Turning Clicks into Purchases: Revenue Optimization for Product Search in E-Commerce

Liang Wu
Arizona State University
699 South Mill Street
Tempe, AZ
wuliang@asu.edu

Diane Hu, Liangjie Hong
Etsy Inc.
117 Adams Street
Brooklyn, NY
{dhu,lhong}@etsy.com

Huan Liu
Arizona State University
699 South Mill Street
Tempe, AZ
huanliu@asu.edu

ABSTRACT

In recent years, product search engines have emerged as a key factor for online businesses. According to a recent survey, over *55%* of online customers begin their online shopping journey by searching on an E-Commerce (EC) website like Amazon as opposed to a generic web search engine like Google[1]. Information retrieval research to date has been focused on optimizing search ranking algorithms for web documents while little attention has been paid to product search. There are several intrinsic differences between web search and product search that make the direct application of traditional search ranking algorithms to EC search platforms difficult. First, the success of web and product search is measured differently; one seeks to optimize for relevance while the other must optimize for both relevance and revenue. Second, when using real-world EC transaction data, there is no access to manually annotated labels. In this paper, we address these differences with a novel learning framework for EC product search called LETORIF (LEarning TO Rank with Implicit Feedback). In this framework, we utilize implicit user feedback signals (such as user clicks and purchases) and jointly model the different stages of the shopping journey to optimize for EC sales revenue. We conduct experiments on real-world EC transaction data and introduce a a new evaluation metric to estimate expected revenue after re-ranking. Experimental results show that LETORIF outperforms top competitors in improving purchase rates and total revenue earned.

KEYWORDS

E-Commerce, Search logs, Revenue

ACM Reference format:
Liang Wu, Diane Hu, Liangjie Hong, and Huan Liu. 2018. Turning Clicks into Purchases: Revenue Optimization for Product Search in E-Commerce. In *Proceedings of The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, Ann Arbor, MI, USA, July 8–12, 2018 (SIGIR '18),* 10 pages.
DOI: 10.1145/3209978.3209993

1 INTRODUCTION

For any E-Commerce (EC) website, there can be millions of products to surface for a given search query. How can an EC platform determine which items are most relevant? More importantly, how can it produce a ranked list for its consumer that promises to generate optimal revenue? EC sales are expected to reach $*414* billion by 2018[2]. Given that EC search engines are the starting point for many online consumers, a subtle change in the ranking algorithm can lead to drastic changes in user satisfaction and revenue. In the past, LEarning TO Rank (LETOR) [20] has been effective for optimizing search engine results, but little attention has been paid to its use in EC platforms. In this work, we adapt LETOR by designing a ranking system that specifically caters to the needs of EC platforms.

Traditionally, LETOR constructs ranking models using manually obtained labels to sort documents according to their relevance to the query. These labels are usually static in the sense that, for a particular query, relevant documents remain the same for different users. Ranking metrics like NDCG (Normalized Discounted Cumulative Gain) and MRR (Mean Reciprocal Rank) are firstly introduced and defined on these static labels. Optimizing these static-based ranking measures has become a central topic of LETOR in the past decade. Since product search in an EC platform can also be formulated as a ranking problem, LETOR methods seem a natural fit for such scenarios. However, intrinsic differences between web search and product search have exposed several important challenges for applying LETOR to product search. One key issue is that, the notion of "relevance" is *blurred.* Different users come to EC websites with a wide spectrum of intents. Some wishes to purchase as soon as possible while others are just wondering on the sites to get inspired. Therefore, "relevance" for an EC search engine requires multiple signals to be integrated, such as clicks, favorites and most importantly purchases. This is very different from classic LETOR when labels can be clearly defined. It is very challenging, if not impossible, to utilize the same procedure from generic web search to obtain a static set of human labels for each query. Thus, two critical problems for product search is to define labels with multiple signals and optimize a revenue-related objective function.

In most EC platforms, the gold standard for measuring success is *Gross Merchandise Volume* (GMV). This indicates the total dollar amount transacted from merchandise sales; the overall revenue generated for the EC site is proportional to the GMV. An optimal ranking algorithm for an EC platform should therefore maximize the value of purchases per search session. Figure 1 illustrates the two stages of an EC product search session: (1) A consumer first

[1] https://www.bloomberg.com/news/articles/2016-09-27/more-than-50-of-shoppers-turn-first-to-amazon-in-product-search

[2] https://www.forbes.com/sites/forrester/2014/05/12/us-ecommerce-grows-reaching-414b-by-2018-but-physical-stores-will-live-on/

SIGIR '18, Ann Arbor, MI, USA
© 2018 ACM. 978-1-4503-5657-2/18/07…$15.00
DOI: 10.1145/3209978.3209993

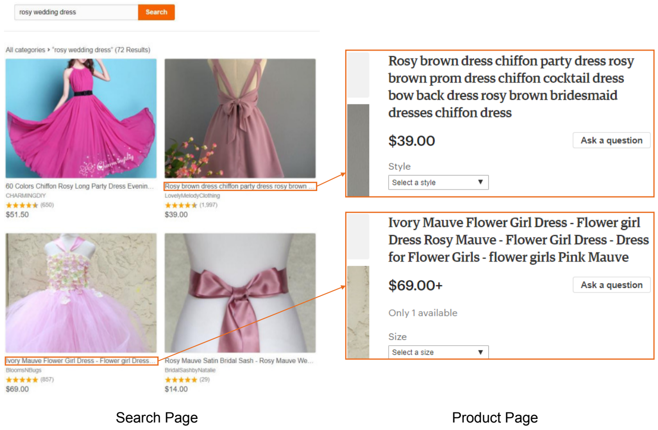
rosy wedding dress
Search
All categories › "rosy wedding dress" (72 Results)
60 Colors Chiffon Rosy Long Party Dress Evenin...
CHARMINGDIY
★★★★★ (650)
$51.50
Rosy brown dress chiffon party dress rosy brown ...
LovelyMelodyClothing
★★★★★ (1,997)
$39.00
Ivory Mauve Flower Girl Dress - Flower girl Dress...
BloomsNBugs
★★★★★ (857)
$69.00
Rosy Mauve Satin Bridal Sash - Rosy Mauve We...
BridalSashbyNatalie
★★★★★ (29)
$14.00
Rosy brown dress chiffon party dress rosy brown prom dress chiffon cocktail dress bow back dress rosy brown bridesmaid dresses chiffon dress
$39.00
Ask a question
Style
Select a style
Ivory Mauve Flower Girl Dress - Flower girl Dress Rosy Mauve - Flower Girl Dress - Dress for Flower Girls - flower girls Pink Mauve
$69.00+
Ask a question
Only 1 available
Size
Select a size
Search Page
Product Page

Figure 1: Illustration of a sample search session in an EC platform. The query is "rosy wedding dress", and the search result page is shown on the left and a portion of the product page for two items are shown on the right. This session consists of two stages: (1) selecting a product to click from a ranked list, and (2) deciding whether to purchase the product by reading its detailed description.

searches by a query and selects a product to click, and then (2) decides whether or not to purchase the product by examining its detailed description. As such, the first stage is a ranking problem, where listings should be presented to the consumer in the order of relevance, while the second stage is a binary classification problem, which models the consumer's decision to purchase or not. This is in contrast to a classical LETOR model that only considers the ranking problem (stage one) and does not address the classification task (stage two). This scenario is also different from classic sponsored search advertising, which usually only shows a limited number of ads per search page and usually there is no need to consider to optimize an explicit ranking metric like NDCG or MRR.

