Learning with Sparse and Biased Feedback for Personal Search

Michael Bendersky, Xuanhui Wang, Marc Najork and Donald Metzler
Google, Inc.
{bemike, xuanhui, najork, mezler}@google.com

Abstract

Personal search, including email, on-device, and personal media search, has recently attracted a considerable attention from the information retrieval community. In this paper, we provide an overview of challenges and opportunities of learning with implicit user feedback (e.g., click data) in personal search. Implicit user feedback provides a convenient source of supervision for ranking models in personal search. This feedback, however, has two major drawbacks: it is highly sparse and biased due to the personal nature of queries and documents. We demonstrate how these drawbacks can be overcome, and empirically demonstrate the benefits of learning with implicit feedback in the context of a large-scale email search engine.

1 Introduction

Researchers have been exploring how to successfully leverage user feedback to improve search quality for over a decade [Joachims, 2002; Joachims et al., 2005]. User feedback most often comes in the form of clicks on links to search results, but may be derived from other sources, including page visits [Richardson et al., 2006], cursor tracking [Guo and Agichtein, 2012], or touch gestures [Guo et al., 2013]. Such user interaction data has been shown to be particularly useful for training learning to rank models [Agichtein et al., 2006; Richardson et al., 2006] and click-through rate prediction [Richardson et al., 2007].

However, even though the use of interactions for improving search over public search corpora (e.g., web search) is commonplace, there is little to no research regarding its use for search over personal corpora, a.k.a. personal search. Personal search has many real-life applications including (but not limited to) email search [Carmel et al., 2015; Wang et al., 2016], desktop search [Dumais et al., 2003],

Figure 1: Illustrative example of email search results for query [book order number]. The first two results are skipped, and the last one is clicked. and, most recently, on-device search [Kamvar et al., 2009] and personal media search [Anguera et al., 2008; Guy et al., 2018].

In all of these personal search applications, leveraging user feedback for improving search quality has been limited by its sparseness. This sparseness arises from the fact that in the personal search scenario each user has access only to their own private corpus (e.g., emails, documents or multimedia files). This means that cross-user interactions with the same item, which are common in web search (i.e., millions of users visiting the same web page) are non-existent in personal search.

Second, user queries in personal search may not generalize as well as in web search due to the private nature of the underlying corpora. For instance, one common use case in email search is retrieving some personal information of a correspondent, e.g. [marta schedule], or [from:john highest-priority] [Carmel et al., 2015]. This is very different from web search, where the most common queries are issued by multiple users with the same underlying target page in mind.

For instance, consider the email search example in Figure 1. In this case the user skipped the first two results (even though they might have more terms in common with the query book order number) and clicked on the last result. It would be impossible to directly leverage this specific interaction to learn a model for other users given the private nature of the interaction (since no other user received an email with the same exact order number).
However, by aggregating non-private query and document attributes (i.e., those that exclude any personal information such as order number) across a large number of user interactions, it is possible to identify privacy-preserving query-document associations that can be leveraged to improve search quality across all users. For instance, by using term associations, we can learn that emails with the frequent term receipt in the subject are likely to be relevant to queries containing the frequent n-gram order number. As another example, using structural associations [Ailon et al., 2013], we can learn that emails from an online bookstore AliceBookseller.com that correspond to a subject template Your order receipt * are more likely to be relevant to queries containing the frequent n-gram book order.

In addition to the sparsity problem, implicit user feedback such as click data is biased [Joachims et al., 2005; Wang et al., 2016]. For instance, consider Figure 2, which shows two queries Q1 and Q2. The relevant document for Q1 is at position 1 and is clicked every time the query is issued. On the contrary, the relevant document for Q2 is at position 2 and is clicked only half of the time, due to the user propensity to pay less attention to the lower rank results.

