# Learning from User Interactions in Personal Search via Attribute Parameterization

Michael Bendersky, Xuanhui Wang, Donald Metzler, Marc Najork
Google Inc., Mountain View, CA
{bemike, xuanhui, metzler, najork}@google.com

## ABSTRACT

User interaction data (e.g., click data) has proven to be a powerful signal for learning-to-rank models in web search. However, such models require observing multiple interactions across many users for the same query-document pair to achieve statistically meaningful gains. Therefore, utilizing user interaction data for improving search over personal, rather than public, content is a challenging problem. First, the documents (e.g., emails or private files) are not shared across users. Second, user search queries are of personal nature (e.g., "alice's address") and may not generalize well across users. In this paper, we propose a solution to these challenges, by projecting user queries and documents into a multi-dimensional space of fine-grained and semantically coherent attributes. We then introduce a novel parameterization technique to overcome sparsity in the multi-dimensional attribute space. Attribute parameterization enables effective usage of cross-user interactions for improving personal search quality – which is a first such published result, to the best of our knowledge. Experiments with a dataset derived from interactions of users of one of the world's largest personal search engines demonstrate the effectiveness of the proposed attribute parameterization technique.

## Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

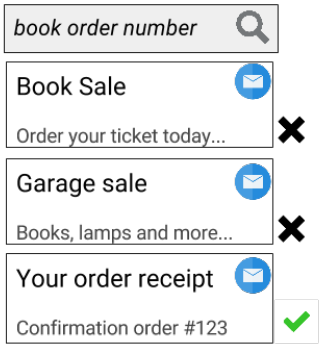
## Keywords

User interactions, personal search, attribute parameterization

## 1. INTRODUCTION

Researchers have been exploring how to successfully leverage user interaction data to improve search quality for over a decade [22, 23]. User interactions most often come in the form of clicks on links to search results, but may be derived from other sources, including page visits [27], cursor tracking [18], or touch gestures [19]. Such user interaction data has been shown to be particularly useful for training learning-to-rank models [2, 6, 27] and click-through rate prediction [26].

*WSDM 2017 February 06-10, 2017, Cambridge, United Kingdom*
© 2017 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-4675-7/17/02.
DOI: http://dx.doi.org/10.1145/3018661.3018712



book order number

Book Sale
Order your ticket today...
✗

Garage sale
Books, lamps and more...
✗

Your order receipt
Confirmation order #123
✓

**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.**

However, even though the use of interactions for improving search over public search corpora (e.g., the web) 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 [11, 28], desktop search [15], and, most recently, on-device search [24] and personal media search [4].

In all of these personal search applications, the use of user interaction data for improving search quality has been limited by several factors. First, 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]* [11]. This is very different from web search, where the most common queries are navigational [10] and are issued by multiple users with the same underlying target page in mind (e.g., *[facebook]* → facebook.com, *[youtube]* → youtube.com).

To address these factors, we propose a simple yet effective technique that projects both documents and queries into an *attribute space.* Our general approach is independent of the particular attribute choices, but in general these attributes have to be semantically coherent and fine-grained enough to aggregate user interactions with semantically closely related items.

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 exact same order number). In addition, doing so could leak potentially sensitive information. To preserve privacy, terms, n-grams and queries should be *frequent*, that is, they should be contained in the private corpora or query logs of sufficiently many users.

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.

Some examples of privacy-preserving query-document associations that could potentially be learned by aggregating across a large number of private user interactions include:

- *Term-to-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.*
- *Category-to-term associations.* We can learn that emails that are *purchase-related* are likely to be relevant to queries containing the frequent n-gram *order number.*
- *Category-to-category associations.* We can learn that *finance-related* emails are likely to be relevant to queries classified as *investment-related.*
- *Structure-to-term associations.* Using structural templatization approaches recently proposed by Ailon et al. [3], we can learn that emails from sender *usps.com* that correspond to a subject template *Your package number ** are more likely to be relevant to a frequent query *[usps]* than a subject template *USPS package discount for *!* from the same sender.

Hence, instead of looking at specific and potentially private user interactions, we instead propose aggregating non-private query and document attributes across a large number of user interactions. To the best of our knowledge, this is a first published work on leveraging user interaction data in a privacy-preserving manner for personal search.

To summarize, our contributions in this paper are as follows. First, we introduce a general formal model of user interactions in personal search, which utilizes a novel *attribute parameterization* technique. Then, we provide a concrete instantiation of this technique that focuses on email search, a personal search scenario that received some attention from the research community in the recent years [11, 28]. Finally, using a series of experiments conducted on a dataset derived from search interactions of users of one of the world's largest email services, we demonstrate that the attribute parameterization technique can lead to substantial search quality improvements.

## 2. RELATED WORK

Query-independent static document quality priors have a long history in web search, including link-based priors [9], content-based priors [7] and user interaction-based priors [13, 27]. Interaction priors (e.g., search result clicks or page visits) are especially relevant to this work, since they allow to directly incorporate user behavior into the ranking model. In this work, we extend the notion of *document* interactions to *attribute* interactions, which makes interaction-based priors feasible in the context of personal search, where documents are not shared across users.

