Exploiting User Preference for Online Learning in Web Content Optimization Systems

JIANG BIAN, Microsoft Research
BO LONG, LinkedIn
LIHONG LI, Microsoft Research
TAESUP MOON, University of California at Berkerley
ANLEI DONG and YI CHANG, Yahoo! Labs

Web portal services have become an important medium to deliver digital content, e.g. news, advertisements, etc., to Web users in a timely fashion. To attract more users to various content modules on the Web portal, it is necessary to design a recommender system that can effectively achieve Web portal content optimization by automatically estimating content items' attractiveness and relevance to users' interests. The state-of-the-art online learning methodology adapts dedicated pointwise models to independently estimate the attractiveness score for each candidate content item. Although such pointwise models can be easily adapted for online recommendation, there still remain a few critical problems. First, this pointwise methodology fails to use invaluable user preferences between content items. Moreover, the performance of pointwise models decreases drastically when facing the problem of sparse learning samples. To address these problems, we propose to explore a new dynamic pairwise learning methodology for Web portal content optimization, in which we exploit dynamic user preferences extracted based on users' actions on portal services to compute the attractiveness scores of content items. In this paper, we introduce two specific pairwise learning algorithms, a straightforward graph-based algorithm and a formalized Bayesian modeling one. Experiments on large-scale data from a commercial Web portal demonstrate the significant improvement of pairwise methodologies over the baseline pointwise models. Further analysis illustrates that our new pairwise learning approaches can benefit personalized recommendation more than pointwise models, since the data sparsity is more critical for personalized content optimization.

Categories and Subject Descriptors: H.3.3 [**Information Search and Retrieval**]: Information filtering; H.3.5 [**On-line Information Services**]: Web-based services

General Terms: Algorithms, Experimentation, Performance

Additional Key Words and Phrases: Content optimization, user preference, pairwise learning, bayesian model

1. INTRODUCTION

Recent years have witnessed a rapid growth of the Internet, which has become an important medium to deliver digital content to Web users instantaneously. Digital content publishers, including both portal websites, e.g. MSN [1] and Yahoo! [2], and homepages of news media, e.g. CNN [3] and the New York Times [4], have all started providing Internet users with Web content in a timely fashion via a wide range of content mod-

[1]http://msn.com/
[2]http://yahoo.com/
[3]http://cnn.com/
[4]http://nytimes.com/

DOI 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

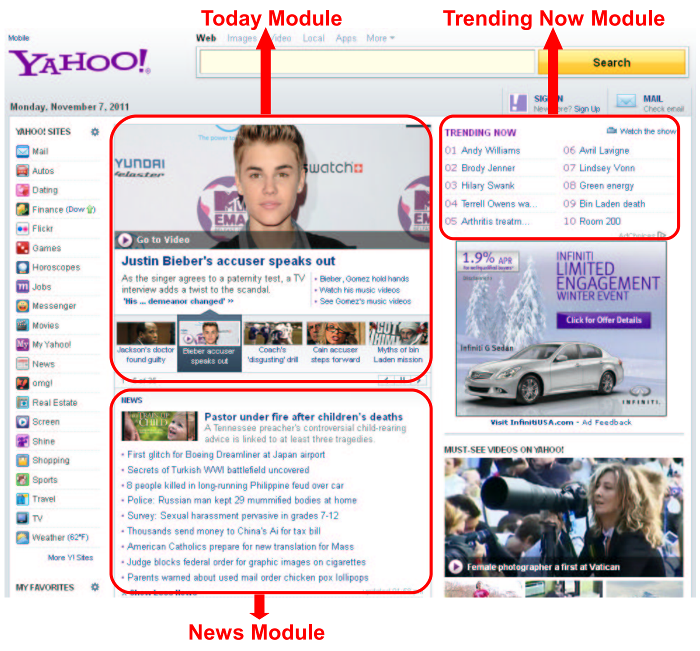


Fig. 1. A snapshot of Yahoo! front page. The page contains multiple recommendation modules such as the *Today module*, *Trending Now module* and *News module*.

ules. For example, as shown in Figure 1, the Yahoo! front page presents users with a number of content modules, including today's emerging events, news of various aspects, and trending queries from the search engine. However, Web users usually have short attention span while facing the explosion of Web content, often referred as *information overload*. Thus, it is necessary for those Web publishers to *optimize* their content by recommending content items that are most attractive to users for any given time in order to retain users to their portal sites at an ongoing basis.

In general, the process of Web portal content optimization consists of gathering information about portal website users, managing the content assets, analyzing current and past user interactive actions, and, based on the analyses, delivering the right content to each user at the right time. While above general process of content optimization looks similar to that of traditional personalized recommendation scenarios, it is not feasible to just simply adopt the conventional recommendation algorithms such as content-based filtering or collaborative filtering; the reason is because the Web portal content optimization problem has a couple of unique issues, such as extremely dynamic content pool and time-sensitive user preferences. Namely, in a Web portal, the content pool, e.g., the trending search queries, can be varying very frequently, and each content item may have a lifetime of only several hours. Therefore, most of the traditional personalized recommendation algorithm would suffer from a severe cold start problem. Moreover, user preferences over the content items may also vary quickly as users who come to Web portal are usually interested in timely topics, hence a specific user may look for different types of contents depending on the time of visit.

A key challenge to address above issues for content optimization is to devise *online learning* algorithms that can quickly learn what item a user may be interested for a given time, based on users' past interactions with the content items. The main problem that arises in devising such scheme is the explore-exploit dilemma, a well-studied problem in statistics and machine learning communities. That is, the content optimization system should explore the content pool enough to find out the best item for a given user at a given time, and simultaneously, sufficiently exploit the best item for users so that overall user satisfaction could be maximized. Recently, some variations of the *multi-armed bandit* schemes, which are online learning algorithms that are designed to solve above dilemma, have been successfully applied to optimizing the content offerings for Today Module on Yahoo! front page [Agarwal et al. 2008] [5]. The main idea is to apply the ϵ-greedy strategy; namely, constantly explore to estimate the attractiveness of each content item from a small fraction of user traffic in which the items are shown in random orders, then exploit the best item learned from the exploration at a given time for the rest of the user traffic. For the personalization, of course, several features that reflect user characteristics were used so that the "best" in above procedure can be found for each user. In results, when the click-through rate (CTR) of an item is used as a proxy for the attractiveness, above scheme has significantly improved the CTR of the Today Module.

Despite the practical success of above method, the approach is somewhat limited as it is more appropriate for the setting in which the user traffic size is large and the content pool size is relatively small, like the popular Today Module. In such cases, obtaining a reliable CTR estimate for each content item from the random exploration is realistic since there will be a large number of views and clicks for any single item. However, in applications like Trending Now or News Module on the Yahoo! front page, in which users' interactions via clicks are relatively lower while the content pool size can be larger, simply trying to estimate CTR of each item by treating every logged views and clicks equally could cause more problems than solutions; that is, recommending content items based on inaccurate estimates of CTRs would not make the recommendation quality any better. In order to partly remedy such low-clicks and views problem, recently [Dong et al. 2011] has proposed a method to effectively interpret users' actions, i.e., views and clicks, such that the estimates of CTRs of contents could become more reliable when using the data that have been appropriately interpreted.

While the online learning framework and the action interpretation mentioned above could alleviate some of the weaknesses associated with traditional recommendation techniques for the content optimization problems, there still are a few critical problems that are unique to the content optimization at portal websites and have not been studied in the literature. First, as shown in Figure 1, each content module usually presents a couple of content items in a limited space, thus, users' actions can not only represent their interests on a single item but also strongly indicate users' preferences among displayed items. However, most of previous online learning schemes that deal with the explore-exploit challenge only employs dedicated pointwise models, thus overlooking some invaluable user preferences between content items implied by user actions. Second, the personalization of content optimization typically happens by segmenting users into several smaller groups based on user features, and the sparse samples problem mentioned above not only would persist, but would become worse. That is, since the life time of an item could be very short, a pointwise model would never be able to collect enough samples to obtain a reliable CTR estimate for the item before it disappears in the content pool. Such unreliable estimate of CTR would result in unsatisfying recommendation.

