms. The Probabilistic Interleave procedure is used to alter the query-document pairs list using DCM model to produce the online ranking model. Then, it was used as initial chromosome vector for (1 + 1)-Evolutionary Strategy and (1 + 1)-Simulated Annealing Strategy.

The main contributions of this paper are as follows:

- Introducing the first Hybrid Online–Offline Learning to Rank approach in Information Retrieval (IR).
- Proposing a novel SAS-Rank technique for improving LTR methods.
- Comparing among Offline, Online and Hybrid Online–Offline LTR methods using five fitness evaluation metrics.

The previous research studies in this field motivated us toward examining the exploration procedure by using a Hybrid Online–Offline approach and a novel combination of evolution strategy with metaheuristic techniques. The exploration procedure can help in searching a wider range of learning solutions in the solution search space attempting to improve ranking accuracy. Moreover, the Online LTR approaches use the implicit relevance labels (implicit ground truth label) rather than actual user click interaction which is difficult to maintain for checking the quality of the evolved ranking model during executing the ranking algorithm. Thus, this research is the first experimental research study in the LTR field. It provides a comparison between using explicit and implicit relevance labels for generating a better ranking model based on Computational Intelligence methods and the first hybrid Online–Offline LTR approach.

The rest of the paper is organized as follows. Section 2 presents a brief literature review of related work in this research subject. The proposed LTR method is described in Sect. 3. Experimental results are presented in Sect. 4, while conclusions are given in Sect. 5.

2 Related work

A typical problem in IR research is ranking based on the relevance of the retrieved documents in response to the user's query. Unsupervised Term Vector Model (TVM), such as Vector Space Model (VSM) based on Term Frequency-Inverse Document Frequency (TF-IDF) or Okapi-BM25, and language models were utilized in early IR research [9]. These models were used to rate the documents that were retrieved based on their relevancy to the user's queries. Using only one scoring method (TWS) in IR systems. These methods proved to be insufficient for effective IR systems. The reason for this is that scoring approaches such as Okapi-BM25 and various language models are limited in terms of returning appropriate search results in terms of relevance judgment [10–12]. This emphasizes the importance of having multiple TWS methods to rank pages in relation to user queries. In addition, the ranking of documents should take into account the importance of the documents on the web and the host server, among other desirable aspects. Tao Qin et al. [13] presented a new research trend for document ranking by creating LETOR datasets. These benchmarks were distilled from search engines and the well-known TREC conference collections. As part of the benchmark features, these benchmarks include more than one term-weighting scheme (scoring approaches). They also have some other characteristics that reflect how important the documents are on the world wide web. For Learning to Rank (LTR) research challenges, the documents in these datasets were mapped into completely assessed query-document pairs. In LTR research, Evolutionary and Machine Learning techniques are used to adjust and improve the scoring methods and other desirable features for these datasets, resulting in efficient ranking models for successful IR systems.

There are three types of LTR methods, according to [2] which are the pointwise method, the pairwise method, and the listwise method. This classification is based on the measurement of loss function or fitness function. Each individual item (query-document pair) is viewed as a learning instance in the pointwise approach. Linear Regression (LR) [14], Boosting [15], Gradient Boosted Regression Trees (GBRT or MART) [16, 17], and Random Forest (RF) [18] are examples of pointwise techniques. The learning instance in the pairwise technique is a pair of items (two query-document pairs for the same question). RankNET (Rank Neural Net) [19], RankBoost, and SVMRank (Rank Support Vector Machine) [5] are examples of a pairwise technique. The learning instance in the listwise approach is the full retrieved list of items (the list of query-document pairs for each query). ListNET (Listwise Neural Net) [20], RankGP [21, 22], Coordinate Ascent [23], AdaRank [24], and RankGPES [25] are examples of listwise techniques. Because listwise approaches have been proved to perform better than pointwise and pairwise approaches [20], the suggested SAS-Rank technique detailed further in this study is based on the listwise approach.

Researchers have proposed hybrid approaches that combine methods from the three LTR categories in order to improve the performance of LTR approaches. Sculley [26] proposed a method (CoRR) that merged LR (pointwise technique) and SVM (support vector machine) (pairwise approach). This method was implemented in the Sofia-ml package by [26] and takes a fair amount of time to execute. Its performance in terms of NDCG and MAP, on the other hand, is restricted. Mohan et al. [17] suggested a hybrid machine learning strategy for