In this work, we also propose the use of query-dependent interactions for personal search. Prior work explored the use of such interactions as features [2] or noisy labels [23] in learning-to-rank models for web search or as a source for recommendations of authoritative pages on the query topic [30]. We expand this prior work by learning attribute-level interactions, which enables their use in situations where click data for each $\langle query,\ doc \rangle$ pair is inherently sparse.

Most recently, researchers have started using interactions such as clicks for deep architectures for learning semantic matches between queries and documents in web search (see, e.g., [16, 21]). While some of these deep learning models can be in principle adapted to personal search, they have several shortcomings compared to the approach proposed in this work.

First, these models generally take a "bag-of-words" assumption and are not able to leverage semantic information about the document like its category, structure or type. This is detrimental in the personal search settings where the retrieved document types may be heterogeneous (e.g., documents, slides, spreadsheets or photos in the context of cloud storage search). Second, due to their complexity, the amount of data used to train these models is inherently limited, while the proposed attribute parameterization model is compact, and can handle a virtually unlimited amount of data through aggregation. Finally, as some recent work suggests, attribute parameterization can serve as a potentially useful feature for the *wide and deep* architectures that were shown to be useful in large scale retrieval and recommendation systems [12].

While the findings of this paper are applicable to any personal search scenario, our experimentation is focused on a large email corpus. Prior work in email search [11, 15, 28] recognized the importance of relevance ranking and applying learning-to-rank for this task. However, to the best of our knowledge, this work is the first attempt to adapt aggregate user interactions for improving email search quality.

## 3. MODEL

Historical user interaction 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. Most often, user interaction data is used in search in its most direct form. For instance, if we observe previous interactions for a given $\langle query,\ doc \rangle$ pair, we may use it as a query-dependent matching feature in a learning-to-rank model (e.g., aggregate number of clicks [2]).

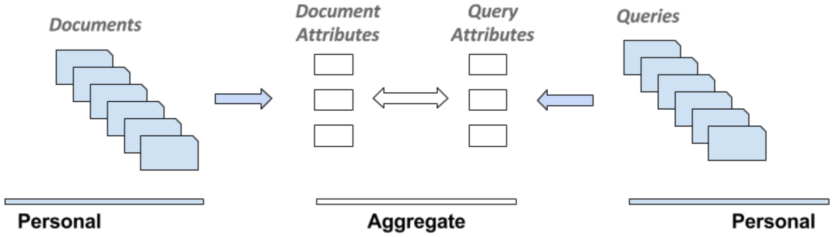


**Figure 2: System diagram of attribute aggregation and matching process.**

| Entity notation | Entity Set Notation | Description |
|---|---|---|
| $d$ | $\mathcal{D}$ | Private documents (e.g., emails) in the indexed corpus. |
| $q$ | $\mathcal{Q}$ | User queries observed in a search log (e.g., *[bob's address]*). |
| $a^d_{ij}$ | $\mathcal{A}^{\mathcal{D}}$ | Document attributes (e.g., document category). |
| $a^q_{kl}$ | $\mathcal{A}^{\mathcal{Q}}$ | Query attributes (e.g., query terms). |

**Table 1: Summary of the entity definitions.**

Similarly, if we observe that some *doc* is often clicked across searches, we may use it as a static a-priori feature of the document's overall quality [6, 27].

The case in private search is different. Users usually do not share documents (e.g., emails or personal files), and therefore directly aggregating interaction history across users becomes infeasible. To address this problem, instead of directly learning from user behavior for a given ⟨*query, doc*⟩ pair like in web search, 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 2. 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.

In the remainder of this section, we provide a more formal model of this intuitive relationship between private documents, queries and aggregates. First, in Section 3.1, we formally define a quadrilateral structure of bipartite graphs which is used for feature derivation. Then, we explain how both query-independent (Section 3.2) and query-dependent (Section 3.2) features can be derived from this structure, and used in a learning-to-rank model (Section 3.4). Finally, we conclude in Section 3.5 by providing some practical attribute instantiations for an email corpus.

## 3.1 Attribute graphs

We start the discussion by formally defining in Table 1 four entities that will be used throughout the discussion that follows: documents, queries and their corresponding attributes. Most generally, we assume that each document can be represented using a (potentially sparse) *attribute matrix* $A^d$, where $i$-th row corresponds to an *attribute type* and $j$-th column corresponds to an *attribute value*. Similarly, each query can be represented by an attribute matrix $A^q$, indexed by $(k, l)$.

In the remainder of this section, we shall assume that the number of attribute types (matrix rows) is $m$ and $n$, for document and query attributes, respectively. The attribute types may be either dense (e.g., distribution over tens of topical categories) or very sparse (e.g., millions of unique terms or bigrams). We defer the exact instantiation of these types to Section 3.5, in order to keep the discussion in this section as general as possible. We will denote $v$ the maximum number of allowed attribute values per type (number of columns in the attribute matrix).

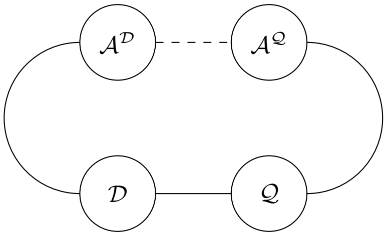


**Figure 3: Quadrilateral structure modeling cross-entity associations discussed in Section 3.1. Observed or pre-constructed associations are marked by solid lines, inferred associations are marked by dashed lines.**

With these definitions in mind, we can define a quadrilateral structure that models the association between the different entities via four bipartite graphs (see Figure 3), which we formally define next.