In this paper, to address above challenges, we propose new dynamic pairwise learning methodologies for devising an online learning algorithm in the context of Web portal content optimization that deals with the explore-exploit dilemma. In particular, we will first take a deep look on user action information and explore how to extract

[5]See http://visualize.yahoo.com/core/ for concrete examples.

preferences between content item pairs accordingly. Then, we introduce two specific dynamic pairwise learning algorithms that estimate the attractiveness score for each item from the extracted preferences during the exploration phase: the first algorithm applies a graph-based method, while the other introduces a general Bayesian modeling process. These two algorithms are dynamic in the sense that they both rely on users' more recent preferences rather than old ones. We examine the effectiveness of pairwise learning approaches by conducting experiments on two large-scale data sets from a commercial Web portal. The results demonstrate significant improvements of pairwise methodologies in terms of the precision metric over the baseline pointwise model approach. Moreover, further analysis shows that pairwise learning approaches can be more beneficial for the sparse learning samples caused by personalization than the pointwise models. Finally, to complement the empirical findings, we provide a few theoretical insights that further justify the advantages of pairwise learning approaches over pointwise ones. To summarize, our specific contributions include:

— A general pairwise learning methodology that exploits dynamic user preferences for online learning in the Web content optimization system. To the best of our knowledge, this work is an early attempt to introduce pairwise learning methodology for resolving the explore-exploit dilemma prevalent in the Web content optimization.
— Two specified pairwise learning algorithms, one graph-based and the other based on Bayesian modeling, for Web content optimization using preferences.
— Analyses of the effects of important factors, including dynamic user preferences and user action positions, for pairwise online recommendation.
— Theoretical insights for justifying that pairwise learning methods can benefit online learning more in recommender systems compared with pointwise models.

The rest of this paper is organized as follows. Section 2 reviews the literature in pairwise learning and content optimization in recommender systems. Section 3 introduces our specific online learning framework for recommendation and points out the critical challenges for achieving better content optimization. To address these challenges, Section 4 proposes pairwise learning approaches and introduces two specified algorithms. A large-scale evaluation and discussion based on real-world data collected from a commercial portal website are presented in Section 5. To complement our empirical findings, we provide a few theoretical insights about the advantages of pairwise approaches in Section 6. Finally, we conclude the paper and point out future work in Section 7.

2. RELATED WORK

2.1. Pairwise Learning

Considerable research on pairwise learning has been conducted in the context of learning a ranking function for search applications, so-called learning-to-rank. While there are vast amount of papers on this topic, some representative work include RankBoost, [Freund et al. 1998], RankSVM [Joachims 2002], RankNet [Burges et al. 2005], and GBRank [Zheng et al. 2007]. For a more comprehensive reference, one can see [Liu 2009].

As mentioned in the Introduction, this paper is an early attempt to introduce pairwise learning methodology for dealing with the explore-exploit dilemma in Web content optimization. Although it looks similar, our proposed methodology is still quite different from pairwise learning-to-rank schemes in terms of several aspects of problem formulation:

• *Less rich features:* For the learning-to-rank problems, content items to rank for each search query are usually represented as a rich set of features, e.g., TF-IDF or BM25 from information retrieval. Thus, the goal of the learning-to-rank is to obtain a mapping function from the feature space into the ranking scores, and it is usually solved by a discriminative learning process; however, since the query, i.e., the user preference, is mostly implicit in Web portal content optimization, there are not so rich features yet to describe content items as in the learning-to-rank problems.

- *Existence of temporal dynamics of the user preferences:* In learning-to-rank for search, the user preferences are usually assumed to be stationary over time since the relevance between any pair of query and document does not change very frequently. However, user preferences under Web portal content optimization may yield more dynamics as Web users' interests in emerging information on Web portals can change very often. Thus, reflecting such time-varying user preferences to the recommendation algorithm is one of the key challenges in content optimization.
- *Difference in learning process:* Due to the essential difference in terms of necessity to update the model in real time, learning-to-rank and Web content optimization yield quite different pairwise learning processes. In particular, the pairwise learning-to-rank leverage user preferences to build the pairwise objective function, but its ranking function still uses an individual content item's features as input to compute the ranking score of this item. And, it takes an offline optimization process to obtain the parameters of the ranking function. Nevertheless, for our online pairwise content optimization, all extracted user preferences are used directly to infer the attractiveness score of each content item. As a result, the recommendation models can be updated in an online process such that the effects of user preferences can be reflected in the recommender system in real time.

2.2. Personalized Recommendation and Content Optimization

As mentioned in the Introduction, the Web content optimization problem is in general defined as the problem of selecting and presenting content items that may be most relevant to a user at a given time who intends to browse the Web for information. Depending on different applications and settings, there are many variants of the problem such as selecting articles published on portal websites [Agarwal et al. 2008; Agarwal et al. 2010], news personalization [Das et al. 2007; Li et al. 2010], computational advertising [Broder 2008; Richardson et al. 2007] and many others. Since the Web content optimization problem is a variation of personalized recommendation problems, we summarize below some previous work in recommender systems that are relevant to our work.

For general personalized recommendation problems, there are two major classes of standard approaches: content-based filtering and collaborative filtering. The former one reflects the scenario where a recommender system monitors a document stream and pushes documents that match a user profile to the corresponding user. Then, the filtering system uses explicit relevance feedback from users to update the user's profile using relevance feedback retrieval models [Zhang and Koren 2007; Yu et al. 2004; Zigoris and Zhang 2006] or machine learning algorithms [Yang et al. 2005; Gabrilovich et al. 2004]. Collaborative filtering goes beyond merely using document content but takes advantage of information from other users with similar tastes and preferences [Konstan et al. 1997; Jin et al. 2004; Hofmann and Puzicha 1999; Herlocker et al. 1999]. Some of previous studies [Melville et al. 2002; Wang et al. 2006] have tried to combine both techniques to build more effective recommender systems.

More recently, implicit user-grouping by latent-group or factor models[Srebro et al. 2005; Koren 2008], matrix completion [Candes and Recht 2008], and Bayesian approaches [Stern et al. 09] also have become popular for general personalized recommendation problems. Furthermore, in the similar spirit as in the pairwise learning-to-rank algorithm for search, using pairwise preferences for devising personalized recommendation algorithm also has been used for recommender systems, e.g., [Rendle et al. 2009; Takács and Tikk 2012; Yang et al. 2011; Kanagal et al. 2012]. To address the problem of sparse learning samples in recommender system, some previous studies introduced factorized parametrized models which can make estimation with a small sample size. For example, [Rendle et al. 2010] proposed a Markov Chain model which factorizes the transition matrix such that transitions can be estimated with little or even no observations. In addition, [Koren 2009; Xiong et al. 2010; Kanagal et al. 2012] have considered incorporating some temporal variations of user preferences in the recommender system.

There also has been many previous studies that try to explicitly build some user features that can reflect user preferences for the personalized recommendation task. For instance, many studies proposed building user profiles to support personalization in the recommender system. Billsus et al. [Billsus and Pazzani 2007] created user profiles for adaptive personalization in the context of mobile content access. Ahn et al. built a news recommender system, YourNews [Ahn et al. 2007], which allows users to customize their interest profiles through a user model interface. In addition, a personalized service may not exactly serve the individual user. The content of the portal website can be tailored for a pre-defined categories of audiences, based on offline research and conjoint analysis. In very early studies [Wind 1978], homogeneous groups of consumers are entailed by the use of a priori segmentation. Chu et al. [Chu et al. 2009] recently proposed user behavior feature-based models for personalized services at individual and segmentation levels, respectively. Those personalized models are shown to outperform several demographic segmentation models.

While above various methods have seen success in practical recommender systems, our work in this paper has following differences with above.

- *Online vs. Batch:* Most of previous work in recommender systems, including the pairwise schemes mentioned above, are tailored for somewhat stationary pools of contents and users, hence are batch learning method. Therefore, when a new user or content frequently arrives and leaves the pool, just like in the Web content optimization problems, it is not clear how to quickly update the algorithms obtained from above methods. In contrast, we follow the online learning setting in [Agarwal et al. 2008] to deal with such cold start problem and the explore-exploit dilemma, and devise pairwise learning method in that setting which has not been considered before.
- *Much faster temporal dynamics of user preferences:* The schemes that deal with temporal variations of user preferences mentioned above were mainly for capturing the long-term variations of user preferences. However, in the Web content optimization problems, the time scale of user preference variations is much finer, sometimes in minutes. Thus, we try to incorporate such fast varying user preferences in the content optimization system.
- *Sparser learning samples:* The user profiling techniques mentioned above are suitable when there are enough learning samples so that the model for the high-dimensional space could be obtained reliably. However, there are many applications in which such abundance of learning data is not available. Our proposed methods try to combat the sparse learning data problem by considering the information that could be obtained by the pairwise preferences inferred from the users' feedback on the whole list of content items.

In summary, our proposed methods have several similarities with previous work in search ranking and personalized recommendation, but also have significant differences as described above. In the next section, we will describe our scheme more in detail.