initializing GBRT using RF in order to get better Normalized Discounted Cumulative Gain (NDCG). Experiments revealed, however, that this approach consumes too much computation time when compared to other approaches in the literature [5, 27]. LambdaRank and LambdaMART, on the other hand, are two proposed hybrid techniques that integrate pairwise and listwise procedures [28]. The boosted tree from LambdaRank, which is based on RankNET, is LambdaMART. Related to the Yahoo! Learning to Rank Challenge [29], both LambdaMART and LambdaRank performed better than the method proposed by Mohan et al. Overall, all of these techniques suffer from high computational runtime or shortcomings in the prediction results evaluation accuracy. For example, in the MSLR-WEB10K fold, the computational training time of IGBRT or Coordinate Ascent in the Ranklib package, among other approaches, is more than 9 h [30]. For the LETOR 3 datasets [2], the RankGPES approach suggested by Islam [25], which merged Genetic Programming with Evolutionary Strategy, required significant computational time. On the Ohsumed dataset fold, the approach cost from 30 to 50 min of training time, and 1–2 h for each .Gov dataset fold. ES-Rank took less time to train than RankGPES. ES-Rank takes 83–177 s to train on each .Gov dataset fold, but it takes 30 s to train using the Ohsumed dataset fold. Moreover, LETOR 3 is smaller than MSLR-WEB10K, and other machine learning methods took less time than RankGPES. The previous methods were proposed for offline LTR using FVM. Katja Hofmann et al. [3, 31] proposed a new research trend in LTR based on the simulation of user click models. This type of research is known as online LTR. Recently, Jianxun Lian et. al. [32] proposed a Deep Learning method to combine implicit and explicit feature interaction in recommendation systems. Whereas, Fatih Cakir et. al. [33] proposed a Deep Learning method to do an intensive learning and evaluated it using Average Precision metric. However, Deep Learning methods among other approaches except LR, SVM-Rank and ES-Rank LTR approaches do not accept linear ranking models as their initialization values in the learning process. Moreover, ES-Rank proved its capability to outperform LR and SVM-Rank as demonstrated in the previous research [30]. Thus, we choose ES-Rank in comparison with a novel approach SAS-Rank for the first Hybrid Online–Offline LTR approaches that are introduced here.

3 Materials and methods

3.1 Dependent click model

According to the Cascade Click Model (CCM), a user scans the sorted retrieved document list from top to bottom until she or he finds one appropriate document and clicks it [34]. The user may, however, be required to look through more than one document from the retrieved list until he/she is satisfied. As a result, Guo et al. [35] presented a more sophisticated click model as a CCM extension. The Dependent Click Model (DCM) is the name of this model. The user is satisfied with the browsed relevant clicked documents from the ranked retrieved list when the DCM is similar to CCM with several clicks. In its calculations, the DCM takes into account the position of the clicked documents (instances), the average of user clicks, and the document examination characteristics. Assume that μ_i is a location-dependent parameter that represents the likelihood that the user will need to examine more documents after clicking on position i.

If the user does not click the document, the probability of the following document being examined is one. The μ parameters are a set of behavior parameters that a user can utilize across numerous query sessions. The following recursive method ($1 \leq i \leq M$, where M

is the number of documents in the search result) can be used to demonstrate the Click and Examination probability in DCM:

$$\begin{aligned} E_{d_{1,1}} &= 1, \\ C_{d_{i,i}} &= E_{d_{i,i}} \cdot r_{d_i}, \\ E_{d_{i+1,i+1}} &= \mu_i \cdot C_{d_{i,i}} + (E_{d_{i,i}} - C_{d_{i,i}}). \end{aligned} \tag{1}$$

where $E_{d_{i,i}}$ represents the likelihood of examining document d_i, while $C_{d_{i,i}}$ represents the probability of clicking on document d_i, r_{d_i} represents the relevance degree of document d_i, and μ_i represents the user behavior parameter. The DCM model is explained in the following way. After the user examines the document in the search result, its relevancy determines its click likelihood. The probability for the following document examination is therefore calculated by μ_i after clicking on a document. The examination and click formula for document $d_{i,i}$ is as follows, based on equation 1:

$$\begin{aligned} E_{d_{i,i}} &= \prod_{j=1}^{i-1} (1 - r_{d_j} + \mu_j \cdot r_{d_j}), \\ C_{d_{i,i}} &= r_{d_i} \cdot \prod_{j=1}^{i-1} (1 - r_{d_j} + \mu_j \cdot r_{d_j}). \end{aligned} \tag{2}$$

If the click event use $\{C_1, C_2, \ldots, C_M\}$ in query session, the M stands for the number of documents in the search result (ranked retrieved list) and the r_{d_i} stands for the degree of document relevance of document d_i. The log-likelihood for DCM clicks in each query session is provided by [35]:

$$\begin{aligned} L_{\mathrm{DCM}} = & \sum_{i=1}^{l-1} \Big(C_i * (\log(r_{d_i}) + \log(\mu_i)) \\ & + (1 - C_i) * \log(1 - r_{d_i}) \Big) \\ & + C_l * \log(r_{d_l}) \\ & + (1 + C_l) * \log(1 - r_{d_l}) \\ & + \log\Big(1 - \mu_l + \mu_l * \prod_{j=l+1}^{n} (1 - r_{d_j})\Big), \end{aligned} \tag{3}$$

$$\begin{aligned} L_{\mathrm{DCM}} \geq & \sum_{i=1}^{l} C_i * \log(r_{d_i}) + (1 + C_i) * \log(1 - r_{d_i}) \\ & + \sum_{i=1}^{l-1} C_i * \log(\mu_i) + \log(1 - \mu_l) \end{aligned} \tag{4}$$