In this work, we study the entire search session, including the two stages of clicking (ranking) and purchasing (classification). The problem is challenging as the two tasks are inherently related to different loss functions: a ranked list is usually evaluated by a list-wise metric like NDCG, while a binary variable is usually measured by precision, accuracy, or error rate. In particular, we aim to investigate how implicit feedback can be utilized. It is difficult to directly apply implicit feedback like clicks for a generic search engine [14]. Our investigation on utilizing implicit feedback signals leads to a novel framework of revenue optimization for a two-stage EC search engine. Experiments on real-world data show the superiority of LETORIF over other competitive baselines, such as Random Forest and LambdaMART. To the best of our knowledge, this is the first piece of work that aims to discover the optimal ranking algorithm for optimizing revenue for an EC search platform. The contributions of this work can be summarized as follows:

- Introduce an emerging problem of optimizing revenue of search engines in E-Commerce;
- Present a novel learning framework to jointly utilize implicit user feedback signals to optimize revenue;
- Suggest mathematical formulation to efficiently solve the optimization problem;
- Evaluate the proposed method on real-world EC transaction data and elaborate on the effects of using different parameters and user feedback signals.

The remainder of this paper is structured as: In Section 2, we introduce the problem of search in EC. In Section 3, we introduce our proposed framework and the optimization method with theoretical analysis. We conduct experiments on real-world datasets in Section 4, and then introduce related work in Section 5. Finally, in Section 6, we conclude the paper and present future directions.

Table 1: Notations and Description

Notation	Description
s	search session
i, q, $\rho(i)$	item, query, position of i
i^s	an item impressed in session s
q^s	the query of session s
$\Psi(i), \Phi(i)$	event of click and purchase
$Price(i)$	price of item i
m	number of items in a search result page

2 PROBLEM STATEMENT

Let $s \in S$ be a search session consisting of a query is $q^s \in Q$ and m items will be displayed in a ranked list. An item $i^s \in I$ may be displayed with rank $\rho(i^s)$, and may be clicked $\Psi(i^s) = 1$ or purchased $\Phi(i^s) = 1$, or $\Psi(i^s) = 0$, $\Phi(i^s) = 0$. Given the price of each item $Price(i)$. We define the expected Gross Merchandise Volume (GMV) as:

$$GMV = \underbrace{\sum_{\forall s \in S}}_{\text{A search session}} \underbrace{\sum_{\forall i^s}}_{\text{An item in s}} \underbrace{Price(i^s)}_{\text{Price of } i^s} \underbrace{Pr(\Phi = 1 | i^s, q^s)}_{\text{Prob of purchase}}, \qquad (1)$$

where the expected sales value is accumulated for each search session. Our goal is to learn a ranking function $f(\cdot)$ that generates ranking lists to maximize the Expected Gross Merchandise Volume. Note that, a purchased product must have been clicked, while a clicked one is not necessarily to be purchased, so we omitted Ψ here. We will use revenue and GMV interchangeably for rest of the work. Notations throughout the work are presented in Table 1.

3 OPTIMIZING GMV WITH IMPLICIT FEEDBACK

One of the main challenges in utilizing existing search ranking techniques is the lack of labeled information in EC product search. Traditional LETOR methods are effective in optimizing list-wise loss functions given high-quality labeled information, which requires eliciting relevance ratings from human experts and crowdsourcing. However, there is no standard way of obtaining ground-truth information in the context of EC as employing manual annotation is nearly infeasible: hundreds of products may be similarly relevant to a query. Meanwhile, implicit user feedback is prevalently available on EC platforms and can be used to reveal the "quality" of a search result. We focus on user clicks and user purchases.

3.1 Optimizing Click

Given a set of search sessions S, for a certain query q, the quality of a product can be measured by how often it is being clicked. Hence, the query-specific Click-Through Rate (CTR) can be used to produce annotations as

$$CTR(i|q) = \sum_{s^q \in S} \frac{\Psi^{s^q}(i)}{|\{s^q \in S\}|}, \tag{2}$$

where $CTR(i|q)$ denotes the probability of item i being clicked given query q and s^q is a session that is with the query q. $CTR(i|q)$ reflects the "utility" of a listing and can be used for annotation and constructing a benchmark dataset. A ranked list sorted by CTR is similar to the training data for web search. LETOR methods can then be applied to learn a ranking function.

In the simplest form, LETOR models can directly predict the ranking or relevance score of items using regression or classification [18], commonly referred to as point-wise approaches. For example, a squared loss function can be used as

$$Loss_{point}(y_i^q) = (y_i^q - l_i^q)^2, \tag{3}$$

where the point-wise loss function $Loss_{point}$ denotes how far the predicted value y_i^q deviates from the actual score l_i^q and $l_i^q := CTR(i|q)$. However, point-wise approaches assume items to be independent and identically distributed (*i.i.d.*) and the interdependency among web documents is not considered. The position of an item in a ranked list depends on other items, so a point-wise ranking model is unable to see the position of an item or the final ranking. The usage of point-wise estimation greatly limits the effectiveness of these methods since most information retrieval tasks focus on the relative order of results instead of the absolute score.

Instead, pair-wise LETOR methods directly model the relative order of items. Items are modeled as a pair (i_a, i_b) and the loss function measures the inconsistency between the true relative order and the predicted one as

$$Loss_{pair}(y_{a,b}^q) = -y_{a,b}^q \log l_{a,b}^q - (1 - y_{a,b}^q)\log(1 - l_{a,b}^q), \tag{4}$$

where a cross-entropy loss function is used and $l_{a,b}^q$ is the probability that a is ranked higher than b. $l_{a,b}^q$ can be obtained with a sigmoid function,

$$l_{a,b}^q = \frac{1}{1 + e^{l_a^q - l_b^q}}. \tag{5}$$

A complete ranked list can be recovered by modeling the relative order with Eq.(4). As such, positions of items are indirectly visible to a pair-wise LETOR method [2]. A limitation is that pairs generated by different queries are mixed and the structure a query manifests cannot be fully exploited. In order to tackle the challenge, list-wise approaches jointly model the entire group of items.

A common evaluation metric for ranked lists is Normalized Discounted Cumulative Gain (NDCG) [13]. NDCG is often counted from the top and truncated at a rank position K. The formulation is written as

$$NDCG_K(\varrho) = N_{max}^{-1} \sum_{r=0}^{K-1} \frac{2^{l(r^{-1})}}{\log(1 + r)}, \tag{6}$$

where ϱ is a ranking order and $NDCG_K$ denotes the NDCG score at top K (NDCG@K). N_{max} is the maximum value of $\sum_{r=0}^{K-1} \frac{2^{l(r)}}{\log(1+r)}$ that is used as a partition function. r^{-1} is the item index that is ranked r in ϱ. $2^{l(\cdot)}$ is a gain function and $l(i)$ is the label value of item i. The maximum value is achieved when items are optimally ordered by decreasing label value. Through optimizing list-wise cost functions, positions of items are directly visible to the learning algorithm and the query-level structures are also considered. In this work, by contrast, we propose to directly optimize the entire ranked list for product search engines.