### 3.1.1 Query-document graph

The *query-document* graph models the interactions between queries and documents, as observed in some historical search log. The bipartite graph is represented by a triple $\mathcal{G}^{\mathcal{QD}} = (\mathcal{Q}, \mathcal{D}, \mathcal{E}^{\mathcal{QD}})$, such that the nodes are queries and documents and the edges in the edge set $\mathcal{E}^{\mathcal{QD}}$ are parameterized by tuples of the form

$$e(q, d) = \langle \gamma_o(q, d), \gamma_c(q, d) \rangle. \tag{1}$$

Parameterization functions $\gamma_o(a, b)$ and $\gamma_c(a, b)$ indicate that entities $a$ and $b$ were observed or clicked in the same search session, respectively. As we show next, these functions can be expanded beyond queries and documents.

### 3.1.2 Document-attribute graph

The *document-attribute* graph models the relationship between documents and some attribute set (e.g., categories or terms). In this paper, we assume that this relationship is pre-constructed and fixed, as described in more detail in Section 3.5. Formally, the bipartite graph is represented by a triple $\mathcal{G}^{\mathcal{D}} = (\mathcal{D}, \mathcal{A}^{\mathcal{D}}, \mathcal{E}^{\mathcal{D}})$, and the edges are indicator functions of the form[1]

$$e(d, a^d_{ij}) = \begin{cases} 1 & d \text{ is associated with attribute } a^d_{ij} \\ 0 & \text{else.} \end{cases} \tag{2}$$

### 3.1.3 Query-attribute graph

Similarly to the document-attribute graph presented above, the *query-attribute* graph models a fixed relationship between a query and some attribute set (e.g., query terms or annotations). Formally, the bipartite graph is represented by a triple $\mathcal{G}^{\mathcal{Q}} = (\mathcal{Q}, \mathcal{A}^{\mathcal{Q}}, \mathcal{E}^{\mathcal{Q}})$, and the edges are indicator functions of the form

$$e(q, a^q_{kl}) = \begin{cases} 1 & q \text{ is associated with attribute } a^q_{kl} \\ 0 & \text{else.} \end{cases} \tag{3}$$

### 3.1.4 Attribute-attribute graph

This graph models *cross-attribute* interactions between the document and the query attributes. While these interactions are never directly observed in the historical search logs, they can be *inferred* using a combination of the three bipartite graphs discussed above. Formally, the attribute graph is represented by a triple $\mathcal{G}^{\mathcal{A}} = (\mathcal{A}^{\mathcal{Q}}, \mathcal{A}^{\mathcal{D}}, \mathcal{E}^{\mathcal{A}})$, and edges are parameterized as

$$\begin{aligned} e(a^q_{kl}, a^d_{ij}) &= \sum_{q: e(q, a^q_{kl}) = 1} \; \sum_{d: e(d, a^d_{ij}) = 1} e(q, d) \\ &= \langle \gamma_o(a^q_{kl}, a^d_{ij}), \gamma_c(a^q_{kl}, a^d_{ij}) \rangle, \end{aligned} \tag{4}$$

where edge functions $e(\cdot)$ are as defined in Equations 1, 2 and 3, respectively. Intuitively, Equation 4 models query-document attribute observed and co-click associations via summation over all the queries and documents that can be associated with their respective attributes.

## 3.2 Query-Independent Attribute Parameterization

Given the bipartite association graphs discussed in the previous section, in this section we attempt to derive a document feature representation based on these associations. Using the edge set of the document-attribute graph $\mathcal{G}^{\mathcal{D}}$ (see Equation 2), it is easy to see that document $d$ can be represented using a binary feature vector

$$\mathcal{I}_d = [e(*, d) \in \mathcal{E}^{\mathcal{D}}]. \tag{5}$$

In other words, we can use a set of edges in the document-attribute bipartite graph, where document $d$ is a target, to derive a set of binary query-independent (a.k.a. static) features.

Assuming $m$ document attribute types, and maximum attribute dimensionality $v$, this will create a document feature representation with $O(mv)$ dimensions. If attribute matrix $A^d$ is sparse, such representation is limiting for many machine learning tasks, both computationally and in terms of model effectiveness.

Instead, we can use historical user interactions to *parameterize* this feature representation. In this work we use a simple parameterization function based on the attribute click-through ratio (CTR), which is defined as a *clicked* to *observed* fraction. CTR is a well-understood user interaction found to be useful in search [2] and sponsored search [26] settings. However, it is possible to extend our parameterization approach to other user interactions like document access counts [27], cursor tracking [18], or touch gestures [19].