If the query session has no clicks, then L_{DCM} is a special case with $l = M$, $C_l = \mu_l = 0$. A learning approach is carried for DCM learning by maximizing the lower bound of L_{DCM}

in Eq. 4. The user behavior parameter is estimated by:

$$\mu_i = 1 - \frac{\text{No. of query sessions when last clicked position is } i}{\text{No. of query sessions when position } i \text{ is clicked}} \tag{5}$$

Equation 5 specifies the empirical probability of location i not being a non-last-clicked position for all query sessions in the training set for $1 \leq i \leq M-1$. Finally, the DCM model sampling technique for examining variables E_i and click variables C_i, while users review the search result list one by one from top to bottom, is as follows:

$$E_1 \leftarrow 1, E_i == 0 C_i \leftarrow 0, E_{i+1} \leftarrow 0, C_i \approx Bernoulli(r_{d_i}), E_{i+1} \approx Bernoulli(1 - C_i + \mu_i * C_i)$$

$Bernoulli()$ is the Bernoulli probabilistic distribution [36]. For the most state-of-the-art click models, Aleksandr et al. surveyed their details in [37].

3.2 The proposed approaches

The proposed approach, Simulated Annealing Strategy Ranking, is presented in this section (SAS-Rank). In SAS-Rank, the chromosomes are initialized by assigning a 0.5 value to each gene (each query-document feature weight), which signifies the equally likely importance of the associated feature to the gene in each case. A further initialization is performed which is SAS-Rank with chromosomal initialization from the DCM ranking model and it is called SAS-Click. The capacity of Simulated Annealing (SA) to converge the initial solution to a better-evolved solution in terms of computational runtime is why we chose it for our novel approach [38, 39]. Furthermore, the relevance of exploration for various proposed solutions based on the Hybrid solution in Meta-heuristic and Computational Intelligence approaches [40, 41] initial prompts the need for improved SAS-Rank initialization procedures with a hybrid online–offline procedure. Thus, this paper compares the performance of SAS-Rank with ES-Rank and the most well-known Online LTR (DCM) approach, while initializing the ranking model chromosomes with normal initialization values in offline LTR and a hybrid online–offline LTR methodology. The usage of online LTR model feature weights as an initial solution for the evolved chromosome will allow SAS-Rank to be explored for a better solution throughout a broad range of search space. Furthermore, earlier research has shown that the listwise technique outperforms the pairwise, and pointwise approaches in terms of Mean Average Precision (MAP), and Normalized Discounted Cumulative Gain (NDCG) [20]. This motivated us to propose the listwise based method which is called SAS-Rank.

The suggested SAS-Rank algorithm is shown in 1. This method is a (1+1)-Simulated Annealing Strategy, which evolves a single vector using cooling annealing steps across a number of generations. The training set of query-document pairings or feature vectors is used as the input, and the result is a linear ranking function. The chromosome *ParentCh* is a vector of M genes, with each gene represents the importance of the related feature for document ranking. Steps 1 to 4 initialize the chromosomal vector by assigning a 0.5 value to each gene. In Step 5, the Boolean variable *Good*, which indicates whether the previous generation's mutation process should be repeated. It is set to default value as FALSE for not repeated, and changed during SAS-Rank when achieving a good mutation steps for evolving better solution. In step 6, a copy of *ParentCh* is turned into *OffspringCh*. In this paper, the evolution process for $MaxGenerations = 1300$ generations which begins in Step 7, and finishes in Step 24. Steps 8 to 16 demonstrate the technique for controlling the mutation process by selecting the number of genes to mutate (R), and the mutation step.

Algorithm 1: SAS-Rank: (1+1)-Simulated Annealing Strategy Ranking Approach

Input : A training set $\phi(q, d)$ of query-document pairs of feature vectors. Weight Feature Vector $WLR = g(wlr_i)$ from Linear Online LTR Models on $\phi(q, d)$ set.
Output: A linear ranking function $F(q, d)$ that assigns a weight to every query-document pair indicating its relevancy degree.

1 Initialization
2 **for** $(Gen_i \in ParentCh)$ **do**
3 | $Gen_i = 0.5$ or weight from Online LTR model;
4 **end**
5 Good=FALSE;
6 $OffspringCh = ParentCh$;
7 **for** $G = 1$ *to* $MaxGenerations$ **do**
8 | **if** *(Good==TRUE)* **then**
9 | | Use the same mutation process of generation $(G - 1)$ on $OffspringCh$ to mutate $OffspringCh$, that is, mutate the same R genes using the same $MutationStep$;
10 | **else**
11 | | Choose R at random, the number of genes to mutate;
12 | | **for** $j = 1$ *to* R **do**
13 | | | Choose at random, Gen_i in $OffSpringCh$ for mutation;
14 | | | Mutate $Gene_i$ using $MutationStep$ according to equation (M1)
15 | | **end**
16 | **end**
17 | **if** *(Fitness(ParentCh,*$\phi(q, d)$*) <Fitness(OffspringCh,*$\phi(q, d)$*))* **then**
18 | | $ParentCh = OffspringCh$;
19 | | Good=TRUE;
20 | | Restart Temperature;
21 | **else**
22 | | $OffspringCh = ParentCh$;
23 | | Good=FALSE ;
24 | **end**
25 | OR IF(cooling temperature is reached);
26 | $ParentCh = OffspringCh$;
27 | Restart Temperature;
28 **end**
29 **return** the linear ranking function $F(q, d) = ParentCh^T \bullet \phi(q, d) = W^T \bullet \phi(q, d)$, that is $ParentCh$ at the end of the $MaxGenerations$ contains the evolved vector W of M feature weights, T indicates the transpose