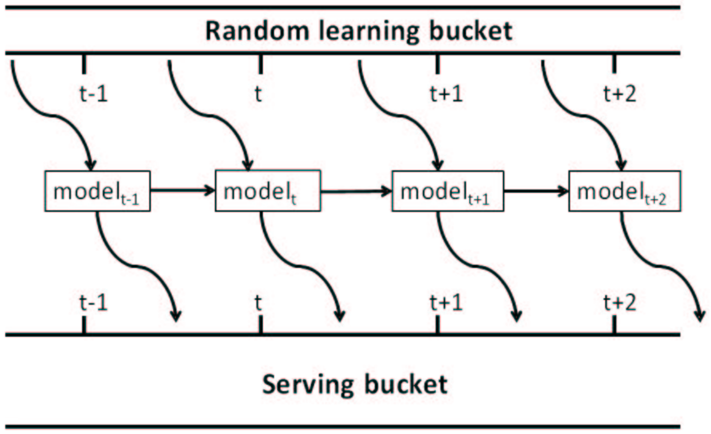


Fig. 2. Online learning flowchart: A random learning bucket is used for exploration purpose. At the end of each time interval, the model for each candidate item is updated based on the users' clicks and views from the random learning bucket during this time interval. In the next time interval, the updated model is applied to the corresponding candidate item in the serving bucket. In this way, all the candidate items are displayed by ranking scores (computed by their corresponding updated models) in the serving bucket.

3. ONLINE LEARNING FRAMEWORK FOR CONTENT OPTIMIZATION

As we target recommendation at content modules on web portals, our goal is to optimize content recommendation such that a certain user engagement metric, such as overall click-through rate (CTR), is maximized. For a pool of candidate items, human editors can be employed to manually rank the candidate items according to content attractiveness and users' interests and then recommend top ranked items to users. However, it requires expensive human effort and cannot guarantee that the most attractive and relevant items are recommended to users due to the interest gap between editors and Web users. Therefore, we attempt to design a recommender system that achieves content optimization by automatically estimating candidate items' attractiveness and relevance to users' interests. To achieve this goal, we currently employ a state-of-the-art online learning framework, which has three critical characteristics: *online learning*, *pointwise model*, and *personalization*. In the rest of this section, we will discuss these three aspects in details. Then, we will point out the critical challenges of this framework, which will be addressed by our new dynamic pairwise learning methodology in the next section

3.1. Online Learning

To attract more users to browse and click content displayed on the portal modules, an online learning methodology is necessary as it enables us to model users' behavior (i.e. clicks and views) on the content modules as implicit feedback and to adjust the recommendation model in real time (or almost real time) accordingly.

To enable online learning in our recommender system, we apply a *parallel-serving-buckets* framework, where '*bucket*' means a part of user visiting traffic on Web portal. Figure 2 illustrates the flowchart of this online learning framework. Specifically, there are two parallel buckets serving simultaneously in the system: a *random learning bucket* and a *serving bucket*. When a user visits the Web portal, this visit event will be assigned into the traffic of *random learning bucket* with a certain probability, otherwise it will be assigned into the visiting traffic of *serving bucket*.

Within the *random learning bucket*, a certain number of items are randomly picked from a pool of candidates to serve as recommended items for each user visit. In our system, we limit the *random learning bucket* to occupy only a small fraction of the whole traffic, i.e. the probability that a user visit falls into this bucket is very small. Although randomly serving candidate items is obviously not the optimal recommending strategy, this *random learning bucket* can benefit online learning in another way. In particular, since each item from the candidate pool has equal chances to be served to users in the *random learning bucket*, we can obtain an unbiased estimate of its CTR based on user feedback in this bucket. Such unbiased CTR estimates can be further used as strong signals of users' interests on the corresponding items to improve the recommendation model in the *serving bucket*.

In our *parallel-serving-buckets* approach, as shown in Figure 2, all the models in both buckets are updated simultaneously every 5 minutes (i.e., the time interval $[t, t+1]$ equals 5 minutes in Figure 2). In general, within the *serving bucket*, the recommendation model, at a certain time point $t+1$, is updated based on the observations, i.e. unbiased estimated CTRs, from the *random learning bucket* during the time interval $[t, t+1]$. The updated model is then applied to estimate attractiveness scores of candidate items in *serving bucket* during $[t+1, t+2]$, and the items are displayed by the scores in the descending order.

3.2. Pointwise Model

Pointwise model is the most straightforward approach adaptable to the online learning framework. In particular, each candidate item in the recommender system uses a dedicated model individually to estimate its attractiveness score. In our online learning framework, as real-time user feedbacks, which imply strong signals of items' attractiveness to users, is available in the *random learning bucket*, the attractiveness score of an item can be estimated by its unbiased CTR obtained from the *random learning bucket*.

To build effective pointwise models, we employ an *Estimated Most Popular* (EMP) model [Agarwal et al. 2009]. Assume during the time interval $[t, t+1]$, an item was displayed to users $n_{[t,t+1]}$ times, which resulted in $c_{[t,t+1]}$ clicks, and assume this item's CTR estimation is p_t predicted by its previous model at time t; then, for the model of this item at time $t+1$, the CTR estimation of this item is updated as

$$p_{t+1} = \frac{\gamma_t p_t + c_{[t,t+1]}}{\gamma_t + n_{[t,t+1]}}, \tag{1}$$

where γ_t is the sample size of the prior belief.

The intuition in Equation (1) is that, given its prior probability p_t, the CTR estimation is updated according to new observations of clicks and views during time interval $[t, t+1]$. The sample size γ_t is used to control the balance between the prior probability and new observations. The higher the value of γ_t, the more confidence we have on the prior probability. If the value of γ_t is set lower, the CTR estimation relies more on the new observations. More details of EMP can be found in [Agarwal et al. 2009]. In this paper, the EMP approach is regarded as the baseline, which is to be compared with new proposed dynamic pairwise learning methods as introduced in the next section.

3.3. Personalization

Personalization has become more important for recommender systems as it provides users with a customized experience of highly attractive and relevant content, so as to enhance user engagement, conversions, and long-term loyalty. To achieve personalization for content optimization, our online learning framework employs a *user segmentation* approach, in which homogeneous groups of users are entailed by a priori segmentation [Wind 1978]. Each segment has its exclusive online learning and serving process. In other words, within one user segment, the recommendation model for each item is learned based on clicks and views only from the users belonging to this user segment, and the serving results using pointwise models are also only applicable to the users belonging to the same group. There are a few other kinds of personalization approaches for recommender systems; however, the user segmentation approach yields advantages in terms of both simplicity and reliability, especially for real-world commercial recommender systems.

To obtain this user segmentation, we generalize a set of user features and then apply clustering techniques to group users based on extracted features. We collect two major categories of user features that is available to the portal website: 1) *Explicit features*: the personal information explicitly requested by the portal website, such as age, gender, occupation, preferences, etc. 2) *Implicit features*: various types of user behavior tracked by the portal website, such as browsing and purchasing patterns of users on the pages within this website, etc. In our work, each user is represented as a vector of features, and, we empirically generate 10 user segments by using K-means, which is a well-known clustering technique. Due to the limited space, we will not present the specific clustering process in this paper.

3.4. Challenges

Although our online learning framework combined with pointwise models and personalization has shown effectiveness for the content optimization on portal websites, there still exists a couple of critical challenges:
1) *How to take advantage of rich preference information?*
Currently, our online learning framework leverages users actions, in the form of clicks or views on each item, to build the corresponding pointwise recommendation model. Although such user action information can provide explicit signals of users' recent interests, the dedicated pointwise model only uses such information to improve the model of the specific item that user clicks or views; it overlooks invaluable user preferences between item pairs, and may result in the problem of inconsistency caused by interaction of displayed items, more insights of which will be provided in Section 6.
2) *How to deal with the problem of sparse learning samples?*
Some content modules on portal websites may not receive a large amount of user clicks, such that they suffer the problem of sparse learning samples for training the recommendation model. This challenge becomes more severe when applying user segmentation based personalization. Our experiments in Section 5 and analysis in Section 6 will show that the performance of pointwise models decreases drastically when there is not ample learning samples.
3) *How to deal with the unsatisfying recommendation for new emerging content items?*
As dedicated pointwise models cannot collect enough learning samples in a short time, they fail to achieve accurate estimation of the attractiveness of new emerging content items.

To address these challenges, we will propose the new pairwise learning methodology for Web content optimization in the next section. The pairwise learning methods can take much advantage of rich dynamic user preferences to estimate content item's attractiveness and relevance to users' interests. The effectiveness of this new methodology will be demonstrated via large-scale evaluations in Section 5, together with more analysis on the effects of various critical factors in this methodology. More theoretical insights justifying the advantages of new pairwise methodology will be provided in Section 6.

4. PAIRWISE LEARNING FOR ONLINE CONTENT RECOMMENDATION

Content modules on portal website usually present a few content items in a small space, thus, user actions, in the form of clicks and views on those content items, can indicate not only the relevance signal for each clicked item but also the dynamic preferences between different items in the same module. Those preferences contain invaluable information for comparing content items' attractiveness in a timely fashion beyond the relevance on individual content items. Hence, when there are inadequate user clicks for accurate estimation of single item's attractiveness, user preferences can still supply much information about relative relevance judgments to estimate items' attractiveness for recommendation instantaneously. Consequently, we argue that dynamic preference information extracted from user actions can effectively reduce the effects of sparse learning samples. In this section, we first explore how to extract preference information based on user actions. Then, we will propose two specific algorithms for exploring dynamic preference to achieve Web content optimization. The first one is a graph-based method, followed by a more formalized Bayesian generative algorithm.