Parameterized document feature representation can be derived from the attribute graph by marginalizing out the query attributes from the edges in Equation 4. Formally,

$$\mathcal{P}_d = [\sum_j \frac{\sum_{a^q} \gamma_c(a^q, a^d_{1j})}{\sum_{a^q} \gamma_o(a^q, a^d_{1j})}, \ldots, \sum_j \frac{\sum_{a^q} \gamma_c(a^q, a^d_{mj})}{\sum_{a^q} \gamma_o(a^q, a^d_{mj})}]. \tag{6}$$

It is easy to see that the size of the $\mathcal{P}_d$ vector does not depend on the dimensionality of the individual attribute types. Therefore, Equation 6 always generates an $m$-dimensional document feature representation – one feature per attribute type – regardless of the attribute sparsity.

## 3.3 Query-Dependent Attribute Parameterization

Thus far we have only discussed query-independent document feature representations, not related to a particular search query. In this section, we shall show how this discussion can be further extended to query-dependent feature representations.

Similarly to a query-independent document representation, we can define query-document feature representation using the intersection among the edge sets of the document-attribute and query attribute graphs as

$$\begin{aligned} \mathcal{I}_{q,d} = [e(q, *) \cdot e(*, d) \colon & e(q, *) \in \mathcal{E}^{\mathcal{Q}}, \\ & e(*, d) \in \mathcal{E}^{\mathcal{D}}]. \end{aligned} \tag{7}$$

This is an even sparser representation than the one in Equation 5, since involves a cross-product between the edges of both the query-attribute and document-attribute bipartite graphs. Assuming $m$ document attribute types, $n$ query attribute types, and maximum attribute dimensionality of $v$, this will create an $O(mnv^2)$-dimensional feature representation. For large values of $v$, such representation is clearly too expensive, or even infeasible.

As in the query-independent case, this feature vector dimensionality can be reduced using attribute parameterization

$$\mathcal{P}_{q,d} = [\sum_j \sum_l \frac{\gamma_c(a^q_{1l}, a^d_{1j})}{\gamma_o(a^q_{1l}, a^d_{1j})}, \ldots, \sum_j \sum_l \frac{\gamma_c(a^q_{nl}, a^d_{mj})}{\gamma_o(a^q_{nl}, a^d_{mj})}]. \tag{8}$$

Note that, as in the query-independent case, the number of terms in vector $\mathcal{P}_{q,d}$ does not depend on the dimensionality of the attribute types, since they are marginalized out in the sums in Equation 8. Therefore, the vector $\mathcal{P}_{q,d}$ generates an $mn$-dimensional query-dependent document feature representation – one feature per query-document attribute type pair – regardless of the attribute sparsity.

[1] We use binary associations as edges for the purposes of presentation clarity, and since they empirically proved to be effective. However, the presented approach is applicable to the case of weighted edges without loss of generality.

| | **Document** | **Query** |
|---|---|---|
| *Categories* | Small set of commonly used email labels, e.g., `Purchases, Promos, Forums`, etc. (see, e.g., [1] for label examples). | |
| *Structure* | Frequent machine-generated subject templates, e.g., `Your package number 123` → `Your package number *` (see, e.g., Ailon et al. [3] for more details on subject template generation). | |
| *Content* | Set of frequent n-grams appearing in the email subject, e.g., `Friday lunch invitation for Alice` → `[‘friday lunch’, ‘lunch invitation’]` | Longest frequent n-gram appearing in the query, e.g., `bob weekly schedule` → `[‘weekly schedule’]` |

**Table 2: Summary of the query and document attribute types. Only attribute values that appear across more than $u$ users in our dataset are considered to be *frequent*. The infrequent attribute values are discarded.**

## 3.4 Ranking with Attribute Parameterization

In general, we are going to assume that we have a *base score* $sc_b(q, d)$ for every query-document pair. This score can be based on keyword matching or some other ranking features used in private corpora (see, e.g., Carmel et al. [11] for an overview). In addition, if our goal is to incorporate attribute parameterization in ranking, the final score in its most generic form can be expressed as

$$sc(Q, D) = f(sc_b(q, d), \mathcal{P}_d, \mathcal{P}_{q,d}). \qquad (9)$$

$f(\cdot)$ can be a hand-tuned score combination or an arbitrary machine learned ranking function [25]. In this paper we use an adaptive approach, which keeps the base score fixed, and incrementally adds the attribute parameterization features to the model. This is different from the standard approach, which learns a scoring function using the entire set of features (including base features) simultaneously.

Instead, in the adaptive approach, we aim to train the adjustment $\delta(\mathcal{P}_d, \mathcal{P}_{q,d})$ over the base score $sc_b(q, d)$. The scoring function $f(\cdot)$ thus becomes:

$$f(sc_b(q, d), \mathcal{P}_d, \mathcal{P}_{q,d}) = sc_b(q, d) + \delta(\mathcal{P}_d, \mathcal{P}_{q,d}). \qquad (10)$$

This adaptive formulation 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 [20]. In every iteration, MART trains a new tree to be added to the existing list of trees. In our setting, we start with our base score $sc_b(q, d)$ and then train additive trees over this score.

## 3.5 Attribute Instantiations

### 3.5.1 Email Corpus

While the attribute parameterization model described in the previous section can be applicable in many personal search scenarios, in this paper we focus on email search, which received some attention from the research community in recent years [11, 15, 28]. Email search is an attractive candidate for attribute parameterization since emails are private by nature, and query-email interactions do not generalize beyond a single user.