The mutation step is determined using equation (m1), where *ZGaussian(0,1):* is a ziggurat random number with 0 mean, and 1 standard deviation [42], and *Fitness(ParentCh), and Fitness(OffspringCh):* are the fitness values for the parent, and offspring chromosome with the training dataset consequently.

$$Mutated\_Gene\_i = Gene\_i + Gaussian(0, 1) * \frac{Gene\_i}{(Fitness(ParentCh) - Fitness(OffspringCh))} \quad (m1)$$

As illustrated in Step 9, a successful mutation process (producing better offspring) in generation $(G - 1)$ is reproduced in generation G. Otherwise, as indicated in Steps 11 to 15, the parameters of the mutation process are reset. Steps 17 to 23 choose between the *ParentCh*, and the *OffspringCh* based on their MAP or NDCG@10 or P@10 or RR@10 fitness scores. Steps 25 and 26 indicate the cooling annealing condition for accepting bad solution when the cooling temperature is reached. Finally, in Step 28, SAS-Rank returns the ranking function,

Table 1 The properties of the LTR dataset folds used in the experimental study

Dataset	Queries	Query-document pairs	Features
MQ2007	1692	69623	46
MQ2008	784	15211	46

which is determined by the transpose of the evolved vector of feature weights as well as the query-document pairs. The initial step in using the suggested LTR technique is to get the datasets that comprise the training, validation, and test benchmarks. Then, the suggested SAS-Rank algorithm is used to evolve a linear ranking function from the training data. The performance of the evolved linear ranking function is then evaluated using the test set to determine the learning algorithm's predicted performance. The proposed method is the first in the LTR domain, and AI research field by combining Simulated Annealing with Evolutionary Strategy, and also this paper is the first paper which proposes a hybrid online–offline LTR approach. The experimental analysis of these approaches as a comparison is shown in the next section. This algorithm has a computational complexity of $\Omega(N * n * \log(M))$, where N represents the number of training query-document pairs, n represents the number of evolving iterations, and R represents the number of genes in the chromosome.

4 Experimental results

4.1 Datasets description

LETOR 4 (MQ2007, and MQ2008) datasets were utilized in the experimental study [13, 43, 44]. The characteristics of these datasets are summarized in Table 1. The number of features in MQ2007 or MQ2008 is 46, whereas MQ2007, and MQ2008, on the other hand, include 1692, and 784 unique queries, respectively. Both datasets have the same levels of relevance labels, which are 0, 1, and 2. The relevance label 0 indicates that the query is irrelevant to the document in the query-document combination, whereas the relevance labels 1 and 2 show the degree of relevance for the query, ranging from slightly relevant to completely relevant to the document.

4.2 Fitness and evaluation metrics

In this study, *MAP*, *NDCG@10*, *P@10*, *RR@10*, and *ERR@10* were utilized [5, 45, 46] as five separate fitness functions on the training sets. They also were utilized as the evaluation metrics for the ranking functions on the test sets.

We define the five metrics that were used in this research. Let $d1, d2, \ldots, d_D$ represent the sorted documents in decreasing order of their similarity measure function value, where D is the number of documents retrieved. The function $r(di)$ provides the document d_i's relevance value. If d_i is relevant, it yields 1; else, it returns 0. The precision of the top-k relevant query-documents returned per query q $(P@k)$ is as follows:

$$P@k = \frac{\sum_{i=1}^{k} r(d_i)}{k} \tag{6}$$

The mean of the precision $P@k$ values across all queries is the MAP for a set of queries. The following equation can be used to determine this:

$$MAP = \frac{\sum_{q=1}^{Q} P@k(q)}{Q} \tag{7}$$

Q denotes the number of queries. The following equation can be used to determine the Normalized Discounted Cumulative Gain of the top-k documents retrieved (NDCG@k):

$$NDCG@k = \frac{1}{IDCG@k} \cdot \sum_{i=1}^{k} \frac{2^{r(d_i)} - 1}{\log_2(i+1)} \tag{8}$$

The following equation is used to compute the *Discounted Cumulative Gain of top-k documents retrieved (DCG@k)*:

$$DCG@k = \sum_{i=1}^{k} \frac{2^{r(d_i)} - 1}{\log_2(i+1)} \tag{9}$$

where $IDCG@k$ is the ideal (maximum) discounted cumulative gain of the top-k documents retrieved, and $r(di)$ yields 1 if the document at position i is relevant, and 0 otherwise. If all of the top-k documents are relevant, the $DCG@k$ equals the $IDCG@k$.