4.1. Preference Extraction from User Actions

For a specific content module, the log of user actions can be represented as a sequence of sessions, and each session is defined as

$$S(u,t) = \langle u; t; (d_1, d_2, \cdots, d_P); C\rangle$$

where $S(u,t)$ denotes the session for user u's actions on time t; $(d_1, d_2, \cdots, d_P)$ denotes the set of content items presented to user u at time t while the index of each item represents its display position; and C represents the set of positions that are clicked by the user.

By examining each user session, we can extract preference information as follows:

> *Given the content item list $(d_1, d_2, \cdots, d_P)$ and a set C containing the positions of clicked-on items, we extract preference examples, $d_i \prec d_j$, for all pairs $j \in C$ and $i \notin C$.*

Note that, many previous studies have investigated position-sensitive preference extraction, e.g. '*skip-above*' and '*skip-next*' strategies [Joachims et al. 2005], in the context of ranking for search. However, the content modules on portal websites are quite different from a ranked list of search results. First, unlike a long ranked list of search results, one content module usually occupies a small area of the portal webpage, the content items in which could be all browsed with quick scan. Second, the organization of content items is usually not the same as the ranked list. As shown in Figure 1, content modules could have more complex organization for presentation. In this paper, we will apply the preference extraction method introduced above by default. In the following experiments, we will show that position-sensitive preference extraction, especially '*skip-above*' and '*skip-next*' strategies, does not bring significant improvement in recommendation. As position-sensitive preference extraction is not the focus of this work, we leave this topic as future work.

4.2. Graph-based Pairwise Learning

The general idea of our graph-based pairwise learning method is to build a weighted, directed preference graph $G = \langle V, E, W\rangle$ from user feedback. The nodes ($v \in V$) are candidate content items, while the edges ($e \in E$) denote the preferences. For example, the directed edge from v_i to v_j represents the preference $v_i \prec v_j$, and the weight of an edge e, denoted as $W(e)$, measures the strength of the preference. Note that, to introduce personalization, we could build respective preference graph for different user segments. For example, for one particular user segment $\mathcal{C}$, the corresponding preference graph can be represented as $G_{\mathcal{C}} = \langle V, E, W_{\mathcal{C}} \rangle$, where candidate content items V are the same across all user segments, while the preference weights W are different in various segments.

In Web portal content optimization, both content items and user preferences change over time. To capture the dynamics of content items and user preferences, we denote the directed preference graph at time slot t as $G_{\mathcal{C}}^{(t)} = \langle V^{(t)}, E^{(t)}, W_{\mathcal{C}}^{(t)} \rangle$, where $V^{(t)}$ denotes the set of candidate content items at time t; $W^{(t)}$ represents the corresponding matrix of dynamic weights for each of edges, i.e. for one edge $E_{\langle ij \rangle}^{(t)} = \langle v_i, v_j \rangle$, the weight of it, $W_{\mathcal{C}\langle ij \rangle}^{(t)}$, equals the dynamic user preference on item v_j over item v_i at time t. In particular, we update such dynamic preference as

$$W_{\mathcal{C}\langle ij \rangle}^{(t)} = \rho W_{\mathcal{C}\langle ij \rangle}^{(t-1)} + \delta_{\mathcal{C}[t-1,t]} \tag{2}$$

where $\delta_{\mathcal{C}[t-1,t]}$ denotes the number of times item v_j is preferred over item v_i by any user in the segment $\mathcal{C}$ during $[t-1, t]$; and, ρ is the time-decay factor that is used to discount the preferences which happened long time ago. We can tune this parameter based on performance on a validation dataset.

After building this dynamic preference graph, we can employ a PageRank-like algorithm to compute the attractiveness scores for each of items at time t, the details of which are shown as Algorithm 1.

Algorithm 1 Graph-based Pairwise Algorithms for Content Recommendation

Input: $G_{\mathcal{C}}^{(t)} = \langle V^{(t)}, E^{(t)}, W_{\mathcal{C}}^{(t)} \rangle$: directed preference graph at time t, where $V^{(t)}$ denotes item set, $E^{(t)}$ is edge set, and $W_{\mathcal{C}}^{(t)}$ denotes the matrix of edges' weights with respect to user segment $\mathcal{C}$ ($W_{\mathcal{C}\langle ij \rangle}^{(t)}$ is the weight of the edge from item i to item j);
ρ: the damping factor, and $0 < \rho < 1$; we set ρ as 0.85 in our experiments.

Output: $S_{\mathcal{C}}^{(t)}$: the vector of recommendation scores for content items at time t w.r.t. user segment $\mathcal{C}$, i.e. $S_{\mathcal{C}\langle i \rangle}^{(t)}$ denotes the score of item $V_i^{(t)}$.

Algorithms:

```
1  Initialize $S_{\mathcal{C}}^{(t)}$ with a uniform probability distribution, denoted as $S_{\mathcal{C}}^{(t)}(0)$,
   where $S_{\mathcal{C}\langle i \rangle}^{(t)}(0) = \frac{1}{|V|}$, $i = 1, \cdots, |V|$;
2  Normalize $W_{\mathcal{C}}^{(t)}$ into the transition probability matrix $\mathbf{W}_{\mathcal{C}}$, i.e. column-
   stochastic with no column consisting of just zeros;
3  $n = 0$;
4  do:
5      $S_{\mathcal{C}}^{(t)}(n+1) = \big(\rho \mathbf{W}_{\mathcal{C}} + \frac{1-\rho}{|V|}\mathbf{E}\big) S_{\mathcal{C}}^{(t)}(n)$
       ($0 < \rho < 1$; $\mathbf{E}$ is matrix of all ones.);
6  while $|S_{\mathcal{C}}^{(t)}(n+1) - S_{\mathcal{C}}^{(t)}(n)| < \epsilon$;
7  $S_{\mathcal{C}}^{(t)} = S_{\mathcal{C}}^{(t)}(n+1)$;
```

This graph-based algorithm is a straightforward method for dynamic pairwise learning. In the following of this section, we propose a formalized probabilistic modeling algorithm for specifying pairwise learning.

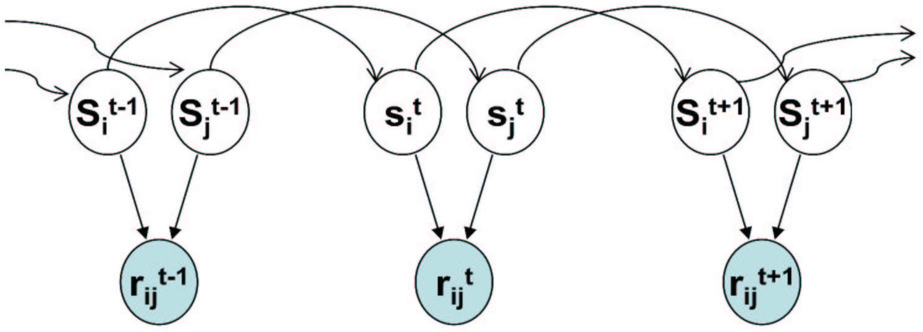


Fig. 3. The graphical model for Bayesian Hidden Score (BHS) model.

4.3. Bayesian Pairwise Learning

In this section, we propose a probabilistic model, Bayesian Hidden Score (BHS) model, for dynamic pairwise learning. Some previous studies [Rendle et al. 2009; Takács and Tikk 2012] have applied Bayesian methods to obtain personalized ranking and recommendation. However, they are tailored for somewhat stationary pools of contents and users, hence are batch learning method. Therefore, when a new user or content frequently arrives and leaves the pool, it is not clear how to quickly update the model. In contrast, our new BHS model devise pairwise learning with following the online learning setting in [Agarwal et al. 2008], which can effectively deal with the problems in terms of temporal dynamics of user preferences as well as sparser learning samples.

We let $r_{ij;c}$ denote the observed preference strength between the ith and the jth item w.r.t. user segment c, such that, under the context of user segment c

$$r_{ij;c} \begin{cases} > 0 & \text{if } i\text{th item is preferred over } j\text{th item,} \\ = 0 & \text{if } i\text{th and } j\text{th instance are equally preferred,} \\ < 0 & \text{if } j\text{th item is preferred over } i\text{th item.} \end{cases} \tag{3}$$

Note that, $r_{ij;c}$ can be specialized in various ways; we can define $r_{ij;c}$ as the number of sessions where users choose to click the ith item instead of the jth item, which is similar to the graph-based method; we can also use the normalized edge weight in the graph to define $r_{ij;c}$. We further let $r^t_{ij;c}$ denote the preference value between the ith and the jth item at time t.

We make two basic assumptions for BHS. First, each item i is associated with a hidden score $s^t_{i;c}$ that denotes its intrinsic attractiveness in user segment c at time t; second, the preference between item i and item j, $r^t_{ij;c}$, is generated conditioning on $s^t_{i;c}$ and $s^t_{j;c}$, i.e.,

$$r^t_{ij;c} \sim p(r^t_{ij;c} | s^t_{i;c}, s^t_{j;c}, \alpha), \tag{4}$$

where α is the parameter for the condition distribution.