We conduct our experiments using an offline dataset derived from several months of user interaction data from one of the world's largest email search engines. We rely on a base ranking system to provide the top-$N$ results (as ranked by the base score $sc_b(q, d)$), and evaluate whether re-ranking these results using attribute parameterization features $\mathcal{P}_d$ and $\mathcal{P}_{q,d}$ can improve standard click-based search quality metrics. For more details on the dataset see Section 4.1.1.

### 3.5.2 Attribute types

In general, attributes can be considered a special case of *multi-view clustering* [8], where documents and queries are represented via several orthogonal views or aspects. While there are multiple ways to define such attributes, in this work we use a simple approach that we found to be very effective. We define three attribute types that provide independent orthogonal views of the personal content.

- *Categories* – This attribute type models the content using a fixed small set of topics or categories. It might be, for instance, an output of some textual classifier or clustering algorithm that runs over the private content [5, 17, 29].
- *Structure* – This attribute type models the content using its inherent structure, regardless of the content topic. For instance, for email corpora, a structure can be represented via structural templates [3, 29]. For personal files, it can be represented, among other options, as a file type.
- *Content* – This attribute type models the content using some summary of its actual contents, e.g., by extracting representative terms or n-grams, or using other extractive summarization methods.

With these broad definitions in mind, Table 2 summarizes the instantiation of these attribute types. Note that while it is theoretically possible to define *Categories* and *Structure* query attributes, we found that in practice simple n-gram content representation is quite effective on its own. We leave further exploration of query attribute types to future work.

# 4. EVALUATION

In this section, we conduct a suite of experiments to evaluate the impact of the proposed attribute parameterization technique on personal search quality. For experimentation, we use a search engine of a large commercial email provider.

In Section 4.1 we describe our experimental setup, data sets, and metrics design. In Section 4.2 we compare an overall performance of the query dependent and query independent attribute parameterization approaches, and a combination of these in terms of standard information retrieval metrics. We also provide a detailed analysis of the performance of the various approaches, including coverage analysis and query analysis.

## 4.1 Experimental Setup

We set up our evaluation using a standard supervised learning-to-rank framework [25]. In this section, we describe the data sets used, the supervision approach, and the evaluation metrics we used in our experiments.

### 4.1.1 Data Sets

The data sets we used in this paper are derived from the search click logs of a commercial email service. There are two data sets we used in our evaluation: a *learning-to-rank* data set and an *attribute parameterization* data set.

- *Learning-to-Rank.* This data set is used to train and evaluate the ranking functions. It is a sample from the search logs from July 2016, resulting in about 4 million queries, and an average of 4 documents per query. We use a 50/50 random split on this data to form our training and test data sets.
- *Attribute Parameterization.* This data is used for attribute parameterization. It is from the logs of the same search engine as the learning-to-rank data set. The difference is: (1) This data is "older" than the learning-to-rank data set, and does not overlap with it; (2) This data is much larger than the learning-to-rank data set. In fact, we used 3 months of un-sampled search logs from April 2016 to June 2016. As our parametrization techniques only involve simple data aggregation (see, e.g., Equations 6 and 8), it easily scales to such a large data set.

The output from the attribute parameterization data set is stored in a lookup table keyed by the attributes. We enrich the learning-to-rank data set using the parameterized attribute features by joining and aggregating them over the corresponding document and query attributes.

### 4.1.2 Click Bias Correction

We use click data as a source of ground truth labels for the supervision of the learning-to-rank algorithm. This is a standard practice in personal search applications, and in particular for email search (see, e.g., [11, 28]). This is due to the fact that the editorial judgments may not be feasible to obtain due to the private nature of the corpora. In addition, the distribution and the content of emails may vary significantly across users, and using a small set of paid raters may present a very skewed picture of the algorithm performance, biased by the population of the raters.

On the other hand, click data collected from the search logs does not suffer from this population bias. However, the click data is likely to be noisy and suffer from position and presentation biases, as observed in [28]. Therefore we apply the position bias correction in our training process, as suggested by some prior work [14, 28].

In order to quantify the click position bias, which will be used for weighting queries during training and evaluation, we perform a uniform randomization of the top results presented to a small fraction of users, and collect user click data on this randomized result set. We denote $b_k$ the bias at $k$-th rank (probability that the user will click on rank $k$ in the randomized experiment). As Wang et al. show [28], weighting the $i$-th query by $w_i = \frac{1}{b_{k(i)}}$ corrects for the position bias across queries where $k(i)$ is the clicked rank in the logs of the $i$-th query.

| | ΔMRR | |
|---|---|---|
| Attribute Type | Query-Independent | Query-Dependent |
| Categories | +0.48* | +0.80** |
| Structure | +1.56** | +1.22** |
| Content | +1.27** | +2.11** |
| All | +2.10** | +2.60** |
| **Full Model** | **+3.24**** | |

**Table 3: Overall comparison of different variants. * and ** mean the improvement is significant at 0.05 and 0.01 levels respectively.**

### 4.1.3 Evaluation Metric

The main evaluation metric used in this paper is a variant of Mean Reciprocal Rank (MRR). Given a test data set with $N$ queries, the standard MRR is defined as follows:

$$\mathrm{MRR} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{\mathrm{rank}_i} \tag{11}$$

where $\mathrm{rank}_i$ is the rank of the first clicked document of the $i$-th query. Let $w_i$ be the position bias correction described in Section 4.1.2 above for the $i$-th query, then the weighted MRR is defined as

$$\mathrm{MRR} = \frac{1}{\sum_{i=1}^{N} w_i} \sum_{i=1}^{N} w_i \frac{1}{\mathrm{rank}_i}, \tag{12}$$

which is the MRR definition that we will use in the remainder of this paper.