The Reciprocal Rank metric (RR). If the first relevant document in position i, then the reciprocal rank score can be calculated by this equation:

$$Reciprocal\ Rank\ Score = \frac{1}{i} \tag{10}$$

The Reciprocal Rank at top-K relevant retrieved query-document pairs (RR@K) is as follows:

$$RR@K = \sum_{i=1}^{K} \frac{1}{i} \tag{11}$$

while the Expected Reciprocal Rank at top-N retrieved query-document pairs (ERR@N) is as follows:

$$ERR@N = \sum_{i=1}^{N} \frac{r(d_i)}{i} \tag{12}$$

4.3 Experimental results and discussion

In this section, the summary of the results obtained when applying this approach using the LETOR 4 dataset benchmarks were presented. Loret, and then ES-Rank [30, 47] or SAS-Rank library packages were used for the implementation of these experiments. The Loret package is an online LTR package that is used for producing ranking models for the LTR datasets. The experimental settings of ES-Rank, SAS-Rank, and Loret are the default of these

Table 2 Average results of offline, online and hybrid proposed methods using 5 fitness functions on MQ2008 data

MQ2008 Methods Vs Dataset		MAP	NDCG@10	RR@10	ERR@10	P@10
Test	DCM	0.2982	0.3248	0.3290	0.0536	0.2068
	ES-Click	0.4592	0.4982	0.5258	0.0931	**0.2682**
	SAS-Click	**0.4635**	**0.5386**	0.4769	**0.0990**	0.2630
	ES-Rank	0.4427	0.4664	0.4931	0.0881	0.2552
	SAS-Rank	0.4614	0.5304	0.5016	0.0964	0.2636
Training	DCM	0.3094	0.3495	0.3445	0.0578	0.2113
	ES-Click	0.4729	0.5045	0.5341	0.0962	0.2705
	SAS-Click	0.4828	0.5565	0.5448	**0.1003**	**0.2795**
	ES-Rank	0.3752	0.5083	0.5376	0.3366	0.1895
	SAS-Rank	**0.4736**	**0.5573**	**0.5454**	0.1000	0.2789
Validation	DCM	0.3041	0.3331	0.3385	0.0565	0.2090
	ES-Click	**0.4793**	0.5142	0.5582	0.0980	0.2740
	SAS-Click	0.4635	**0.6085**	**0.5912**	0.0971	**0.2774**
	ES-Rank	0.4097	0.5145	0.5319	**0.1160**	0.1904
	SAS-Rank	0.4736	0.6067	0.5879	0.0966	0.2768

Bold values indicate highest effectiveness in the corresponding fitness evaluation metric

packages. The number of evolving iterations is 1300 iteration in ES-Rank and SAS-Rank, while configuration setting in Loret is the default one provided with the package. The ES-Rank, and SAS-Rank (Offline LTR) take the ranking model produced by Loret package using DCM model (Online LTR) as initial parent chromosome. Then, it evolves a better ranking model for each training dataset using various evaluation fitness metrics. The p-value of the null hypothesis of F-test is less than 0.05 for these results. This indicates the high degree of confidence in the result improvements for the various experimental settings between ES-Click, SAS-Click, and DCM.

This paper demonstrates the behavior of SAS-Rank, and ES-Rank improvements for predictive, and evolving LTR results. This is according to evolve the ranking using training, and validation datasets, and then using the evolved ranking model as a predictive ranking model on unseen test dataset. This also can be another objective for checking the behavior of evolutionary computation methods not only in evolving data benchmarks, but also on unseen data to simulate machine learning methods. This paper shows the capability of Hybrid offline–online LTR approach with the recent well-known LETOR 4 data. It also examined its behavior as offline approach compared with the ES-Rank [30] which was compared with other recent approaches for offline LTR.

From Tables 2 and 3, the ES-Rank, and SAS-Rank can improve the DCM ranking model when applied on the training, validation, and test data of well-known LETOR dataset benchmarks. We used five fitness evaluation metrics to show the optimization capability of the hybrid Online–Offline LTR approaches, and compared between them separately. The MAP, NDCG@10, P@10, RR@10, and ERR@10 are the evaluation fitness metrics used in this experiment. The green color in the tables indicates the best values obtained by the methods used in this paper. For example, the predictive MAP value of the hybrid SAS-Click on test, and validated MQ2008 data outperformed online DCM, and ES-Click, and also outperformed

Table 3 Average results of offline, online and hybrid proposed methods using 5 fitness functions on MQ2007 data