The problem illustrated in this example is confirmed by eye tracking studies as well, which found that the users are less likely to see, and hence click on, lower-ranked documents [Joachims et al., 2005; Richardson et al., 2007]. This click position bias leads, in turn, to selection bias – queries with clicks on lower rank positions tend to be under-represented in training data for learning to rank models, as shown by [Wang et al., 2016].

While in a public search setting such as web search selection bias could be corrected by collecting explicit human ratings, in the personal search setting collecting such ratings is much harder since raters can only label their own queries and the corresponding results. In addition, such ratings will be heavily dependent on the selected raters, creating yet another source of bias. Finally, such ratings are costly to maintain due to the dynamic nature of personal search collections. Therefore, in personal search, researchers and practitioners [Tromba et al., 2017] often have to rely on click data to optimize and evaluate ranking models.

In the next section, we provide an exposition of our recent work on dealing with the challenges of sparse and biased user feedback in the personal search setting. Then, in Section 3, we provide a brief overview of empirical evaluation of our methods in the context of a large-scale email search engine. We conclude the paper in Section 4.

2 Model Overview

In this section, we overview several models that deal with the sparsity and bias that are inherent to learning with click data in personal search, and in particular in an email search setting. We start with a brief overview of the learning to rank [Liu, 2009] methodology for email search in Section 2.1. In Section 2.2, we propose a novel way to reduce sparsity in user feedback via cross-user aggregation [Bendersky et al., 2017]. In Section 2.3 we describe how we deal with position bias in user feedback [Wang et al., 2016; Wang et al., 2018].

2.1 Learning to Rank in Email Search

In the most general setting, a training data for learning the optimal search ranking (a.k.a. learning to rank [Liu, 2009]) consists of a query set Q, where for each query q, we are given a list of corresponding documents D_q. Each query-document pair (q ∈ Q, d ∈ D_q), is associated with a feature vector x_{q,d} and a corresponding relevance label l_{q,d}. While in scenarios like web search, relevance labels l_{q,d} are often obtained using explicit relevance judgments, in the personal search setting such as email search labels are usually derived from click data, and thus are sparse and biased.

The goal of learning to rank algorithms is to produce an optimal ranking function sc(x_{q,d}). There are many approaches to this problem, roughly categorized as pointwise, pairwise and listwise [Liu, 2009]. Their overview is out of the scope of this paper, however it is important to note that the techniques discussed in the next sections are agnostic to the choice of any particular approach.

Feature vector x_{q,d} may contain any signals derived from query q, document d or both. In particular, for email search, the features may be derived from the message metadata, sender, recipient and textual similarity between the message and the query (see [Carmel et al., 2015] for an overview).

In the next section, we demonstrate how click-based features can also be incorporated in the feature vector.

2.2 Learning with Sparse Feedback

Historical user click data, as observed in the search logs, can provide a powerful signal for click-through rate prediction and learning to rank models, since it directly reflects user behavior. For instance, if we observe previous interactions for a given query-document pair (q, d), we may use it as a query-dependent matching feature in a feature vector x_{q,d} in a learning to rank model (e.g., aggregate number of
Frequent machine-generated subject templates, e.g., Your package number 123 → Your package number * (see, e.g., Ailon et al. [Ailon et al., 2013] for more details on subject template generation).

Table 1: Summary of the query and document attribute types. Only attribute values that appear across more than \( n \) users in our dataset are considered to be frequent. The infrequent attribute values are discarded.

<table>
<thead>
<tr>
<th>Categories</th>
<th>Document</th>
<th>Query</th>
</tr>
</thead>
<tbody>
<tr>
<td>Small set of commonly used email labels, e.g., Purchases, Promos, Forums, etc. (see, e.g., [Agarwal, 2014] for label examples).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Frequent machine-generated subject templates, e.g., Your package number 123 → Your package number * (see, e.g., Ailon et al. [Ailon et al., 2013] for more details on subject template generation).</td>
<td>Longest frequent n-gram appearing in the query, e.g., bob weekly schedule → ['weekly schedule']</td>
<td></td>
</tr>
</tbody>
</table>

The case in personal search is different. Users usually do not share documents (e.g., emails or personal files), and therefore directly aggregating click history across users becomes infeasible. To address this problem, instead of directly learning from user behavior for a given \((q, d)\) pair, we instead choose to represent documents and queries using semantically coherent attributes that are in some way indicative of their content.