In our experiments, we use the metric ΔMRR that is the relative improvement of a ranking function compared to the base ranking function $sc_b(q, d)$, in terms of percentage. For example, +3.5 means 3.5% MRR improvement.

## 4.2 Experimental Results

We report our experimental results in this section. We first give the overall comparison and then analyze the feature coverage. Finally, we show some insights on what types of queries benefit the most from the attribute parameterization based features.

### 4.2.1 Overall Comparison

The overall comparison is listed in Table 3. 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.

- For individual attribute types, query-dependent features result, in general, in a higher overall improvement that the query-independent ones.
- 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

| | | QIM | QDM | FM |
|---|---|---|---|---|
| Q-I query cvg. | 99.08% | +2.10 | – | +3.24 |
| Q-D query cvg. | 66.50% | – | +3.74 | +4.31 |

**Table 4: Coverage comparison for query-dependent and query-independent features. Column 2 shows the query coverage sliced by query independent (QI) and query dependent (QD) attributes. Columns 3-5 show Covered ΔMRR for the Query Independent (QIM), Query Dependent (QDM), and Full (FM) models, respectively.**

that the selected attribute types are indeed complimentary to each other, and can provide incremental improvements.

- As a stand-alone feature, the *Categories* attribute type performs the worst for both query-dependent and query-independent features. This is in line with the fact that our categories are rather coarse grained. Both *Structure* and *Content* are much more fine grained, and achieve good results.

- Combining all the features and attribute types in the full model results in the best performance, and outperforms the baseline by +3.24. This shows the usefulness of the combination of both the query-dependent and the query-independent features.

### 4.2.2 *Coverage Analysis*

In the above, we report the overall performance improvement. However, due to the different degrees of sparsity for different attribute types, the overall improvement is impacted by the feature coverage (fraction of the corpus covered by the attribute types defined in Table 2). In this section, we analyze the results in term of the coverage.

In particular, we are interested in the query coverage. A query is deemed *covered* by a feature if there is at least one document in the query result list that has that feature populated. The query coverage for a feature is the percentage of the queries that is covered by the feature. Note that following this definition, the results of *non-covered* queries will be unaffected by the attribute parameterization features, compared to the base score. Therefore, it makes sense to focus on the covered queries, and ΔMRR for these queries in particular.

We compute the query coverage for each of the ranking functions and we also report the *Covered ΔMRR* metric on this subset of covered queries. In Table 4, we show the coverage and the Covered ΔMRR for query-dependent, query-independent and the full models. Naturally, compared with the query-dependent ones, we see that the query-independent features have a much higher coverage[2]. However, the improvement on the covered queries is smaller for the query-dependent model compared to the query-dependent model (Covered ΔMRR of +2.10 vs +3.74, respectively). The full model can leverage both of them to achieve a good tradeoff between coverage and improvement on the covered queries (+3.24).

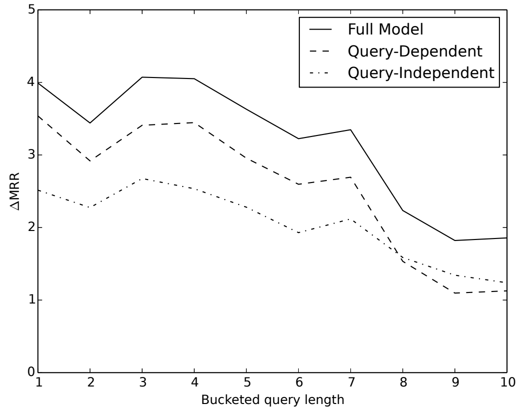


**Figure 4: Query length impact.**

| | Queries w/ n-gram | Queries w/o n-gram |
|---|---|---|
| Percentage | 67.4% | 32.6% |
| ΔMRR | +4.25 | +1.08 |

**Table 5: Queries with n-gram and without matching frequent n-grams.**

One interesting question is whether the query-independent features contribute to the model when the query-dependent features are already present. To understand this, we compare the full model vs the query-dependent one on the queries that have query-dependent feature coverage. The Covered ΔMRR is +4.31, which is significantly larger than +3.74. This shows that the query-independent features provide substantial benefits, even if the query-dependent ones are present.

### 4.2.3 *Query Analysis*

In this section, we partition the queries into different subsets based on their properties, and then study the impact of different models to gain more insights on how our proposed methods work along different query properties.

In Figure 4, we show the results on different query lengths. We partition the queries based on their length into 10 equal-sized buckets – meaning each bucket has an equal number of queries (with lower numbered bucket containing shorter head queries). In Figure 4, we see that all the methods perform the best for short head queries in the lower numbered buckets. This indicates that the attribute parameterization features are more reliable for these head queries, likely due to the larger amount of aggregated data available for these queries.