3.2 Optimizing Purchase

One main goal of search engines in EC is to maximize the number of purchases. Modeling a purchase is different from optimizing a search result page. A search result page presents items in a ranked list, while the decision to purchase is made after reading the product description page. A similar problem is predicting the conversion of a clicked ad: when a user clicks a sponsored link, the post-click reaction, such as purchase, download, registration and subscription, can be independently modeled [25]. Motivated by current studies on ad conversions [25, 44], we adopt a logistic regression model to estimate the conversion. The problem is to solve the following convex optimization problem,

$$\min_{w_p, b} \sum_{i=1}^{N} \log\{1 + \exp[-l_i'(w_p x_i + b)]\} + ||w_p||^2, \tag{7}$$

where w_p is the vector of model parameters and b is the model bias. For simplicity of presentation, we augment w_p by incorporating b, which can be implemented by adding an additional dummy feature. l_i' is the purchase label of an item in the list that represents whether the item was purchased or not in the search session.

3.3 Proposed Method for Product Search

Product search in EC consists of two stages: (1) clicking on a product from the search result page and (2) deciding whether to purchase. A similar problem is that of sponsored ad revenue estimation [25]. Revenue from displaying an ad comes from user clicks and/or some predefined action, such as registration, subscription or purchase. Current research can be classified into two genres: First, purchases can be modeled as higher weighted clicks (e.g., a purchase equals 1,000 clicks) [41]; Second, purchases can be modeled separately by assuming that they are independent of the previous search result page [25]. However, these current methods are not well-suited for the problem of product search either. Clicks on a search result page do not necessarily lead to a purchase since there are other intents beyond simply purchasing, such as browsing and searching [22]. Hence, regarding purchases as weighted clicks may introduce bias from noisy clicks. In addition, purchases and clicks are not independent. For example, such relationships usually hold in a search session: $P(\Phi = 1|\Psi = 0) = 0$ and $P(\Psi = 1|\Phi = 1) = 1$.

In our proposed method, we decompose the purchase into two stages: (1) clicking related items and (2) purchasing a specific one. We propose a nested framework to holistically model the interdependence of the two stages. We aim to predict the purchase behavior of users, and the conditional probability can be written as

$$Pr(\Phi = 1|i, q) = \underbrace{Pr(\Psi = 1|i, q)}_{\text{click model}}\,\underbrace{Pr(\Phi = 1|\Psi = 1, i, q)}_{\text{purchase model}}, \tag{8}$$

where $Pr(\Psi = 1|i, p, q)$ denotes the conditional probability of click given a displayed item, the display position and the query, and $Pr(\Phi = 1|\Psi = 1, i, q)$ is the conditional probability of purchase given the item and the query.

We focus on a typical search session, so being clicked is a prerequisite for an item to be purchased $Pr(\Phi = 1|\Psi = 0, i, q, u) = 0$. The display position ($p$) is omitted in the purchase model, namely

$$(\Phi \perp\!\!\!\perp P) \quad | \quad \Psi, \tag{9}$$

where the position is only considered in the search result page. Hence, the modeling of clicks is dependent on other displayed items, which leads us to a ranking problem, while the purchase model is a binomial classification task.

As discussed in Section 3.1 and Section 3.2, we aim to optimize user clicks in a list-wise manner while purchases in a binomial classification manner. Let us assume that we have a click probability ranking function $f_c(\cdot)$ that takes in product features x_{i_a} and produces the probability of the item being clicked $Pr(\Psi = 1|i_a)$, and a purchase probability function $f_p(\cdot)$ that takes in x_{i_a} and produces the probability of the item being purchased $Pr(\Phi = 1|i_a)$. The vectors of parameters to learn are w_c and w_p, respectively.

Learning $f_c(\cdot)$: Given the click probability ranking function $f_c(\cdot)$, the probability that an item i_a is ranked higher than i_b is proportional to:

$$Pr(r(i_a) < r(i_b)) \propto f_c(x_{i_a}) - f_c(x_{i_b}), \tag{10}$$

where $r(i_a)$ is the rank of i_a, e.g., $r(i_a) = 1$ represents i_a is the top ranked item. We denote the probability as $\tilde{Pr}(r(i_a) < r(i_b)) = f_c(x_{i_a}) - f_c(x_{i_b})$ for simplicity. Hence, the expected rank of an item can be estimated as

$$\tilde{r}(i_a) = \sum_{i_b=1,\, i_b \neq i_a}^{m} \mathbf{1}((f_c(x_{i_a}) - f_c(x_{i_b})) \leq 0), \tag{11}$$

where $\mathbf{1}$ is an indicator function that equals 1 if the condition holds and 0 otherwise. Therefore, the ranking can be estimated with the predicted score. In order to optimize the ranking order in a list-wise setting, we replace the r in Eq.(6) with the estimated rank in Eq.(11). The objective function can be formulated as

$$\max_{\varrho} N_{max}^{-1} \sum_{i=1}^{m} \frac{2^{l(i)}}{\log(1 + \tilde{r}(i))}, \tag{12}$$

where we rewrite the NDCG formulation with an item index i. However, it is difficult to directly optimize the objective function in Eq.(12) since the ranking $r(i)$ is not continuous due to the sorting in Eq.(11). We propose to relax the estimation of rank in Eq.(11) as

$$\bar{r}(i_a) = \sum_{i_b=1,\, i_b \neq i_a}^{m} \sigma(\tilde{Pr}(r(i_a) < r(i_b))), \tag{13}$$

where $\sigma(\cdot)$ is the sigmoid function. By incorporating $\sigma(\cdot)$, the pairwise comparison in Eq.(11) for sorting is transformed into a differentiable function. In addition, the new formulation is smooth and convex and can be conveniently optimized. By integrating Eq.(13) with 12, the new objective function can be formulated as

$$\max_{\varrho} N_{max}^{-1} \sum_{i=1}^{m} \frac{2^{l(i)}}{\log(1 + \sum_{j=1,\, j \neq i}^{m} \sigma(\tilde{Pr}(r(i) < r(j))))}, \tag{14}$$

where the rank is approximated with the estimated score. The objective function can be further decomposed as

$$\mathcal{L}_c = N_{max}^{-1} \sum_{i=1}^{m} \frac{2^{l(i)}}{\log(1 + \sum_{i_b=1,\, i_b \neq i_a}^{m} \sigma(f_c(x_a) - f_c(x_b)))}, \tag{15}$$

where the subscript for q is omitted since the calculation only happens within a search session. The maximization of Eq.(14) leads to a ranked list with an optimal estimated NDCG. In order to obtain the optimal parameters, we differentiate the objective function with respect to w_c. The derivative can be written as,

$$\frac{\partial \mathcal{L}_c}{\partial w_c} = \frac{\partial \mathcal{L}_c}{\partial \bar{r}(i)} \frac{\partial \bar{r}(i)}{\partial w_c}, \tag{16}$$

where the first term is the gradient of the NDCG score with respect to the ranking order, and the second term is defined by the logistic regression model in Eq.(13). Each search session contains a list of items, and we try to define a gradient vector for each item i.

Given the estimated rank of an item, the gradient of NDCG can be formulated as,

$$\frac{\partial \mathcal{L}_c}{\partial \bar{r}(i)} = N_{max}^{-1} \sum_{i=0}^{m} \frac{-2^{l(i)}}{(1 + \bar{r}(i)) \log^2(1 + \bar{r}(i))}, \tag{17}$$

where a large difference between the ground truth label $l(i)$ and the predicted ranking $\bar{r}(i)$ can lead to a significant change in the parameters, which will in turn change the item order.