This approach is schematically described in Figure 3. Both documents and queries are projected into an aggregated attribute space, and the matching is done through that intermediate representation, rather than directly. Since we assume that the attributes are semantically meaningful, we expect that similar personal documents and queries will share many of the same aggregate attributes, making the attribute level matches a useful feature in a learning to rank model.

Some examples of privacy-preserving query-document associations that could potentially be learned by aggregating across a large number of private user interactions are presented in Table 1.

In [Bendersky et al., 2017] we demonstrate that even very highly-dimensional attributes like n-grams can be efficiently incorporated into the learning to rank paradigm described in Section 2.1 without dramatically increasing the size of the feature vector \( x_{q,d} \). This is achieved via an attribute parameterization technique, in which sparse attributes are parameterized using their respective clickthrough rates. [Bendersky et al., 2017] show that for \( m \) document attribute types and \( n \) query attribute types, attribute parameterization will generate an \( m \)-dimensional feature vector \( P_d \) of query independent features and an \( mn \)-dimensional feature vector \( P_{q,d} \) of query-dependent features.

In general, we will assume that there exists a base score \( sc_b(x_{q,d}) \) for every query-document pair. It can be based on keyword matching or some other ranking features used in private corpora (see, e.g., [Carmel et al., 2015] for an overview). In [Bendersky et al., 2017], we use an adaptive approach, and train the adjustment \( \delta(P_d, P_{q,d}) \) over the base score \( sc_b(x_{q,d}) \). The scoring function thus becomes \( sc_b(x_{q,d}) + \delta(P_d, P_{q,d}) \), which is convenient for our production-environment system, where the base score is already highly optimized, and is disjoint with the newly introduced attribute parameterization features.

The additive nature in this adaptive formulation naturally fits the Multiple Additive Regression Trees (MART) learning algorithm [Hastie et al., 2001]. In every iteration, MART trains a new tree to be added to the existing list of trees. In our setting, we start with the base score \( sc_b(x_{q,d}) \) and then train additive trees over this score.

### 2.3 Learning with Biased Feedback

The position bias model assumes that the observed click – modeled by Bernoulli variable \( C \) – depends on two factors: (a) whether a user examines a document at position \( k \), and (b) whether document \( d \) is relevant to query \( q \). We can then model the probability of a click as

\[
P(C = 1 | q, d, k) = \theta_k \gamma_{q,d}.
\]

where \( \theta_k \) is the probability that position \( k \) is examined, and \( \gamma_{q,d} \) is the probability that document \( d \) is relevant to query \( q \). Note that the model assumes that the examination only depends on the position and the relevance only depends on the query and document, a common assumption in click models [Chuklin et al., 2015].

Both \( \theta_k \) and \( \gamma_{q,d} \) are hidden, and there are several ways to estimate these parameters. It is easy to show that by randomizing the results shown to the user, the expected relevance at each position is constant, and \( \theta_k \) will be proportional to the number of clicks at position \( k \) in the randomized data. However, this can hurt the performance by up to 30% in live traffic systems [Wang et al., 2018]. To avoid this quality degradation, we can instead resort to random pair inversion, which can significantly reduce the quality decrease. Can we do even better and estimate the position bias directly from click data?

To this end, we propose a novel regression-based EM algorithm [Wang et al., 2018]. Note that in a standard EM algorithm, we would require multiple observations from each query-document pair \((q, d)\) to reliably estimate the relevance \( \gamma_{q,d} \). This is not feasible in a personal search scenario where click data is highly sparse. Therefore, in the regression-based EM algorithm we use the feature vector \( x_{q,d} \), and use a function \( f \) to compute the relevance: \( \gamma_{q,d} = f(x_{q,d}) \). The Maximization step attempts to find a regression function \( f(x_{q,d}) \) to maximize the likelihood given the estimation from the Ex-
Figure 4: Position bias estimated by several methods and normalized by the top position.