Interestingly the query-dependent model is slightly worse that the query-independent one for longer tail queries. This is probably due to our n-gram query matching technique discussed in Table 2. When the query is longer, the meaning of the longest matching frequent n-gram may drift away from the meaning of the original query, thus making the attribute parameterization features less relevant.

Longest matching frequent n-gram is the attribute type that we used for the query-dependent features. For rare queries, we may not have matched n-grams. In Table 5, we show the percentage of queries with and without frequent n-gram matches. For these subsets of queries, we show the

[2] A small minority of the documents may not have any associated attributes, therefore the query-independent coverage is not exactly 100%.

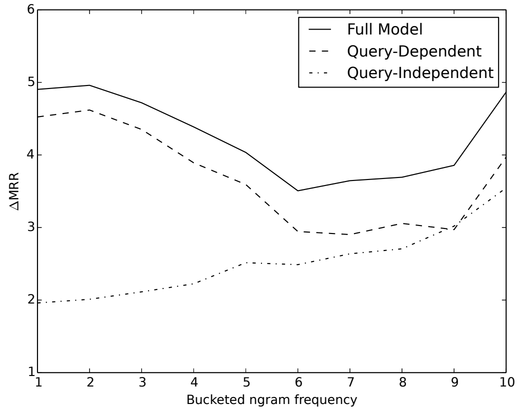


**Figure 5: Impact of the n-gram frequency.**

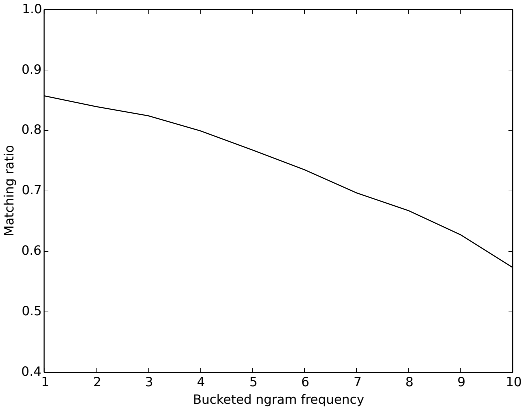


**Figure 6: N-gram frequency vs matching ratio between frequent n-gram and query.**

ΔMRR metric. It can be seen that frequent n-grams have a reasonable coverage of 67.4%. Also, the improvement of queries with matching n-gram is much larger than the ones which do not have matching n-grams (and therefore we can only utilize the query-independent attribute aggregates for these queries). This suggests that there is a large room for improvement for these long-tail queries, if we can leverage other query attributes types (like category or structure) for better coverage.

In Figure 5, we study the impact of the matched n-gram frequency. Recall, that for each query, we use the longest n-gram match. This is an approximate match between queries and n-grams. We partition the queries into 10 equal-sized buckets, such that a lower numbered buckets have queries that match lower-frequency n-grams.

For the query-independent model, we see a larger improvement on the subset of queries that match the most frequent n-grams. This indicates that the query-independent features are dominated by the most frequent queries, as a result of the aggregation process in Equation 6.

On the other hand, for both the full model and the query-dependent model, we observe a $U$-shaped curve. On the right-hand side, we have the high-frequency n-grams for which we are likely to have enough aggregate data to estimate a reliable attribute parameterization in Equation 8. On the left-hand side, we conjecture that the matched n-grams are more faithful to the original query intent. To verify this, we estimate the strength of the match as the ratio of the length of matched n-gram and query length. This number is plotted in Figure 6. This figure confirms our conjecture – as the matched n-gram has a lower frequency, the matched ratio is larger. This makes the query-dependent features more relevant.

# 5. CONCLUSIONS

In this paper we presented a novel *attribute parameterization* approach for modeling user interactions in personal search. In this approach, both documents and queries are projected into a multi-dimensional space of fine-grained and semantically coherent attributes based on content categories, structure and terms. These attributes address the problem of the sparsity of user interactions, which arises from the private nature of documents and queries in personal search.

We then use both query-independent and query-dependent attribute parameterization for feature derivation in a learning-to-rank model. We apply the resulting model to a large sample of query logs of an email search engine. Experimental results demonstrate that attribute parameterization is indeed an effective way of leveraging user interactions for improving personal search quality. Combining all the query-dependent and query-independent features results in an overall search quality gain of over 3% (as measured by the MRR metric), a highly significant increase for a test sample of this size. We also provide a detailed analysis of our technique, demonstrating high attribute coverage, and consistent improvements across different query types.

There are multiple promising venues for future work on modeling user interactions via attribute parameterization. First, we only used rudimentary n-gram based attributes for modeling user queries. As our experiments demonstrate, further gains can be achieved by more sophisticated ways of modeling query structure, including entity extraction, intent detection, synonym expansion, etc. Second, it would be interesting to further explore whether attribute parameterization can contribute to existing work on modeling semantic similarity with deep neural networks, which are currently mostly based on bag-of-words content representations. Finally, while in this work we focused on the personal search task, the proposed attribute parameterization techniques are general enough to be applied in other ad hoc retrieval scenarios, including web search.

# 6. REFERENCES

[1] S. Agarwal. Official Gmail Blog: A bit about Bundles in Inbox. https://gmail.googleblog.com/2014/11/a-bit-about-bundles-in-inbox.html, 2014.