Furthermore, we assume that each hidden score $s^t_{i;c}$ at time t is generated conditioned on the the hidden score $s^{t-1}_{i;c}$ from the time t-1 such that

$$s^t_{i;c} \sim p(s^t_{i;c} | s^{t-1}_{i;c}, \lambda), \tag{5}$$

where λ is a parameter for the condition distribution.

Figure 3 shows the graphical model for BHS. BHS is a general model that applies to different online learning applications with different data distribution assumptions.

In this study we focus on the most popular distribution, normal distribution, to derive stochastic gradient descent based algorithm for BHS. However, the derivation can be extended to the other popular distributions.

The likelihood function of BHS can be written as follows,

$$p(D; \alpha, \lambda) = \prod_{t} \prod_{r^t_{ij;c} \in D} (p(r^t_{ij;c} | s^t_{i;c}, s^t_{j;c}, \alpha) p(s^t_{i;c} | s^{t-1}_{i;c} \lambda) p(s^t_{j;c} | s^{t-1}_{j;c}, \lambda)) \tag{6}$$

where D denote the observed pairwise preferences.

We assume the following normal distributions,

$$r^t_{ij;c} | s^t_{i;c}, s^t_{j;c}, \alpha \sim N(s^t_{i;c} - s^t_{j;c}, \alpha) \tag{7}$$

$$s^t_{i;c} | s^{t-1}_{i;c}, \lambda \sim N(s^{t-1}_{i;c}, \lambda) \tag{8}$$

Substituting the density functions of (7) and (8) into (6), taking negative log, and dropping the constant terms, we obtain the following negative log-likelihood function,

$$\mathbb{O} = \sum_{t} \sum_{r^t_{ij;c} \in D} (||r^t_{ij;c} - (s^t_{i;c} - s^t_{j;c})||^2 + \gamma(||s^t_{i;c} - s^{t-1}_{i;c}||^2 + ||s^t_{j;c} - s^{t-1}_{j;c}||^2), \tag{9}$$

where $\gamma = \frac{\alpha}{\lambda}$.

Our main task is learning the hidden scores s to minimize the objective function in (9). The parameter γ can be viewed as the weight of prior distribution. In practice, we found that using cross-validation to pre-determine γ is more efficient and effective than directly learning γ. Therefore, our final task is,

$$\min_{s} \sum_{t} \sum_{r^t_{ij;c} \in D} (||r^t_{ij;c} - (s^t_{i;c} - s^t_{j;c})||^2 + \gamma(||s^t_{i;c} - s^{t-1}_{j;c}||^2 + ||s^t_{i;c} - s^{t-1}_{j;c}||^2), \tag{10}$$

We propose a stochastic gradient descent based algorithm to solve the optimization in (10). The algorithm loops over all the observations r^t_{ij} and updates the hidden scorse by moving in the direction defined by negative gradient. The algorithm is summarize in Algorithm 2. BHS algorithm has two main advantages: 1) it can be easily parallelized to handle large scale data; and 2) it yields a much faster converging speed. Based on experiments in the next section, we can also find that BHS can achieve better performance than graph-based pairwise learning.

Algorithm 2 Bayesian Hidden Score Algorithm

Input: Observed pairwise preferences, $\{r^t_{ij;c}\}$, learning rate τ
Output: Hidden scores for all items, $\{s^t_{i;c}\}$

Algorithms:

```
1  Initialize $\{s^0_{i;c}\}$.
2  do:
3      for each $t$:
3          for each $r^t_{ij;c}$:
4              $s^t_{i;c} \leftarrow s^t_{i;c} - \tau \frac{\partial \mathbb{O}}{\partial s^t_{i;c}}$;
5              $s^t_{j;c} \leftarrow s^t_{j;c} - \tau \frac{\partial \mathbb{O}}{\partial s^t_{j;c}}$;
6  Until Converges.
```

5. EXPERIMENT

In this section, we design experiments to validate that our proposed pairwise learning approaches can improve the performance of Web content optimization. We first describe the data sets collected from a commercial portal website and evaluation metrics in Section 5.1. Then, we report the results of large-scale evaluations for our proposed pairwise learning approaches compared with baselines in Section 5.2.

5.1. Experimental Setup

5.1.1. Data Set. To validate our proposed pairwise learning approaches, we conduct experiments on the data from two real-world content recommendation modules, the Trending Now module and News module on the Yahoo! portal website (as shown in Figure 1). For the Trending Now module, we collected events in terms of *views* and *clicks* from its *random learning bucket* during ten days from April 1st, 2011 to April 10th, 2011. And, for the News module, we similarly collected events in terms of *views* and *clicks* from its *random learning bucket* during fourteen days from May 1st, 2011 to May 14th, 2011. For each of these two modules, the pool of candidate items may change multiple times during each day. To protect privacy, all the users are anonymized in this data set.

As introduced in Section 3.1, in the *random learning bucket*, candidate content items, i.e. queries in Trending Now or news articles in News, are randomly selected and they are displayed at all positions with equal chance. An event records a user's action on the served content items, which is either "*view*" or "*click*". More specifically, we represent each event e as a set of tuples:

$$e = \langle u, t, p, i_p, a \rangle,\ p = 1, 2, \ldots, 10$$

where u denotes the user; t represents the time stamp for this event; i is the served content item; p denotes the position at which the content item i is displayed (Note that there are ten positions on both Trending Now and News module, but they are organized differently); a represents the action which is either *view* or *click*.

For the Trending Now data set, there are totally hundreds of millions ($\sim\ 10^8$) of events with multiple millions of unique users; while for the News data set, there are in total tens of millions ($\sim\ 10^7$) of events with multiple millions of unique users. The size of Trending Now data set is about five times larger than that of News data set. And, the average sizes of candidate content pools per five minutes for both data sets are comparable.

5.1.2. Evaluation Metrics. To evaluate the performance of our online recommendation models, we simulate the online learning procedure as illustrated in Figure 2. In particular, all pointwise models at time t are updated based on the click/view samples during the latest 5-minute interval $[t-1, t]$ in the *random learning bucket*; the updated models are then applied to compute relevance scores for the candidate items and recommend those with higher scores during the next time interval $[t, t+1]$. Similarly, the pairwise models at time t are updated by extracting new preference evidences from the click/view samples during $[t-1, t]$ in the *random learning bucket*; the updated pairwise models are then applied to recommend the candidate items during $[t, t+1]$. For the clicks that actually happened during $[t, t+1]$ in the *random learning bucket*, the evaluation metric is computed by comparing these actual clicks with the predicted ranking. Intuitively, a good modeling approach should lead to high correlation between the actual clicks and the predicted ranking.

More specifically, for those clicks that actually happened at Position 1 in the *random learning bucket*, we define $precision_i$ as the number of the clicked items that are ranked at Position from 1 to i according to the model prediction. Table I illustrates two

Table I. An illustrative example for evaluation metric (precision) computation. For an actual event in the random learning bucket, Item 1 was ranked at Position 1 and clicked by the user. For Model 1: $precision_1$ = 1, $precision_2$ = 1, $precision_3$ = 1, and so on. For Model 2: $precision_1$ = 0, $precision_2$ = 0, $precision_3$ = 1, and so on. Model 1 is regarded as a better model as the clicked item is ranked higher by Model 1

actual ranking	predicted ranking by Model 1	predicted ranking by Model 2
1 (clicked)	**1**	2
2	5	3
3	4	**1**
4	3	5
5	2	4

examples for such precision computation. Note that the reason we only use the actual clicks at Position 1 for evaluation is that the clicks on other positions might need to be counted with more weight due to position bias, so it is more straightforward to use clicks at Position 1 for evaluation. To protect business-sensitive information, we report only relative precision, instead of precision itself.

In the following experiments, we evaluate the proposed pairwise learning methods compared with the baseline pointwise models on both Trending Now and News data sets. By following the online learning simulation procedure during the 10-day period for Trending Now module and 14-day period for News module, the overall precision values are computed respectively by aggregating the precision of recommendation in each of the 5-minute time intervals.