Given the click probability function $f_c(\cdot)$, the gradient of the ranking order with respect to the parameters can be written as

$$\begin{aligned} \frac{\partial \bar{r}(i)}{\partial w_c} &= \frac{\partial \sum_{i_b=1,\, i_b \neq i_a}^{m} \sigma(f_c(x_a) - f_c(x_b))}{\partial w_c} \\ &= \frac{\partial f_c(x_a)}{\partial w_c} \sum_{i_b=1,\, i_b \neq i_a}^{m} \sigma[f_c(x_a) - f_c(x_b)]\{1 - \sigma[f_c(x_a) - f_c(x_b)]\}, \end{aligned} \tag{18}$$

where the first term is the gradient of the point-wise scoring function. Following existing LETOR studies [2], we adopt a neural net model, and the gradient $\frac{\partial f_c(x_a)}{\partial w_c}$ can be computed with back-propagation [16]. Note that other regression methods can also be used here, though in our experience the choice did not significantly impact performance. The second part is the gradient for the sigmoid function, where a similar ranking order between i_a and i_b leads to a greater change to the parameters. The gradient in Eq.(18) updates the ranking order according to the loss function and distinguishes the positions of items by enlarging the distance between them.

Learning $f_p(\cdot)$: As discussed in Section 3.2, prediction of purchases is modeled in a binary classification setting. We first rewrite the loss function in Eq.(7) by augmenting w_p with a model bias b.

$$\mathcal{L}_p = \sum_{i=1}^{N} \log\{1 + \exp[-l_i'(w_p x_i)]\} + ||w_p||^2, \tag{19}$$

where b is incorporated with w_p and the feature vector x_i is augmented by adding a dummy feature. Here, l_i' denotes whether the item has been purchased. Through minimizing $\mathcal{L}_p$, which is a convex optimization problem, we can achieve a vector of parameters w_p that optimally predicts purchases. The gradient of $\mathcal{L}_p$ can be written as

$$\frac{\partial \mathcal{L}_p}{\partial w_p^j} = \sum_{i=1}^{N} w_p^j (l_i' - w_p x_i) - 2w_p^j, \tag{20}$$

where w_p^j is the j^{th} feature, and the incorporated model bias is also jointly updated. A potential problem is that Eq.(19) maximizes the number of purchases, while the goal of an EC platform is to optimize GMV that also correlates with price of items. By introducing the functions for modeling clicks and purchases, the GMV can be estimated as

$$\begin{aligned} GMV &= \sum_{\forall s \in S} \sum_{\forall i^s} Price(i^s)\Phi(i^s) \\ &= \sum_{\forall s \in S} \sum_{\forall i^s} Price(i^s) Pr(\Psi = 1 | i^s, p(i^s), q^s) Pr(\Phi = 1 | i^s, q^s), \end{aligned} \tag{21}$$

where the price $Price(i^s)$ is introduced as a weighting term. We further incorporate the price of each item into the purchase model, leading to a new loss function,

$$\mathcal{L}_p = \sum_{i=1}^{N} Price(i) \log\{1 + \exp[-l_i'(w_p x_i)]\} + ||w_p||^2, \tag{22}$$

where $Price(i^s)$ is used to weight each item. Since cheaper items are usually purchased more frequently, incorporating the price of each item substantially increases the output of the purchase model when the items are more expensive. This is an appealing bias that we aim to achieve in order to optimize GMV.

Algorithm 1 Training Algorithm of LETORIF

```
Input: Search sessions: $S$
    Maximum number of iterations: $Max_{iter}$
    Early termination function : $EarlyStop()$
Output: Parameters of the model: $w_c, w_p$
1: Initialize $w_c$ and $w_p$ randomly, $VLoss[Max_{iter}]$, $iter = 0$
2: Split $S$ into training and validation set, $S_{tr}$ and $S_{val}$
3: do
4:     Train $f_c(\cdot)$ with $S_{tr}$ for 1 epoch with Eq.(18)
5:     Train $f_p(\cdot)$ with $S_{tr}$ for 1 epoch with Eq.(20)
6:     Test LETORIF with $(S_{val})$ to obtain loss $VLoss[i]$
7:     $iter = iter + 1$
8: while $EarlyStop(VLoss, i) = FALSE$ AND $(i < Max_{iter})$
```

We show the training algorithm of the proposed LETORIF in Algorithm 1. We input the labeled search sessions S which is randomly split into a training and a validation set in line 2. We set the maximum number of iterations to be Max_{iter}, and also use a function *EarlyStop()* to control early termination of the optimization process, taking the loss on the validation set as input. From line 3 to 8, we interchangeably optimize the click and purchase model until the maximum number of epochs is reached or the early termination condition is met. In order to optimize GMV, the price of each item is incorporated into Eq.(20) as a weighting term. NDCG is adopted to calculate the loss in line 6.

Table 2: Statistics of the EC search session dataset used in this study.

Sessions	Queries	Items	Avg. Items per Session
334,931	239,928	6,347,251	19.0
Keywords	Buyers	Sellers	Avg. Items per Query
631,778	270,239	550,025	26.5

4 EXPERIMENTS

In this section, we introduce details of the experiments to validate the effectiveness of the proposed method. We aim to answer two questions through the experiments:

(1) How well can LETORIF utilize implicit feedback signals to learn a ranking model compared with traditional LETOR methods and purchase models?

(2) How effective is the proposed LETORIF in improving the Gross Merchandise Volume through jointly modeling clicks, purchases and prices?

In order to answer the questions, we collect product search session data from a real-world EC platform, and conduct experiments with state-of-the-art approaches for comparison. Traditional evaluation metrics are mainly designed for measuring relevance between search queries and web documents. To better understand effectiveness of product search in an EC platform, we introduce a novel evaluation metric to measure the effect on revenue.

4.1 Datasets

We collect 4 weeks' worth of search log data with clicks and purchases from an EC platform, Etsy. In order to utilize implicit feedback signals of users, we focus on queries that receive at least 1 purchase and over 100 impressions. After randomly sampling across the filtered log data, we have in total $334,931$ search sessions with $239,928$ queries and $6,347,251$ items. There are $270,239$ buyers and $550,025$ sellers involved in the transactions. $631,778$ keywords are used by sellers to describe their items. Detailed statistics about the dataset are shown in Table 2.

For each query, we collect all corresponding search sessions, and calculate the average revenue for every impression. The average revenue for a query-item pair i, q can be calculated as,

$$Avg.Rev(i, q) = \frac{price(i) \times purchase(i, q)}{|\{S_q | i \in S_q\}|}, \tag{23}$$

where $purchase(i, q)$ denotes the number of times i has been purchased in a search session for query q, and $\{S_q | i \in S_q\}$ are the set of search sessions for query q where the item i is impressed (shown). We follow the conventional practice of LETOR research to label the dataset in an ordinal, integer scale [23]. For each query, items are ordered with an evaluation measure. We test average revenue as well as click-through rate and purchase rate. More details can be found in Section 4.3.1

We extract 54 features to represent pairs of queries and items, which are shown in Table 3. These features can be separated into two categories: $51(17 \times 3)$ relevance features and 3 revenue features. Relevance features are computed from titles, descriptions and user-compiled tags of items, and the corresponding search query. The

Table 3: A list of features represented for pairs of queries and items. Titles, descriptions and user-compiled tags are extracted for each item. We focus on relevance-centric and revenue-centric features. Relevance features consist of low level features that can directly be calculated from text, and high level features that are built upon language modeling.