[2] E. Agichtein, E. Brill, and S. Dumais. Improving web search ranking by incorporating user behavior information. In *29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 19–26, 2006.

[3] N. Ailon, Z. S. Karnin, E. Liberty, and Y. Maarek. Threading machine generated email. In *6th ACM International Conference on Web Search and Data Mining (WSDM)*, pages 405–414, 2013.

[4] X. Anguera, J. Xu, and N. Oliver. Multimodal photo annotation and retrieval on a mobile phone. In *1st ACM International Conference on Multimedia Information Retrieval (MIR)*, pages 188–194, 2008.

[5] R. Bekkerman. Automatic categorization of email into folders: Benchmark experiments on Enron and SRI corpora. Technical report, University of Massachusetts Amherst, 2004.

[6] M. Bendersky and K. W. Church. Priors in web search. In *SIGIR 2009 Workshop on Learning to Rank for Information Retrieval (LR4IR)*, 2009.

[7] M. Bendersky, W. B. Croft, and Y. Diao. Quality-biased ranking of web documents. In *4th ACM International Conference on Web Search and Data Mining (WSDM)*, pages 95–104, 2011.

[8] S. Bickel and T. Scheffer. Multi-view clustering. In *4th IEEE International Conference on Data Mining (ICDM)*, pages 19–26, 2004.

[9] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *7th International Conference on World Wide Web (WWW)*, pages 107–117, 1998.

[10] A. Broder. A taxonomy of web search. *SIGIR Forum*, 36(2):3–10, Sept. 2002.

[11] D. Carmel, G. Halawi, L. Lewin-Eytan, Y. Maarek, and A. Raviv. Rank by time or by relevance?: Revisiting email search. In *24th ACM International Conference on Information and Knowledge Management (CIKM)*, pages 283–292, 2015.

[12] H. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, R. Anil, Z. Haque, L. Hong, V. Jain, X. Liu, and H. Shah. Wide & deep learning for recommender systems. *CoRR*, abs/1606.07792, 2016.

[13] N. Craswell and M. Szummer. Random walks on the click graph. In *30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 239–246, 2007.

[14] N. Craswell, O. Zoeter, M. Taylor, and B. Ramsey. An experimental comparison of click position-bias models. In *1st International Conference on Web Search and Data Mining (WSDM)*, pages 87–94, 2008.

[15] S. Dumais, E. Cutrell, J. Cadiz, G. Jancke, R. Sarin, and D. C. Robbins. Stuff I've seen: A system for personal information retrieval and re-use. In *26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 72–79, 2003.

[16] J. Gao, K. Toutanova, and W. Yih. Clickthrough-based latent semantic models for web search. In *34th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 675–684, 2011.

[17] M. Grbovic, G. Halawi, Z. Karnin, and Y. Maarek. How many folders do you really need?: Classifying email into a handful of categories. In *23rd ACM International Conference on Conference on Information and Knowledge Management (CIKM)*, pages 869–878, 2014.

[18] Q. Guo and E. Agichtein. Beyond dwell time: estimating document relevance from cursor movements and other post-click searcher behavior. In *21st International Conference on World Wide Web (WWW)*, pages 569–578, 2012.

[19] Q. Guo, H. Jin, D. Lagun, S. Yuan, and E. Agichtein. Mining touch interaction data on mobile devices to predict web search result relevance. In *36th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 153–162, 2013.

[20] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, 2001.

[21] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck. Learning deep structured semantic models for web search using clickthrough data. In *22nd ACM International Conference on Information & Knowledge Management (CIKM)*, pages 2333–2338, 2013.

[22] T. Joachims. Optimizing search engines using clickthrough data. In *8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 133–142, 2002.

[23] T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting clickthrough data as implicit feedback. In *28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 154–161, 2005.

[24] M. Kamvar, M. Kellar, R. Patel, and Y. Xu. Computers and iPhones and mobile phones, oh my!: A logs-based comparison of search users on different devices. In *18th International Conference on World Wide Web (WWW)*, pages 801–810, 2009.

[25] T.-Y. Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331, 2009.

[26] M. Richardson, E. Dominowska, and R. Ragno. Predicting clicks: Estimating the click-through rate for new ads. In *16th International Conference on World Wide Web (WWW)*, pages 521–530, 2007.

[27] M. Richardson, A. Prakash, and E. Brill. Beyond PageRank: machine learning for static ranking. In *15th International Conference on World Wide Web (WWW)*, pages 707–715, 2006.

[28] X. Wang, M. Bendersky, D. Metzler, and M. Najork. Learning to rank with selection bias in personal search. In *39th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 115–124, 2016.

[29] J. B. Wendt, M. Bendersky, L. Garcia-Pueyo, V. Josifovski, B. Miklos, I. Krka, A. Saikia, J. Yang, M.-A. Cartright, and S. Ravi. Hierarchical label propagation and discovery for machine generated email. In *9th International Conference on Web Search and Data Mining (WSDM)*, pages 317–326, 2016.

[30] R. W. White, M. Bilenko, and S. Cucerzan. Studying the use of popular destinations to enhance web search interaction. In *30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 159–166, 2007.