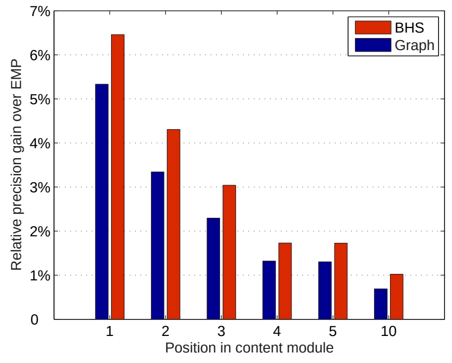
5.1.3. Compared Methods. To evaluate the pairwise learning methods for Web content optimization, we compare the performance of the following methods:

— **EMP** (baseline): In this method, we adopt the estimated most-popular model as introduced in Section 3.2.
— **EMP-seg**: In this method, we incorporate personalization in terms of user segmentation into EMP (Section 3.3), i.e. we train EMP models for different user segments separately.
— **Graph**: In this method, we use the graph-based pairwise approach and PageRank-like learning method, as described in Section 4.2.
— **Graph-seg**: In this method, we incorporate personalization in terms of user segmentation into graph-based pairwise approach, i.e. we learn pairwise models for different user segments separately.
— **BHS**: In this method, we use the Bayesian hidden modeling approach as introduced in Section 4.3.
— **BHS-seg**: In this method, we use the Bayesian hidden modeling approach with incorporated personalization in terms of user segmentation, i.e. we learn the Bayesian hidden models for different user segment separately.
— **PCS**: In the following experiment, we will compare our proposed methods with a state-of-the-art method, i.e. Personalized Click Shaping [Agarwal et al. 2012], which targets for personalized online recommendation without relying on pairwise learning.

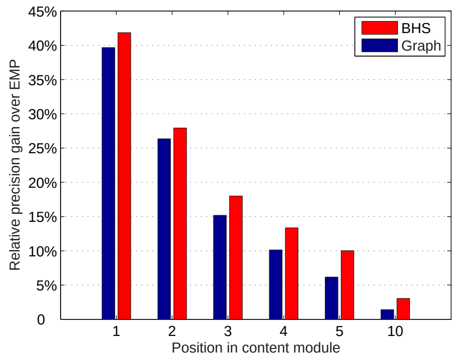
Note that, we applied the same user segmentation for **EMP-seg**, **Graph-seg**, and **BHS-seg**.

5.2. Experimental Results

5.2.1. Performance for Online Content Recommendation. We now compare the performance of our pairwise learning methods with other baselines for Web content optimization. We conduct experiments on both the data sets collected from Trending Now module and News module.

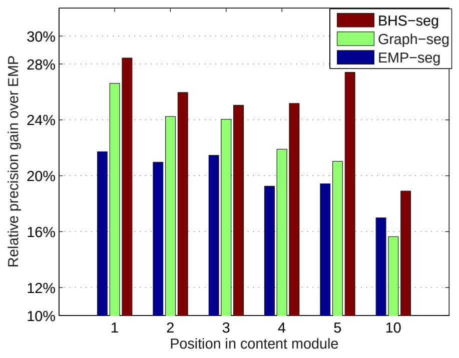


(a) Trending Now dataset

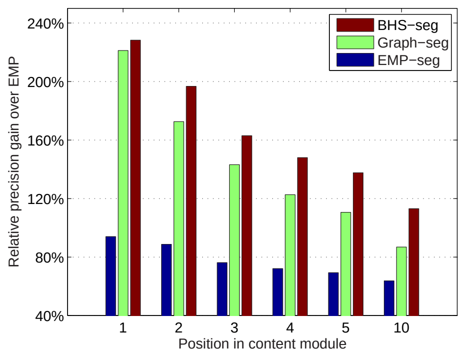


(b) News dataset

Fig. 4. Relative precision gain by pairwise methods (no personalization) over the baseline EMP.



(a) Trending Now dataset



(b) News dataset

Fig. 5. Relative precision gain by pairwise methods with personalization over the baseline EMP.

Figure 4(a) and 4(b) demonstrate the relative precision gain of those pairwise learning methods without personalization (i.e. **Graph** and **BHS**) compared with the baseline **EMP** model. These experiments are conducted on the Trending Now data set and News data set, respectively. From these figures, we can find that the pairwise methods, **Graph** and **Bayesian**, can outperform the baseline pointwise model **EMP** on both data sets. Furthermore, the Bayesian pairwise method, i.e. **Bayesian**, can reach much better performance than straightforward graph-based pairwise method, i.e. **Graph**.

Figure 5(a) and 5(b) demonstrate the relative precision gain of **EMP-seg** together with those pairwise learning methods with personalization (i.e. **Graph-seg** and **BHS-seg**) compared with the baseline **EMP** model. From these figures, we can find that, while personalization can bring significant performance gain for pointwise models, pairwise learning approaches can provide even more benefit to personalized online recommendation. In particular, after applying pairwise learning methods combined with personalization in terms of user segmentation, we can observe that **Graph-seg** achieves about 21% and 126% precision gain on Position 1 over **Graph** on the Trending Now data set and News data set, respectively. And, the **BHS-seg** achieves about 19% and 112% precision gain on Position 1 over **BHS** on Trending Now data set and News data set, respectively. These figures also illustrate that the formalized Bayesian model can reach better performance than the graph-based pairwise learning method.

Note that applying pairwise learning methods can improve recommendation performance on News data set much more than on Trending Now data set. We hypothesize one of the major reasons is that, the size of News data set ($\sim 10^7$) is much smaller than Trending Now data set ($\sim 10^8$); it suffers more from sparse learning samples, and pairwise learning methods can significantly reduce the effect of this problem. We also consider this the reason that pairwise methods can benefit personalized recommendation more, since each user segment has fewer learning samples than the whole data set. Using News data set as an example, as shown in Figure 4(b) and 5(b), the relative precision gain of **BHS-seg** over **EMP-seg**, about 69%, is a bit larger than the gain of **BHS** over **EMP**, about 42%. We can observe the similar results on Trending Now data set or using graph based method instead of BHS method.

Table II. Relative precision gain by BHS over the state-of-the-art online recommendation method.

Position	1	2	3	4	5	10
Relative Gain	2.32%	2.18%	1.89%	1.72%	1.57%	1.35%

In addition, we perform another experiment to compare the performance of our proposed pairwise learning methods with one state-of-the-art online recommendation method, i.e. PCS [Agarwal et al. 2012]. Table II reports the relative precision gain of **BHS-seg** over **PCS**. From this table, we can find that **BHS-seg** can outperform this personalized click shaping method. In particular, the relative precision gain by **BHS** over **PCS** is 2.32% at position 1 and 1.35% at position 10. The t-test result shows that the improvement are statistically significant (p-value ¡ 0.03). It implies that, though **PCS** has introduced very accurate modeling for personalization, it does not leverage the preference information between different content items, which plays a more important role in Web content optimization. Actually, in real commercial content recommendation system, 1% increment in terms of click precision is already a big improvement, which in practice can drive additional millions of clicks per day.

5.2.2. Robustness to Sparse Learning Samples. To verify that our new pairwise learning methodology is more robust to the problem of sparse learning samples, we conduct experiments to compare the precision decline against decreasing amount of learning samples for different recommendation models. Specifically, we reduce the amount of learning samples by randomly picking varying percentages of the whole data set, and evaluate the performance of different models on the reduced data set.

Figure 6 reports the relative precision decline for both pointwise model (**EMP-seg**) and pairwise learning method (**Graph-seg** and **BHS-seg**) against varying percentages of learning samples. From the figure, we can find that the performance of **EMP-seg** decreases drastically even when we remove only 20% learning samples, while **Graph-seg** and **BHS-seg** stays higher performance until we remove more than 80% learning samples. Based on these results, we can find that pairwise learning methods are more robust to sparse learning samples than pointwise models, which indicates that pairwise learning methods are a better choice for personalized recommendation since there are sparser samples for learning personalized models.

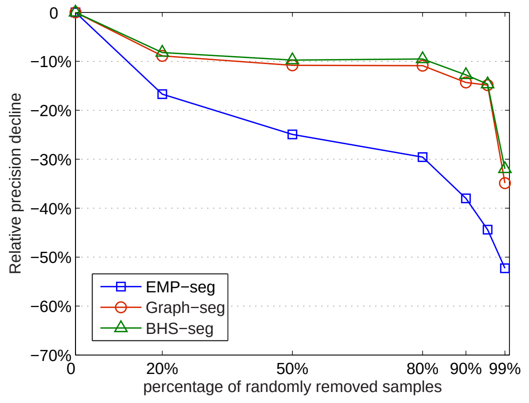


Fig. 6. Relative precision decline for both pointwise model and pairwise learning model against varying percentage of learning samples.

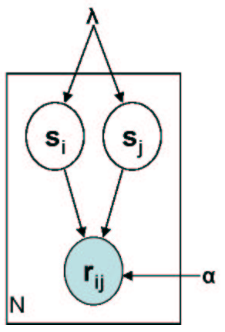
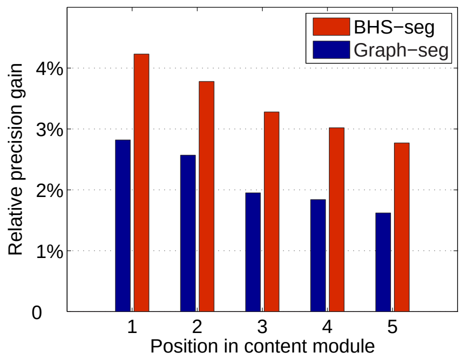


Fig. 7. The simplified Bayesian Hidden Score (BHS) model without considering temporal dynamics.