MQ2007 Methods Vs Dataset		MAP	NDCG@10	RR@10	ERR@10	P@10
Test	DCM	0.3601	0.3000	0.4193	0.0715	0.2988
	ES-Click	0.4859	0.4706	0.5627	0.1079	0.3970
	SAS-Click	**0.4903**	0.5258	0.5681	0.1101	0.3941
	ES-Rank	0.4868	0.4546	**0.5809**	0.1085	**0.4018**
	SAS-Rank	0.4831	0.5220	0.5675	**0.1109**	0.3854
Training	DCM	0.3326	0.2771	0.3925	0.0628	0.2743
	ES-Click	0.4548	0.4344	0.5566	0.0974	**0.3765**
	SAS-Click	**0.4549**	0.4892	**0.5655**	0.1005	0.3751
	ES-Rank	0.4527	0.4270	0.5411	**0.0997**	0.3738
	SAS-Rank	0.4517	**0.4917**	0.5589	0.0996	0.3721
Validation	DCM	0.3499	0.2965	0.3946	0.0676	0.2861
	ES-Click	**0.4855**	0.4569	**0.5960**	0.1024	**0.3885**
	SAS-Click	0.4673	0.5081	0.5839	0.1041	0.3782
	ES-Rank	0.4791	0.4603	0.5912	**0.1070**	0.3817
	SAS-Rank	0.4638	**0.5096**	0.5771	0.1035	0.3758

Bold values indicate highest effectiveness in the corresponding fitness evaluation metric

offline SAS-Rank, and ES-Rank. On the other hand, the offline SAS-Rank outperformed other methods for evolving the ranking model based on training MQ2008 data. In MQ2007, SAS-Click outperformed other methods on test, and training data, while ES-Click outperformed other approaches on validation data. Generally from the tables, SAS-Rank as offline, and Hybrid online–offline methods outperformed other methods in predictive ranking models on training, validation, and unseen test datasets based on the five fitness evaluation metrics.

From another perspective, the offline LTR methods (ES-Rank, and SAS-Rank) outperformed the well-known online DCM method. The reason is that the Offline LTR uses the explicit relevance labels for evaluating the quality of the evolved ranking model in each evolving iteration. On the contrary, online LTR uses implicit relevance labels by simulating Click-through labels for evaluating the quality of the learning ranking model in each learning iteration. This issue causes a drawback for learning an accurate ranking model in Online LTR approaches. Moreover, when comparing between Offline methods, we found that SAS-Rank outperformed ES-Rank in predictive on unseen data, and evolving in training, and validation dataset. This is due to the capability of Simulated Annealing strategy of evolving better solution based on adapting mutation step size using mutation rate which is relying on fitness evaluation metric behavior.

5 Conclusions

To sum up, the Offline SAS-Rank, and ES-Rank proved their capability to improve the performance of Online DCM ranking models using five evaluation fitness metrics. These approaches are called Hybrid Online–Offline LTR. It started with learning Online DCM ranking models from benchmarks, and then the SAS-Rank, and ES-Rank used the DCM ranking models to

evolve better ranking models. In general, the SAS-Click ranking model outperformed other methods in most predictive cases using MQ2008, and MQ2007, while ES-Click is the best for MQ2007 validation data. Moreover, Offline LTR techniques outperformed Online LTR approaches using five fitness evaluation measures. The reason for that is the capability of Offline learning methods when explicit relevance labels are available. Whereas, Online LTR depends on implicit relevance labels. The Loret, ES-Rank, and SAS-Rank packages have been used in this paper for producing DCM ranking models, and improving it using SAS-Rank, and ES-Rank in Hybrid Offline–Online LTR approaches. LETOR 4 datasets were used in the experiments that have been used to prove the usefulness of SAS-Rank, and ES-Rank to optimize the Online DCM models. Thus, Combining Simulated Annealing with Evolutionary Strategy can improve the online LTR models as it proved its capability to improve the offline LTR models. In addition, the LTR in Offline techniques from explicit relevance labels outperformed the Online LTR from implicit relevance labels.

Acknowledgements The authors would like to thank Minia University, Egypt and STDF for their financial support for Open Access publishing in Springer Journals to pursue the research work satisfactorily and make it freely available for everyone.