Relevance	Low Level	Sum of TF
		Sum of Log TF
		Sum of Normalized TF
		Sum of Log Normalized TF
		Sum of IDF
		Sum of Log IDF
		Sum of ICF
		Sum of TF-IDF
		Sum of Log TF-IDF
		TF-Log IDF
		$Length$
		Log $Length$
	High Level	$BM25$
		Log $BM25$
		LM_{DIR}
		LM_{JM}
		LM_{ABS}
Revenue		$Price$
		$Price - Cat.Mean$
		$(Price - Cat.Mean)/Cat.Mean$

low level relevance features include TF (term frequency), IDF (inverse document frequency), ICF (inverse corpus frequency), length (number of words) and their variations. The high level relevance features are the outputs from the Okapi BM25 and LMIR ranking algorithms, which measure the similarity between queries and text. We use different smoothing methods for LMIR including DIR, JM and ABS. Details of relevance features can be found in [23]. Revenue features include the price of an item, and how far it deviates from the average price of the corresponding category.

4.2 Experimental Settings

In order to answer the two questions above, we conduct two experiments to validate the effectiveness of the proposed LETORIF. In the first experiment, we follow the conventional practice of information retrieval studies, and adopt NDCG to evaluate the ranking power of different methods for each query. We collect a query-based dataset, as mentioned in Section 4.1. In order to perform a five-fold cross-validation, we shuffle all data instances and randomly split them into five folds. In each iteration, one fold of data is used for testing while the other four for training. The average NDCG of five folds of experiments is reported. In order to evaluate different aspects of each methods, we introduce three NDCG scores: click-based NDCG, purchase-based NDCG and revenue-based NDCG, which are obtained based on the descending ranking order of average click-through rate, average purchase rate and average revenue, respectively. In this work we mainly focus on revenue-based NDCG while the other two are used to provide insights.

In the second experiment, we aim to evaluate how LETORIF can improve the actual revenue of search sessions. We use the adopted methods to rerank the search results of each search session, and observe how the reranked list would affect the actual revenue. We chronologically partition search sessions into 80/20 splits, where the first split is used for training and the second for test. In order to measure the revenue, we define a metric called *Rev@K*. We assume that an item that has been actually purchased, will also be purchased under the new ranking order if it is shown in the top K positions. *Rev@K* can be seen as the average revenue that a prediction algorithm would generate for each session. In real applications, a user may click various results instead of a fixed number. We make the assumption of top K to measure the ranking quality for optimizing revenue. In particular, calculation of *Rev@K* can be formulated as

$$Rev@K(\varrho) = \sum_{\forall s \in S} \sum_{r_s \leq K} Price(r_s^{-1})\Phi(r_s^{-1}), \qquad (24)$$

where ϱ is the ranking order and $r_s \leq K$ denotes the top K ranked positions in the session s. r_s^{-1} represents the corresponding item at the position.

In addition to LETORIF, we include approaches that fall in three categories: state-of-the-art relevance methods that optimize user clicks, purchase prediction methods that model the purchase behavior, and methods that jointly optimize purchases and clicks.

For relevance methods, we include the following 7 Learning-To-Rank methods:

- RankNet (RNet) is one of the first adopted LETOR algorithms. A neural network is used to minimize the pair-wise loss function (cross entropy).
- RankBoost (RBoost) extends AdaBoost [9] by replacing the loss function for classification with a pair-wise loss function.
- AdaRank (ARank) adapts AdaBoost to solve ranking problems by assigning weights to queries. AdaRank focuses more on the difficult queries/search sessions.
- LambdaRank (LRank) uses a neural network to minimize the pair-wise loss function, which is similar to RankNet. LambdaRank weights each pair by how it affects the list-wise loss function.
- ListNet (LNet) is a list-wise approach that calculates the probability of each position of a ranked list, and uses a neural network to minimize the K-L divergence [15] to the ideal ranking order.
- Multiple Additive Regression Trees (MART) is a tree boosting algorithm that consists of multiple decision trees as weak learners. Pair-wise loss functions are adopted.
- LambdaMART (LMART) extends MART by introducing a weighting term for each pair of data, as how LambdaRank extends RankNet with list-wise measures.

In addition, we study the effectiveness of directly modeling purchases by including 3 classification methods to directly model the purchase behavior:

- Support Vector Machines (SVM) are widely used for binary classification. We test with different combinations

Table 4: Baseline methods implemented for comparison in this work. Popular LETOR methods, purchase prediction methods and two joint models are adopted.

Click	RankNet [2]	RNet
	RankBoost [8]	RBoost
	AdaRank [36]	ARank
	LambdaRank [3]	LRank
	ListNet [4]	LNet
	MART [10]	MART
	LambdaMART [35]	LMART
Purchase	SVM [5]	SVM
	Logistic Regression [25]	LR
	Random Forest [19]	RM
Both	Weighted Purchase [41]	WT
	LMART+RM	LMRM
	LETORIF	LETORIF

of loss functions and regularizations and the best performance is reported, though it did not affect the performance significantly.

- Logistic Regression (LR) has been adopted to predict post-click behaviors of web search engines that directly model the probability of purchase.
- Random Forest (RM) is a bagging method that trains multiple decision tree models and ensembles them. RM is a state-of-the-art nonparametric classifier and is relatively insensitive to noise.

LETORIF jointly utilizes purchases and clicks to optimize the expected GMV. Hence, we also include 2 baselines that optimizes expected revenue through jointly optimizing clicks and purchases:

- A common practice in industry is to assign more weight to the behaviors that lead to more revenue, such as purchases and conversions. In order for comparison, we follow the practice to linearly integrate purchases as WeighTed purchases (WT) and learn a relevance model [41].
- LETORIF simultaneously models clicks and purchases for revenue optimization. In order to investigate the effect of jointly modeling the two data sources, we introduce to ensemble a relevance model and a purchase model by estimating the joint probability. LambdaMART (LMART) and Random Forest (RM) are selected by testing possible combinations and the baseline method is denoted as LMRM.

The selected 12 baseline methods are shown in Table 4 as well as the proposed method LETORIF. WT is the state-of-the-art method for optimizing revenue in computational advertising. Short forms of these methods are introduced for the convenience of notation. All model parameters of baseline methods (*e.g.*, the number of trees, tree depths) are tuned on the holdout validation set.

4.3 Experimental Results

In this section, we discuss our experimental results of NDCG in a conventional LETOR setting, and results *w.r.t. Rev@K* showing how well the methods can improve actual revenue.

4.3.1 Comparison on NDCG. Table 5 illustrates the NDCG@5 for different methods on NDCG@5. We report the NDCG@5 on training (Train), validation (Vali) and test datasets. Based on the experiments, we make following observations:

Revenue NDCG LETORIF achieves the best Revenue NDCG among all methods, showing the effectiveness of jointly optimizing user click and purchase behaviors. MART and RM achieve the best score among click and purchase optimization methods, respectively. Click-based methods generally perform better than purchase-based methods, which indicates that product search is more a ranking problem and the interdependency between products exists. Though LMART and RM get relatively high NDCG@5 on the training dataset, they are outperformed by the runner-up method MART due to over-fitting. By linearly integrating purchase information, WT outperforms most relevance methods except MART. Similar to LMART, LMRM achieves a high training and validation NDCG while outperformed by LETORIF due to overfitting since the performance on test data is much lower than that on validation and training data.