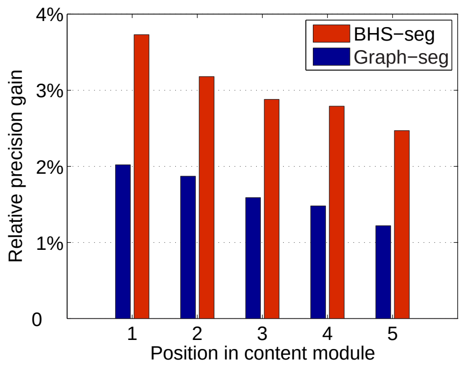
5.2.3. Effects of Dynamic Preferences. As mentioned before, we have modeled the temporal dynamics of user preferences in both graph-based and Bayesian pairwise learning algorithms. In this experiment, we show the influence of modeling temporal dynamics of user preferences on the performance of these two algorithms. In particular, we compare the performance of **Graph-seg** and **BHS-seg** modeling temporal dynamics of preferences to the performance of corresponding algorithms without modeling such temporal dynamics, respectively. With no consideration of temporal dynamics of user preference in **Graph-seg**, we can merely set $\rho = 1$ in Eq. 2, while in **BHS-seg**, we can employ a simplified BHS model, as shown in Figure 7.

Figure 8 demonstrates relative precision gain of **Graph-seg** and **BHS-seg** modeling temporal dynamics of user preferences compared with the same algorithms without modeling such temporal dynamics over two dataset, respectively. From the figure, we can find that modeling temporal dynamics of user preference can give rise to increasing performance for both pairwise learning algorithms, and t-test show that the different is significant (p-value ¡ 0.05). We can also find that **BHS-seg** is more sensitive to temporal dynamics of user preference. All of these imply that it is necessary to model the temporal dynamics of user preferences into the pairwise learning for Web content optimization.

5.2.4. Effects of Position-Sensitive Preferences. We now investigate the effects of applying position-sensitive preferences on pairwise learning methods. As mentioned in Sec-



(a) Trending Now dataset



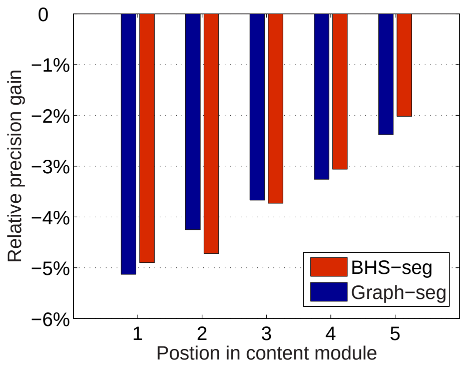
(b) News dataset

Fig. 8. Relative precision gain by pairwise algorithms with modeling temporal dynamics of user preferences over the same algorithms without modeling such temporal dynamics

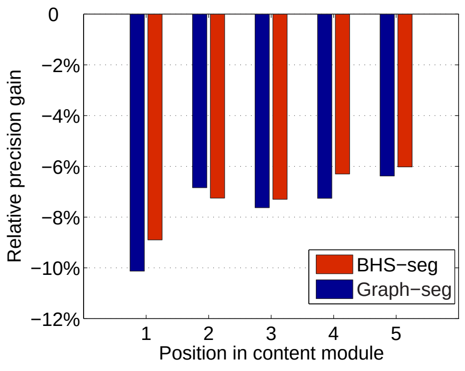
tion 4.1, we employ all preferences without considering the difference in click position in our pairwise learning methods. However, many previous works have studied position-sensitive preference extraction, e.g. '*skip-above*' and '*skip-next*' strategies [Joachims et al. 2005], which are effective in the context of ranking for search. In this experiment, we study if such position-sensitive preferences are effective as well for Web content optimization on Web portals. Specifically, we compare the performance of our proposed pairwise learning algorithms using all preferences with the same algorithms using only preferences based on '*skip-above*' and '*skip-next*' strategies.

Figure 9 illustrates relative precision gain of **Graph-seg** and **BHS-seg** using only preferences based on '*skip-above*' and '*skip-next*' strategies over the same algorithms using all preferences. From the figure, we can find that, using merely preferences generated by '*skip-above*' and '*skip-next*' strategies cannot increase the performance of content optimization but even results in declining recommendation performance. We hypothesize the reasons as: First, unlike a long ranked list of search results, one content module usually occupies a small area of the portal webpage, the content items in which could be all browsed with quick scan; Second, as shown in Figure 1, content modules may have more complex organization for presentation rather than an ordered list; Moreover, using only preferences based on '*skip-above*' and '*skip-next*' could cause insufficiency of learning samples. Therefore, traditional position-sensitive preference extraction in search may not be effective in content optimization on Web portals. Exploring better methods for position-sensitive preference extraction is not the focus of this work, we leave this topic as future work.

In summary, our experimental results have demonstrated that: 1) Pairwise learning methodology can achieve much better performance for online recommendation than pointwise modeling methodology; 2) Pairwise learning methods are more robust to sparse learning samples than pointwise models, which indicates its advantage to be applied for personalized recommendation since there are sparser samples for learning personalized models; 3) Temporal dynamics of user preferences are important factors to improve pairwise recommendation models; 4) Straightforward position-sensitive strategies, popular used in Web search, are not quite effective to select preferences that are more critical to build pairwise recommendation models.



(a) Trending Now dataset



(b) News dataset

Fig. 9. Relative precision gain by pairwise algorithms using only preferences based on '*skip-above*' and '*skip-next*' strategies over the same algorithms using all preferences.

6. DISCUSSIONS

To complement the empirical findings in the previous section, this section aims at providing theoretical insights that further justify the advantages of pairwise approaches over pointwise ones. To make the discussion concrete and free from notation clutter, we take a more careful look at the EMP method with the simplification of $w = 1$; the validity of our conclusions should not be affected when $w < 1$.

1. Robustness to sparse learning samples:

The first issue with pointwise methods like EMP is the need for large amount of training data in order to obtain reliable CTR estimates. To see this, consider an item with CTR p, which is estimated by $\hat{p} = c/n$ in EMP, where c and n are the observed clicks and views of that item in the data. While $\hat{p}$ is unbiased in the sense that $\mathbb{E}[\hat{p}] = p$, standard results from statistics and learning theory (e.g., [Kearns and Vazirani 1994]) imply that the relative error, $|p - \hat{p}|/p$, is on the order of $1/\sqrt{np}$, and the bound is tight up to a constant.

Now suppose we have two items whose CTRs are p_1 and p_2, and the relative difference is $\rho = |p_1 - p_2|/\max\{p_1, p_2\}$. To ensure the EMP estimates $\hat{p}_1$ and $\hat{p}_2$ preserve the same order as p_1 and p_2 (so that the right recommendation can be made), roughly $\rho^{-2}(p_1^{-1} + p_2^{-1})$ samples are needed. In many Internet applications like those studied in the previous section, CTRs are usually quite low, and the relative difference ρ can be small as well. All these factors contribute to the requirement of large data size, and provide insights that predict when EMP is likely to suffer when data is sparse.

In contrast, pairwise methods are able to take advantage of the implicit preference information from clicks, which can be used to refine a recommendation models even for non-clicked items. More formally, the (i, j) entry in the transition matrix built in the PageRank-like algorithm indicates the probability a user prefers item j over i. Under a reasonable assumption that the ratio $P(i, j)/P(j, i)$ equals $x(j)/x(i)$ for some attractiveness scores $x(\cdot)$ that is not necessarily the CTR, we can prove that the computed PageRank score $s(i)$ will be proportional to $x(i)$. Furthermore, how fast the PageRank score $s(i)$ converges to the "correct value", $x(i)/\sum_j x(j)$, depends on the condition number that describes how sensitive s is to errors in the transition matrix [Cho and Meyer 2001]. While it is non-trivial to compare to EMP, preliminary simulation (not reported in the present paper) verifies the greater robustness of pairwise methods. Finally, the ρ parameter in Algorithm 1 may also be used to control the bias/variance trade-off in our graph-based method.

2. Robustness to inconsistency caused by interaction of displayed items:

An even more serious problem is the inconsistency of EMP caused by *interaction* of displayed items. As an example, when a user visits the TrendingNow Module and sees a list of recommended queries, she would compare the candidate queries subconsciously and click on the most interesting one (if she decides to click at all). Clearly, the CTR of an item can depend on what *other* items are displayed in that module; in other words, *there is a competition for user attention among displayed items*.

Suppose we have two items i and j, whose CTR estimates in EMP are robustly obtained from a large amount of data and satisfy $\hat{p}_i > \hat{p}_j$. Can we then infer item i is better than j? The answer is no, unfortunately—it may just be the case that i has been displayed with good items *more often* than j. Thus, a direct comparison between $\hat{p}_i$ and $\hat{p}_j$ in EMP can be misleading, since the critical interaction information is missing. It should be noted that the use of a random learning bucket (Section 3.1) does not eliminate the inconsistency, since an item can be added to or removed from the candidate pool at any moment.