Data Availability Statement The data are public data in the Microsoft Research website by Microsoft research team.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Joachims T (2002) Optimizing search engines using clickthrough data. In: Proceedings of the eighth ACM SIGKDD international conference on knowledge discovery and data mining, KDD '02, pp 133–142, New York, NY, USA, 2002. ACM. ISBN 1-58113-567-X. https://doi.org/10.1145/775047.775067
2. Liu T-Y (2009) Learning to rank for information retrieval. Found Trends Inf Retr 3(3):225–331
3. Schuth A, Hofmann K, Whiteson S, de Rijke M (2013) LEROT: an online learning to rank framework. In: Proceedings of the 2013 workshop on Living labs for information retrieval evaluation CIKM 2013, San Francisco, California, USA, November 1, 2013, pp 23–26. https://doi.org/10.1145/2513150.2513162
4. Al-Maskari A, Sanderson M (2011) The effect of user characteristics on search effectiveness in information retrieval. Inf Process Manag 47(5):719–729. https://doi.org/10.1016/j.ipm.2011.03.002
5. Li H (2014) Learning to rank for information retrieval and natural language processing, 2nd edn. Morgan and Claypool Publishers, Williston
6. Kharitonov E (2016) Using interaction data for improving the offline and online evaluation of search engines. PhD thesis, University of Glasgow
7. Hofmann K (2013) Fast and reliable online learning to rank for information retrieval. SIGIR Forum 47(2):140. https://doi.org/10.1145/2568388.2568413
8. Schuth A (2016) Search engines that learn from their users. PhD thesis, Informatics Institute, University of Amsterdam, http://www.anneschuth.nl/thesis
9. Manning CD, Raghavan P, Schütze H (2008) Introduction to information retrieval. Cambridge University Press, New York
10. Ibrahim OAS, Landa-Silva D (2016) Term frequency with average term occurrences for textual information retrieval. Soft Comput 20(8):3045–3061. https://doi.org/10.1007/s00500-015-1935-7
11. Tonon A, Demartini G, Cudré-Mauroux P (2015) Pooling-based continuous evaluation of information retrieval systems. Inf Retr J 18(5):445–472. https://doi.org/10.1007/s10791-015-9266-y
12. Urbano J (2016) Test collection reliability: a study of bias and robustness to statistical assumptions via stochastic simulation. Inf Retr J 19(3):313–350. https://doi.org/10.1007/s10791-015-9274-y
13. Qin T, Liu TY, Xu J, Li H (2010) LETOR: a benchmark collection for research on learning to rank for information retrieval. Inf Retr 13(4):346–374. https://doi.org/10.1007/s10791-009-9123-y

14. Yan X, Su X (2009) Linear regression analysis: theory and computing. World Scientific Publishing Co., Inc, River Edge
15. Freund Y, Iyer R, Schapire RE, Singer Y (2003) An efficient boosting algorithm for combining preferences. J Mach Learn Res 4:933–969
16. Friedman JH (2001) Greedy function approximation: a gradient boosting machine. Ann Stat 29(5):1189–1232
17. Mohan A, Chen Z, Weinberger KQ (2011) Web-search ranking with initialized gradient boosted regression trees. J Mach Learn Res Workshop Conf Proc 14:77–89
18. Breiman L (2001) Random forests. Mach Learn 45(1):5–32. https://doi.org/10.1023/A:1010933404324
19. Burges C, Shaked T, Renshaw E, Lazier A, Deeds M, Hamilton N, Hullender G (2005) Learning to rank using gradient descent. In: Proceedings of the 22nd international conference on machine learning, ICML '05, pp 89–96, New York, NY, USA, ACM. https://doi.org/10.1145/1102351.1102363
20. Cao Z, Qin T, Liu T-Y, Tsai M-F, Li H (2007) Learning to rank: from pairwise approach to listwise approach. In: Proceedings of the 24th international conference on Machine learning, ICML '07, pp 129–136, New York, NY, USA, ACM. https://doi.org/10.1145/1273496.1273513
21. Lin, JY, Yeh, JY, Liu CC (2012) Learning to rank for information retrieval using layered multi-population genetic programming. In: Computational intelligence and cybernetics (CyberneticsCom), 2012 IEEE international conference on, pp 45–49, July https://doi.org/10.1109/CyberneticsCom.2012.6381614
22. Mick JYL (2016) Accessed http://people.cs.nctu.edu.tw/jylin/lagep/lagep.html
23. Metzler D, Bruce Croft W (2007) Linear feature-based models for information retrieval. Inf Retr 10(3):257–274. https://doi.org/10.1007/s10791-006-9019-z
24. Xu J, Li H (2007) Adarank: a boosting algorithm for information retrieval. In: Proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrieval, SIGIR '07, pp 391–398, New York, NY, USA, ACM. https://doi.org/10.1145/1277741.1277809
25. Islam MA (2013) RankGPES: learning to rank for information retrieval using a hybrid genetic programming with evolutionary strategies
26. Sculley D (2010) Combined regression and ranking. In: Proceedings of the 16th ACM SIGKDD international conference on knowledge discovery and data mining, KDD '10, pp 979–988, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0055-1. https://doi.org/10.1145/1835804.1835928
27. Dang V (2016) RankLib. http://www.cs.umass.edu/~vdang/ranklib.html
28. Christopher JC (2010) Burges from RankNet to LambdaRank to LambdaMART: an overview. Technical report, Microsoft Research, http://research.microsoft.com/en-us/um/people/cburges/techreports/MSR-TR-2010-82.pdf
29. Chapelle O, Chang Y (2011) Yahoo! learning to rank challenge overview. In: Proceedings of the Yahoo! learning to rank challenge, held at ICML 2010, Haifa, Israel, June 25, 2010, pp 1–24. http://www.jmlr.org/proceedings/papers/v14/chapelle11a.html
30. Ibrahim OAS, Landa-Silva D (2018) An evolutionary strategy with machine learning for learning to rank in information retrieval. Soft Comput 22:3171–3185. https://doi.org/10.1007/s00500-017-2988-6
31. Ai Q, Yang T, Wang H, Mao J (2021) Unbiased learning to rank: online or offline? ACM Trans Inf Syst 39(2):1046–8188. https://doi.org/10.1145/3439861
32. Lian J, Zhou X, Zhang F, Chen Z, Xie X, Sun G (2018) Xdeepfm: combining explicit and implicit feature interactions for recommender systems. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining, KDD '18, pp 1754–1763, New York, NY, USA, Association for Computing Machinery. ISBN 9781450355520. https://doi.org/10.1145/3219819.3220023
33. Cakir F, He K, Xia X, Kulis B, Sclaroff S (2019) Deep metric learning to rank. In: 2019 IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 1861–1870, https://doi.org/10.1109/CVPR.2019.00196
34. Craswell N, Zoeter O, Taylor M, Ramsey B (2008) An experimental comparison of click position-bias models. In: Proceedings of the 2008 international conference on web search and data mining, WSDM '08, pp 87–94, New York, NY, USA, ACM. https://doi.org/10.1145/1341531.1341545
35. Guo F, Liu C, Wang YM (2009) Efficient multiple-click models in web search. In: Proceedings of the second ACM international conference on web search and data mining, WSDM '09, pp 124–131, New York, NY, USA, ACM. https://doi.org/10.1145/1498759.1498818
36. Teugels JL (1990) Some representations of the multivariate Bernoulli and binomial distributions. J Multivar Anal 32(2):256–268. https://doi.org/10.1016/0047-259X(90)90084-U
37. Chuklin A, Markov I, de Rijke M (2015) Click models for web search. Synth Lect Inf Concepts Retr Serv 7(3):1–115
38. Cordón O, de Moya Anegón F, Zarco C (2002) A new evolutionary algorithm combining simulated annealing and genetic programming for relevance feedback in fuzzy information retrieval systems. Soft Comput 6(5):308–319. https://doi.org/10.1007/s00500-002-0184-8