Click NDCG Ignoring purchase behaviors makes the EC search sessions identical to a web document dataset. In this experiment, LMART achieves the best Click NDCG@5 among all methods, which is consistent with prior observations on benchmark studies of web search that LMART is the best performing method. MART is the runner-up method and the other click-based method RNet also achieves relatively high NDCG@5. Purchase-based methods can be viewed as point-wise methods that predict the purchase behavior. The superiority of click methods further proves the importance of modeling interdependency between items.

Purchase NDCG LETORIF achieves the best purchase NDCG. Purchase-based methods outperform all click-based methods but MART. The result shows that though the purchase of an item is independent, jointly modeling the interdependency of click behaviors between different products and user purchases positively impact the performance. Revenue NDCG can be viewed as the purchase NDCG weighted by the product price. By taking price into consideration, LETORIF outperforms other baseline methods by a larger margin. This is an appealing bias that we would like to achieve and it helps the optimization to focus on items with greater estimated revenue. The superiority of LETORIF over the other two joint baselines WT and LMRM reveals that linearly integrating purchases with a higher weight or completely dealing with the two problems independently cannot achieve the optimal performance.

4.3.2 Comparison on Revenue. Table 6 illustrates the *Rev@K* for different methods with a varying K. We vary the range from 1 to 10 to focus on the top ranked items. According to the definition of *Rev@K*, a larger K represents that a customer is more likely to purchase, and a ranking algorithm is easier to obtain revenue. Based on the experimental results we make following observations:

When $K = 1$, MART achieves the best actual revenue. The problem here becomes a point-wise task if only the top ranked item brings about revenue. This setting is a little strict compared with actual search sessions. In Figure 2 we show the distribution of positions of actual purchased items in the top 4 ranks of a search result page, where nearly 70% purchases are made below the first position. LETORIF is the runner-up method in the setting.

Table 5: Comparison of different ranking algorithms in terms of NDCG@5 on training, test and validation sets. Five-fold cross-validation has been adopted and a holdout set is used as a validation set. The three NDCG@5 are calculated based on click-through rate, purchase rate and average revenue, respectively.

Category	Method	Click NDCG@5			Purchase NDCG@5			Revenue NDCG@5		
		Train	Vali	Test	Train	Vali	Test	Train	Vali	Test
Click	RNet	0.1743	0.1731	0.1378^{**}	0.1672	0.1721	0.1676^{**}	0.1692	0.1700	0.1356^{**}
	RBoost	0.2150	0.1768	0.1323^{**}	0.2150	0.1768	0.1715^{**}	0.2150	0.1768	0.1311^{**}
	ARank	0.1718	0.1711	0.1351^{**}	0.1718	0.1711	0.1706^{**}	0.1718	0.1711	0.1358^{**}
	LRank	0.1694	0.1688	0.1360^{**}	0.1678	0.1711	0.1672^{**}	0.1713	0.1719	0.1366^{**}
	LNet	0.1665	0.1703	0.1355^{**}	0.1601	0.1682	0.1620^{**}	0.1646	0.1696	0.1348^{**}
	MART	0.2700	0.1758	0.1380^{**}	0.2155	0.1803	0.1796^{*}	0.2696	0.1688	0.1408^{**}
	LMART	0.3056	0.1777	**0.1412**	0.3056	0.1777	0.1717^{**}	0.3056	0.1777	0.1370^{**}
Purchase	SVM	0.1785	0.1772	0.1336^{**}	0.1831	0.1754	0.1755^{**}	0.1816	0.1752	0.1320^{**}
	LR	0.1978	0.1739	0.1310^{**}	0.1978	0.1739	0.1782^{**}	0.1978	0.1739	0.1332^{**}
	RM	0.3359	0.1698	0.1363^{**}	0.3329	0.2305	0.1798^{**}	0.3327	0.1685	0.1376^{**}
Both	WT	0.1970	0.1682	0.1334^{**}	0.1815	0.1763	0.1761^{**}	0.1781	0.1648	0.1375^{**}
	LMRM	0.2943	0.2597	0.1354^{**}	0.3087	0.2530	0.1688^{**}	0.2943	0.2594	0.1332^{**}
	LETORIF	0.1765	0.1550	0.1351^{**}	0.2731	0.1841	**0.1801**	0.2039	0.1698	**0.1494**

Symbol * indicates that the method is outperformed by the best one by 0.05 statistical significance level, ** indicates 0.01.

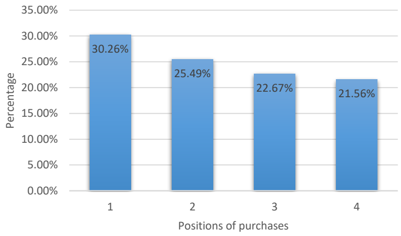

Figure 2: Distribution of positions of items being purchased in the top 4 spots of a search result page. The first position achieves the most purchases, while nearly 70% of purchases are in the lower positions.

When $1 < K \leq 10$, LETORIF outperforms all other baselines. Runner-up methods mostly belong to the category of click. RM is the runner-up method when $K = 2$, and RNet when $K = 3$, and LMART when $3 < K \leq 10$. The result indicates that by reranking the search sessions, the proposed LETORIF can bring about more revenue when items favored by a user is ranked among top 2 to 10.

Based on the observations, LETORIF is proven to be an effective LETOR method for optimizing revenue for an EC search engine in both the classic LETOR evaluation setting. As an individual model LETORIF performs best under most settings and is the runner-up $w.r.t.$ *Rev*@1. It can be used to directly optimize search sessions or easily incorporated to an existing system to complement with a ranking or classification model.

5 RELATED WORK

LETOR models are supervised learning methods that aim to estimate the relevance between queries and entities (documents, products, blog posts, *etc.*). In web search, for example, a query is associated with a number of documents and each document is associated with a relevance score. With the relevance scores, the objective of LETOR methods is to optimize some predefined evaluation metrics, such as NDCG or cross entropy. According to the loss functions of LETOR methods, they can be categorized into point-wise, pair-wise and list-wise methods. Point-wise methods predict each document individually, pair-wise methods predict the relative order of each two of the documents, and list-wise methods directly optimize the list-wise evaluation metrics. Popular LETOR methods that have been widely used in academia and industry are mostly pair-wise and list-wise methods, including RankBoost [8], AdaRank [36], MART [10, 42] which are point-wise, and LambdaRank [3], LambdaMART [35], and ListNet [4] which are list-wise.

In the context of web search, relevance is a main metric for ranking web documents. Besides manually labeling relevance score between a query and a document, user feedback signals can also be used to infer the relevance. Existing work mainly focuses on the click-through rate assuming that a more intensively clicked document is more likely to be relevant to the corresponding search query [32], and various approaches and heuristics have been used to alleviate bias brought by users and positions. In the context of product search, the association between user attention and satisfaction is unclear, and the user satisfaction for a search session cannot be directly obtained. Previous research has also studied how the user satisfaction can be measured and predicted based on user behaviors [6]. A detailed description of how search relevance is measured in real-world search engines has also been reported [39].