In contrast, pairwise methods do not suffer the same problem. The (i, j) entry in the transition matrix captures the *relative* quality between i and j, which is independent (or at most weakly dependent) of what other items are displayed. Computing the PageRank score from the transition matrix becomes the process of identifying individual items' quality measure from pairwise ranking results, a problem closely related to the NP-hard minimum feedback arc set problem [Alon 2006].

7. CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed a new pairwise learning methodology for Web portal content optimization, where the pairwise preferences are extracted based on users actions on content items. With thorough analysis and discussions, we gained a deeper understanding of the advantages of this new methodology over state-of-the-art methods. To apply this pairwise learning approach to real online recommender system, we introduced two specific pairwise learning algorithms, a straightforward graph based algorithm and a more formalized Bayesian modeling one. Experiments on two large-scale data sets, collected from content modules of a real-world portal website, demonstrated that the pairwise learning methods can achieve significant improvement in terms of a precision metric over the baseline pointwise models. Further analysis illustrated that the new pairwise methods can be more beneficial to personalized recommendation than pointwise model since the pairwise methods are more robust to the problem of sparse learning samples.

In the future, we are interested in exploring more user behavior based features to build the pairwise learning methods. We also aim at studying how to better interpret user actions into preference by considering the position bias and users' browsing habits. Moreover, we plan to explore other common model assumption other than Gaussian for building the Bayesian Hidden Score model. Last but not least, we will explore how to replace the proposed scoring model by a factorization model.

REFERENCES

AGARWAL, D., CHEN, B.-C., AND ELANGO, P. 2009. Spatio-temporal models for estimating click-through rate. In *Proc. of WWW*.

AGARWAL, D., CHEN, B.-C., AND ELANGO, P. 2010. Fast online learning through offline initialization for time-sensitive recommendation. In *Proc. of KDD*.

AGARWAL, D., CHEN, B.-C., ELANGO, P., MOTGI, N., PARK, S.-T., RAMAKRISHNAN, R., ROY, S., AND ZACHARIAH, J. 2008. Online models for content optimization. In *NIPS*.

AGARWAL, D., CHEN, B.-C., ELANGO, P., AND WANG, X. 2012. Personalized click shaping through lagrangian duality for online recommendation. In *Proc. of SIGIR*.

AHN, J., BRUSILOVSKY, P., GRADY, J., HE, D., AND SYN, S. Y. 2007. Open user profiles for adaptive news systems: help or harm? In *Proc. of WWW*.

ALON, N. 2006. Ranking tournaments. *SIAM Journal on Discrete Mathematics 20,* 1, 137–142.

BILLSUS, D. AND PAZZANI, M. 2007. Adaptive news access. In *The Adaptive Web - Methods and Strategies of Web Personalization*.

BRODER, A. 2008. Computational advertising and recommender systems. In *Proc. of RecSys*.

BURGES, C., SHAKED, T., RENSHAW, E., LAZIER, A., DEEDS, M., HAMILTON, N., AND HULLENDER, G. 2005. Learning to rank using gradient descent. In *ICML*.

CANDES, E. AND RECHT, B. 2008. Exact matrix completion via convex optimization. *Found. of Comput. Math. 9*, 717–772.

CHO, G. E. AND MEYER, C. D. 2001. Comparison of perturbation bounds for the stationary distribution of a Markov chain. *Linear Algebra and Its Applications 335,* 1–3, 137–150.

CHU, W., PARK, S. T., BEAUPRE, T., MOTGI, N., AND PHADKE, A. 2009. A case study of behavior-driven conjoint analysis on yahoo! front page today module. In *Proc. of KDD*.

DAS, A., DATAR, M., GARG, A., AND RAJARAM, S. 2007. Google news personalization: scalable online collaborative filtering. In *Proc. of WWW*.

DONG, A., BIAN, J., HE, X., REDDY, S., AND CHANG, Y. 2011. User action interpretation for personalized content optimization in recommender systems. In *Proc. of CIKM*.

FREUND, Y., IYER, R. D., SCHAPIRE, R. E., AND SINGER, Y. 1998. An efficient boosting algorithm for combining preferences. *Proc. of ICML*.

GABRILOVICH, E., DUMAIS, S., AND HORVITZ, E. 2004. Newsjunkie: providing personalized newsfeeds via analysis of information novelty. In *Proc. of WWW*.

HERLOCKER, J. L., KONSTAN, J. A., BORCHERS, A., AND RIEDL, J. 1999. An algorithmic framework for performing collaborative filtering. In *Proc. of SIGIR*.

HOFMANN, T. AND PUZICHA, J. 1999. Latent class models for collaborative filtering. In *Proc. of IJCAI*.

JIN, R., CHAI, J. Y., AND SI, L. 2004. An automatic weighting scheme for collaborative filtering. In *Proc. of SIGIR*.

JOACHIMS, T. 2002. Optimizing search engines using clickthrough data. In *Proc. of KDD*.

JOACHIMS, T., GRANKA, L., PAN, B., HEMBROOKE, H., AND GAY, G. 2005. Accurately interpreting clickthrough data as implicit feedback. In *Proc. of SIGIR*.

KANAGAL, B., AHMED, A., PANDEY, S., JOSIFOVSKI, V., YUAN, J., AND GARCIA-PUEYO, L. 2012. Supercharging recommender systems using taxonomies for learning user purchase behavior. In *VLDB*.

KEARNS, M. J. AND VAZIRANI, U. V. 1994. *An Introduction to Computational Learning Theory*. MIT Press.

KONSTAN, J. A., MILLER, B. N., MALTZ, D., HERLOCKER, J. L., GORDON, L. R., AND RIEDL, J. 1997. Grouplens: applying collaborative filtering to usenet news. In *Communications of the ACM*.

KOREN, Y. 2008. Factorization meets the neighborhood: A multifaceted collaborative filtering model. In *KDD*.

KOREN, Y. 2009. Collaborative filtering with temporal dynamics. In *KDD*.

LI, L., CHU, W., LANGFORD, J., AND SCHAPIRE, R. E. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proc. of WWW*.

LIU, T.-Y. 2009. *Learning to Rank for Information Retrieval*. Foundations and Trends in Information Retrieval.

MELVILLE, P., MOONEY, R. J., AND NAGARAJAN, R. 2002. Content-boosted collaborative filtering for improved recommendations. In *Proc. of AAAI*.

RENDLE, S., FREUDENTHALER, C., GANTNER, Z., AND SCHMIDT-THIEME, L. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *UAI*.

RENDLE, S., FREUDENTHALER, C., AND SCHMIDT-THIEME, L. 2010. Factorizing personalized markov chains for next-basket recommendation. In *Proc. of WWW*.

RICHARDSON, M., DOMINOWSKA, E., AND RAGNO, R. 2007. Predicting clicks: estimating the click-through rate for new ads. In *Proc. of WWW*.

SREBRO, N., RENNIE, J., AND JAAKKOLA, T. 2005. Maximum-margin matrix factorization. In *NIPS*.

STERN, D. H., HERBRICH, R., AND GRAEPEL, T. 09. Matchbox: Large-scale online bayesian recommendation. In *WWW*.

TAKÁCS, G. AND TIKK, D. 2012. Alternating least squares for personalized ranking. In *RecSys*.

WANG, J., DE VRIES, A. P., AND REINDERS, M. J. T. 2006. Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In *Proc. of SIGIR*.

WIND, Y. 1978. Issue and advances in segmentation research. In *Journal of Marketing Research*.

XIONG, L., CHEN, X., HUANG, T.-K., SCHNEIDER, J., AND CARBONELL, J. G. 2010. Temoral collaborative filtering with Bayesian probabilistic tensor factorization. In *SDM*.

YANG, S. H., LONG, B., SMOLA, A., ZHA, H., AND ZHENG, Z. 2011. Collaborative competitive filtering: Learning recommender using context of user choice. In *SIGIR*.

YANG, Y., YOO, S., ZHANG, J., AND KISIEL, B. 2005. Robustness of adaptive filtering methods in a cross-benchmark evaluation. In *Proc. of SIGIR*.

YU, K., TRESP, V., AND YU, S. 2004. A nonparametric hierarchical bayesian framework for information filter. In *Proc. of SIGIR*.

ZHANG, Y. AND KOREN, J. 2007. Efficient bayesian and hierarchical user modeling for recommendation systems. In *Proc. of SIGIR*.

ZHENG, Z., CHEN, K., SUN, G., AND ZHA, H. 2007. A regression framework for learning ranking functions using relative relevance judgments. In *Proc. of SIGIR*.

ZIGORIS, P. AND ZHANG, Y. 2006. Bayesian adaptive user profiling with explicit and implicit feedback. In *Proc. of CIKM*.

Received Jun 2012; revised Month 20–; accepted Month 20–