39. Chirila C-B, Şora I (2019) The optimization of a page rank based key classes classifier using simulated annealing with ROC-AUC and recall metrics. In: 2019 IEEE 13th international symposium on applied computational intelligence and informatics (SACI), pp 000021–000026, https://doi.org/10.1109/SACI46893.2019.9111601
40. Landa Silva JD (2003) Metaheuristic and multiobjective approaches for space allocation. PhD thesis, University of Nottingham
41. Diaz-Gomez PA, Hougen DF (2007) Initial population for genetic algorithms: a metric approach. In: Proceedings of the 2007 international conference on genetic and evolutionary methods GEM, pp 43–49
42. Loshchilov I (2014) A computationally efficient limited memory CMA-ES for large scale optimization. In: Proceedings of the 2014 annual conference on genetic and evolutionary computation, GECCO '14, pp 397–404, New York, NY, USA, 2014. ACM. https://doi.org/10.1145/2576768.2598294
43. Qin T, Liu T-Y (2013) Introducing LETOR 4.0 datasets. CoRR, arXiv:1306.2597
44. Liu T-Y (2011) The LETOR datasets. In: Learning to rank for information retrieval. Springer Berlin Heidelberg, Berlin, Heidelberg, pp 133–143, https://doi.org/10.1007/978-3-642-14267-3_10
45. Baeza-Yates R, Ribeiro-Neto B (2011) Modern information retrieval: the concepts and technology behind search, 2nd edn. Pearson Education Ltd., Harlow http://www.mir2ed.org/
46. Chapelle O, Metlzer D, Zhang Y, Grinspan P (2009) Expected reciprocal rank for graded relevance. In: Proceedings of the 18th ACM conference on information and knowledge management, CIKM '09, pp 621–630, New York, NY, USA, Association for Computing Machinery. https://doi.org/10.1145/1645953.1646033
47. Whiteson S, de Rijke M, Schuth A, Hofmann K (2013) Lerot: an online learning to rank framework. In: Living labs for information retrieval evaluation workshop at CIKM-13, http://www.anneschuth.nl/wp-content/uploads/2013/09/cikm-livinglab-2013-lerot.pdf

Dr. Osman Ali Sadek Ibrahim is an Assistant Professor at the Department of Computer Science, Faculty of Science, Minia University, Egypt from 2018 till now. His research interests include Information Retrieval, Evolutionary Algorithms, and Machine Learning techniques. He obtained his B.Sc. degree from the Department of Computer Science, Faculty of Science, Minia University, Egypt, in 2002. He also obtained his M.Sc. degree by research in Computer Science titled: "Using Genetic Algorithms to improve Information Retrieval" in 2007 from the Department of Computer Science, Minia University, Egypt. In 2018, he obtained his Ph.D. degree from the University of Nottingham, Faculty of Science, School of Computer Science. His Ph.D. thesis is titled: "Evolutionary Algorithms and Machine Learning Techniques for Information Retrieval."

Eman M. G. Younis is currently working as an Associate Professor at Minia University, Faculty of Computers and Information and the Head of Information Systems Department. She obtained her B.Sc. degree from Zagazig University, Egypt, in 2002. She obtained her M.Sc. degree from Menoufia University, Egypt, in 2007. She obtained her Ph.D. degree from Cardiff University, UK, in 2014. She joined NTU, the Computing Department, as a researcher in 2016-2017.