<table>
<thead>
<tr>
<th>%MRR</th>
<th>RandPair Correction</th>
<th>EM Correction</th>
</tr>
</thead>
<tbody>
<tr>
<td>+2.14</td>
<td>+1.94</td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Effects of bias correction on ranking performance. All the differences are reported compared to the unweighted baseline, and are statistically significant.

3 Experimental Results

3.1 Position Bias Estimation

To evaluate the EM algorithm described in Section 2.3, we examine how well it approximates the empirical position bias that can be obtained using full result randomization. The results are presented in Figure 4, where we compare four alternatives (a) Full result randomization (Empirical), (b) random pair inversion (RandPair), (c) regression-based EM algorithm described in Section 2.3 (EM), and (d) an embedded approach, where position is directly embedded into the function $g(x_{i,d}, k)$ as a feature to approximate bias (Embedded).

As we can see from Figure 4, all the techniques underestimate the Empirical position bias. EM clearly outperforms the Embedded approach, especially at the lower ranks. It achieves an estimation comparable to RandPair, without incurring any loss in quality (as RandPair requires result randomization on live search traffic).

3.2 Ranking with Biased Feedback

As shown in Figure 4, clicks are biased. Therefore, when evaluating quality changes using click data in lieu of explicit human ratings (as we do in this paper due to the personal nature of the data), this bias needs to be corrected by incorporating it into the evaluation metric. To this end, [Wang et al., 2016] propose a weighted variant of a standard MRR (mean reciprocal rank) metric. In this variant, query $i$ is weighted by $w_i = \frac{1}{k(i)}$, where $k(i)$ is the click position for query $i$, and $b_k(i)$ is the empirical bias at $k(i)$-th position, as shown in Figure 4. Then, the MRR for a set of queries is defined as $MRR = \frac{1}{P} \sum_i \frac{1}{w_i} \frac{b_k(i)}{k(i)}$.

Table 2 shows the effects of correcting the bias when ranking with biased feedback. In both cases, the original training data is reweighted to correct the bias as estimated by either RandPair or EM techniques. As we can see, although the EM algorithm does not require any prior randomization it achieves ranking performance that is roughly 2% better than the unweighted variant, and statistically indistinguishable from the RandPair algorithm (which requires randomization).

3.3 Attribute Parameterization Evaluation

Using the weighted MRR metric presented in the previous section, in Table 3 we evaluate the variants of the attribute parameterization approach described in Section 2.2. In this table, we compare both query-dependent and query-independent features. For each of them, we train our ranking function by adding each attribute type individually as a feature (the first 3 rows in the table). We then combine all the query-dependent and query-independent parameterized attribute types respectively to form the “all” in the two columns of the table. The “Full Model” uses both the query-dependent and query-independent parameterized attribute types as features in a single ranking function.

From Table 3 we can observe that a combination of all the attribute types outperforms each individual attribute type, resulting in overall improvements of +2.10% for query-independent and +2.60% for query-dependent features. This highlights the fact that the selected attribute types are indeed complimentary to each other, and can provide incremental improvements. Further combining all the features and attribute types in the full model results in the best performance, and outperforms the baseline by +3.24%. These improvements unequivocally demonstrate the importance of cross-user feedback aggregation for personal search quality.

4 Conclusions

In this paper we have discussed two novel approaches of dealing with sparsity and bias of user feedback in the personal search setting: query and document attribute parameterization and a regression-based EM algorithm to learn click bias. The proposed approaches are both motivated theoretically as well as demonstrate significant quality improvements in a setting of a large-scale email search engine. They also open up several interesting possibilities for future exploration of other types of bias (e.g., presentation bias [Yue et al., 2010]) and other attribute types.
References