Product search in an EC platform has become an emerging challenge due to the increased popularity of EC websites and their local search engines. Economic theory models and Markov chains [17] have been explored to help rank product search results. Duan *et al.* study a keyword-based product search engine [7]. Recent work explores general guidelines on best practices for adapting LETOR methods for product search [26] where different LETOR methods and experimental settings have been studied. In order to obtain ground-truth data, they also test different kinds of user feedback signals including click-through rate, cart-add rate, order rate and total revenue. Our work differentiates itself from existing research

Table 6: Comparison of average revenue of search sessions in terms of *Rev@K* with a varying *K*, from top 1 to top 10. For each search session, the following methods are adopted to rerank the results and *Rev@K* is calculated based on the actual purchase.

Category	Method	Rev@1	Rev@2	Rev@3	Rev@4	Rev@5	Rev@6	Rev@7	Rev@8	Rev@9	Rev@10
Click	RNet	4.47**	4.69**	4.89**	4.91*	5.06**	5.23**	5.21**	5.33**	5.46**	5.55**
	RBoost	4.57**	4.69**	4.69**	4.76**	4.97**	5.17**	5.23**	5.36**	5.49**	5.57**
	ARank	4.37**	4.66**	4.76**	4.90**	5.06**	5.20*	5.33**	5.47**	5.59**	5.67**
	LRank	4.38**	4.61**	4.74**	4.86**	5.07**	5.25**	5.42**	5.42**	5.67**	5.78**
	LNet	4.30**	4.59**	4.78**	4.99**	5.16**	5.35**	5.49**	5.61**	5.63**	5.63**
	MART	**4.62**	4.72**	4.86**	5.04**	5.26**	5.47**	5.47**	5.64**	5.74**	5.86**
	LMART	4.46*	4.54**	4.73**	5.10**	5.31**	5.56**	5.75**	5.90*	6.01**	6.14**
Purchase	SVM	4.41**	4.54**	4.76**	4.77**	4.95**	5.16**	5.34**	5.50**	5.64**	5.77**
	LR	4.29**	4.65**	4.65**	4.69**	4.74**	4.81*	4.94**	4.97**	5.11**	5.11**
	RM	4.52**	4.82**	4.86**	5.02**	5.18**	5.33*	5.50**	5.66**	5.79**	5.92**
Both	WT	4.52**	4.69**	4.80**	4.85**	5.01**	5.07**	5.23**	5.32**	5.35**	5.41**
	LMRM	4.42**	4.50**	4.72**	5.08**	5.23**	5.41**	5.57**	5.60**	5.73**	5.85**
	LETORIF	4.58**	**4.90**	**5.08**	**5.47**	**5.64**	**5.85**	**6.02**	**6.19**	**6.40**	**6.54**

Symbol * indicates that the method is outperformed by the best one by 0.05 statistical significance level, ** indicates 0.01.

by focusing on optimizing the whole session of product search: clicking on a search result page and purchasing on a product description page. There are also studies exploring other aspects of the EC search problem, such as facet selection [30], search result diversification [40], feature learning [1, 29] and product popularity [21]. Our work is also related to NDCG optimization. NDCG is not differentiable since it requires the ranking order of items, and surrogate functions usually need to be used, such as a function that upper bounds the NDCG measure [28], or a probabilistic position estimate function [27]. Another line of research utilizes the change of NDCG measure to regularize the optimization process [37].

The problem of predicting a purchase behavior is similar to that of post-click prediction of search sponsored ads since both problems consist of a prior stage of clicks and a final stage of conversion. Current research mainly regards the conversion as an independent stage [11, 25], and binomial classification models are usually adopted. The problem is also related to estimating user behaviors by utilizing a user's content [12] and network information [33, 34]. There has also been research focused on directly optimizing revenue of search sponsored ads. Radlinski *et al.* proposes a keyword-based method that linearly combines the bidding price with the estimated click-through rate to maximize the revenue of a search engine platform [24]. Zhu *et al.* propose to jointly optimize the relevance and bidding price [43, 44]. However, current revenue optimization research in search sponsored ads usually models the click stage as a point-wise estimation task, since the number of ads is usually small. Our work optimizes a list-wise cost function since there are usually more products in a typical EC search session and the ranking order matters. There is computational advertising research that studies the maximization of expected revenue by considering clicks and bidding price [31]. However, revenue optimization of product search and advertising are intrinsically different since price of a product is the cost a user has to pay while price of an ad is the cost that an advertiser bids to pay. A detailed description about transforming click streams of online advertising into conversions has been reported [38].

Evaluation metrics for an information retrieval system are used to measure the extent of similarity between a rank list and the ground truth. Several list-wise cost functions are commonly used to evaluate a LETOR task, such as DCG@K, NDCG@K, Precision@K, BEST@K, ERR@K and RR@K. Due to limitation of space, we select NDCG@K in this work, which is relatively more often used than other metrics in the literature. In fact, there is no significant difference between different metrics. In addition to the offline metrics, online metrics such as click-through rate, session success and abandonment rate have also been used to measure the extent of success of an information retrieval system, which focus on data extracted from user logs. However, these metrics are concentrated on web search, and ignoring revenue, which is the key measure to evaluate success of a product search engine, To this end, we propose a novel evaluation metric for ranking lists, Rev@K, which provides a clearer perspective to understand how the ranking methods impact the actual revenue in a real-world setting.

6 CONCLUSION AND FUTURE WORK

With the popularity of EC platforms, product search has been an emerging issue for improving consumer satisfaction of online business. Traditional research regards search in EC platforms as a special case of web search, where products are being processed as web documents. In contrast, our work reveals the key feature of product search that distinguishes itself from web search: it consists of two stages, namely comparing and clicking a product on a search result page and deciding whether to purchase a product on the product description page. A novel LETOR framework is proposed to optimize the two stages jointly with implicit user feedback signals. Mathematical formulation is suggested to efficiently solve the optimization problem, and we conduct extensive experiments on real-world EC search log data to elaborate on different aspects of the proposed method.

Several interesting future directions remain to be studied. Since we focus on implicit user feedback signals, it would be interesting to explore the possibility of having manual annotators or crowdsourcing workers to label the dataset. It would also be interesting to study how personalization can be adopted to further boost the performance.

ACKNOWLEDGMENTS

This material is based upon work supported by, or in part by, the National Science Foundation (NSF) grant 1614576, and the Office of Naval Research (ONR) grant N00014-16-1-2257.

REFERENCES

[1] Kamelia Aryafar, Devin Guillory, and Liangjie Hong. 2017. An Ensemble-based Approach to Click-Through Rate Prediction for Promoted Listings at Etsy. In *Proceedings of the ADKDD'17*. ACM, 10.

[2] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*. ACM, 89–96.

[3] Christopher JC Burges. 2010. From ranknet to lambdarank to lambdamart: An overview. *Learning* 11, 23-581 (2010), 81.

[4] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. 2007. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning*. ACM, 129–136.

[5] Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)* 2, 3 (2011), 27.

[6] Ovidiu Dan and Brian D Davison. 2016. Measuring and Predicting Search Engine Usersfi Satisfaction. *ACM Computing Surveys (CSUR)* 49, 1 (2016), 18.

[7] Huizhong Duan, ChengXiang Zhai, Jinxing Cheng, and Abhishek Gattani. 2013. Supporting keyword search in product database: a probabilistic approach. *Proceedings of the VLDB Endowment* 6, 14 (2013), 1786–1797.

[8] Yoav Freund, Raj Iyer, Robert E Schapire, and Yoram Singer. 2003. An efficient boosting algorithm for combining preferences. *Journal of machine learning research* 4, Nov (2003), 933–969.

[9] Yoav Freund and Robert E Schapire. 1995. A desicion-theoretic generalization of on-line learning and an application to boosting. In *European conference on computational learning theory*. Springer, 23–37.

[10] Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics* (2001), 1189–1232.

[11] Qi Guo and Eugene Agichtein. 2012. Beyond dwell time: estimating document relevance from cursor movements and other post-click searcher behavior. In *Proceedings of the 21st international conference on World Wide Web*. ACM, 569–578.

[12] Liangjie Hong, Aziz S Doumith, and Brian D Davison. 2013. Co-factorization machines: modeling user interests and predicting individual decisions in twitter. In *Proceedings of the sixth ACM international conference on Web search and data mining*. ACM, 557–566.

[13] Kalervo Järvelin and Jaana Kekäläinen. 2000. IR evaluation methods for retrieving highly relevant documents. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 41–48.

[14] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. 2017. Accurately interpreting clickthrough data as implicit feedback. In *ACM SIGIR Forum*, Vol. 51. Acm, 4–11.

[15] Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics* 22, 1 (1951), 79–86.

[16] Yann A LeCun, Léon Bottou, Genevieve B Orr, and Klaus-Robert Müller. 2012. Efficient backprop. In *Neural networks: Tricks of the trade*. Springer, 9–48.

[17] Beibei Li, Anindya Ghose, and Panagiotis G Ipeirotis. 2011. Towards a theory model for product search. In *Proceedings of the 20th international conference on World wide web*. ACM, 327–336.

[18] Ping Li, Qiang Wu, and Christopher J Burges. 2008. Mcrank: Learning to rank using multiple classification and gradient boosting. In *Advances in neural information processing systems*. 897–904.

[19] Andy Liaw, Matthew Wiener, et al. 2002. Classification and regression by randomForest. *R news* 2, 3 (2002), 18–22.

[20] Tie-Yan Liu. 2009. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval* 3, 3 (2009), 225–331.

[21] Bo Long, Jiang Bian, Anlei Dong, and Yi Chang. 2012. Enhancing product search by best-selling prediction in e-commerce. In *Proceedings of the 21st ACM international conference on Information and knowledge management*. ACM, 2479–2482.

[22] Wendy W Moe. 2003. Buying, searching, or browsing: Differentiating between online shoppers using in-store navigational clickstream. *Journal of consumer psychology* 13, 1-2 (2003), 29–39.

[23] Tao Qin, Tie-Yan Liu, Jun Xu, and Hang Li. 2010. LETOR: A benchmark collection for research on learning to rank for information retrieval. *Information Retrieval* 13, 4 (2010), 346–374.

[24] Filip Radlinski, Andrei Broder, Peter Ciccolo, Evgeniy Gabrilovich, Vanja Josifovski, and Lance Riedel. 2008. Optimizing relevance and revenue in ad search: a query substitution approach. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 403–410.

[25] Rómer Rosales, Haibin Cheng, and Eren Manavoglu. 2012. Post-click conversion modeling and analysis for non-guaranteed delivery display advertising. In *Proceedings of the fifth ACM international conference on Web search and data mining*. ACM, 293–302.

[26] Shubhra Kanti Karmaker Santu, Parikshit Sondhi, and ChengXiang Zhai. 2017. On Application of Learning to Rank for E-Commerce Search. (2017).

[27] Michael Taylor, John Guiver, Stephen Robertson, and Tom Minka. 2008. Softrank: optimizing non-smooth rank metrics. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*. ACM, 77–86.

[28] Hamed Valizadegan, Rong Jin, Ruofei Zhang, and Jianchang Mao. 2009. Learning to rank by optimizing ndcg measure. In *Advances in neural information processing systems*. 1883–1891.

[29] Christophe Van Gysel, Maarten de Rijke, and Evangelos Kanoulas. 2016. Learning latent vector spaces for product search. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 165–174.

[30] Damir Vandic, Flavius Frasincar, and Uzay Kaymak. 2013. Facet selection algorithms for web product search. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*. ACM, 2327–2332.

[31] Flavian Vasile, Damien Lefortier, and Olivier Chapelle. 2016. Cost-sensitive learning for utility optimization in online advertising auctions. *arXiv preprint arXiv:1603.03713* (2016).

[32] Xuanhui Wang, Michael Bendersky, Donald Metzler, and Marc Najork. 2016. Learning to rank with selection bias in personal search. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 115–124.

[33] Liang Wu, Xia Hu, and Huan Liu. 2016. Relational learning with social status analysis. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*. ACM, 513–522.

[34] Liang Wu, Jundong Li, Fred Morstatter, and Huan Liu. 2018. Toward Relational Learning with Misinformation. In *Proceedings of the 2018 SIAM International Conference on Data Mining*. SIAM.

[35] Qiang Wu, Christopher JC Burges, Krysta M Svore, and Jianfeng Gao. 2010. Adapting boosting for information retrieval measures. *Information Retrieval* 13, 3 (2010), 254–270.

[36] Jun Xu and Hang Li. 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*. ACM, 391–398.

[37] Jun Xu, Tie-Yan Liu, Min Lu, Hang Li, and Wei-Ying Ma. 2008. Directly optimizing evaluation measures in learning to rank. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 107–114.

[38] Yanwu Yang, Yinghui Catherine Yang, Bernard J Jansen, and Mounia Lalmas. 2017. Computational Advertising: A Paradigm Shift for Advertising and Marketing? *IEEE Intelligent Systems* 32, 3 (2017), 3–6.

[39] Dawei Yin, Yuening Hu, Jiliang Tang, Tim Daly, Mianwei Zhou, Hua Ouyang, Jianhui Chen, Changsung Kang, Hongbo Deng, Chikashi Nobata, et al. 2016. Ranking relevance in yahoo search. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 323–332.

[40] Jun Yu, Sunil Mohan, Duangmanee Pew Putthividhya, and Weng-Keen Wong. 2014. Latent dirichlet allocation based diversified retrieval for e-commerce search. In *Proceedings of the 7th ACM international conference on Web search and data mining*. ACM, 463–472.

[41] Weinan Zhang, Shuai Yuan, and Jun Wang. 2014. Optimal real-time bidding for display advertising. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1077–1086.

[42] Qian Zhao, Yue Shi, and Liangjie Hong. 2017. GB-CENT: Gradient Boosted Categorical Embedding and Numerical Trees. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1311–1319.

[43] Yunzhang Zhu, Gang Wang, Junli Yang, Dakan Wang, Jun Yan, and Zheng Chen. 2009. Revenue optimization with relevance constraint in sponsored search. In *Proceedings of the Third International Workshop on Data Mining and Audience Intelligence for Advertising*. ACM, 55–60.

[44] Yunzhang Zhu, Gang Wang, Junli Yang, Dakan Wang, Jun Yan, Jian Hu, and Zheng Chen. 2009. Optimizing search engine revenue in sponsored search. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 588–595.