Rank by Time or by Relevance?

Conference Paper · January 2015

DOI: 10.1145/2806416.2806471

5 authors, including:

David Carmel
Yahoo
112 PUBLICATIONS **3,248** CITATIONS
SEE PROFILE

Yoelle Maarek
Yahoo
90 PUBLICATIONS **2,294** CITATIONS
SEE PROFILE

Rank by Time or by Relevance? Revisiting Email Search

David Carmel[1], Guy Halawi[2], Liane Lewin-Eytan[2], Yoelle Maarek[1], Ariel Raviv[2]
Yahoo Labs, MATAM Park
Haifa, 31905, Israel
[1]{david.carmel,yoelle}@ymail.com, [2]{ghalawi,liane,arielr}@yahoo-inc.com

ABSTRACT

With Web mail services offering larger and larger storage capacity, most users do not feel the need to systematically delete messages anymore and inboxes keep growing. It is quite surprising that in spite of the huge progress of relevance ranking in Web Search, mail search results are still typically ranked by date. This can probably be explained by the fact that users demand perfect recall in order to "refind" a previously seen message, and would not trust relevance ranking. Yet mail search is still considered a difficult and frustrating task, especially when trying to locate older messages. In this paper, we study the current search traffic of *Yahoo mail*, a major Web commercial mail service, and discuss the limitations of ranking search results by date. We argue that this sort-by-date paradigm needs to be revisited in order to account for the specific structure and nature of mail messages, as well as the high-recall needs of users. We describe a two-phase ranking approach, in which the first phase is geared towards maximizing recall and the second phase follows a learning-to-rank approach that considers a rich set of mail-specific features to maintain precision. We present our results obtained on real mail search query traffic, for three different datasets, via manual as well as automatic evaluation. We demonstrate that the default time-driven ranking can be significantly improved in terms of both recall and precision, by taking into consideration time recency and textual similarity to the query, as well as mail-specific signals such as users' actions.

Categories and Subject Descriptors: H.3.3 [Information Search and Retrieval]: Retrieval models

General Terms: Algorithms, Experimentation

Keywords: Email Search, Ranking

1. INTRODUCTION

Recent market analysis studies have shown that in spite of the rise of communications through social media, email traffic keeps increasing [21]. In addition, Web mail services offer more free storage than in the past, with quotas that range from 15GB for Gmail to 1TB for Yahoo Mail. For most users, cleaning up inboxes is not compulsory anymore, and they adopt the "lazy" approach of rarely deleting messages. We have verified over 2 months of users' activities in Yahoo Web mail service, that 82% of users never deleted a single message. Thus, mailboxes have grown and will keep growing, storing together with useless messages, such as hotel newsletters or obsolete personal exchanges, critical information such as e-tickets or invoices that users will want to re-access at some point. Indeed, mailboxes have been used for data archiving of personal communications for more than a decade [26], and more recently for storing important machine-generated messages [12]. Folders are not of much assistance for discovering past mail, as demonstrated by a study conducted by Whittaker et al [25]. It is even worse in the context of Web email, in which it has been shown that more than 70% of users never define a single folder [14], and among those who do define folders, less than 10% are actually using them [12]. Therefore, the default discovery paradigm for retrieving past messages or attachments is search.

Unfortunately, email search remains "frequently difficult, time-consuming and frustrating" [11]. Some possible reasons for this is that email search mechanisms have not been sufficiently tailored to the specific needs of email users and to the specific structure of email messages. The task is clearly challenging as users, unlike in Web search but very much like in desktop search, will know when a relevant message has not been returned. Indeed, in email search, users usually want perfect recall as they are looking for *stuff they've seen*, to paraphrase the pioneering work by Dumais et al. on desktop search [8]. Email search strategies and email searchers' behavior have both been studied in depth [6, 23, 11, 25, 12, 13]. Yet to the best of our knowledge, there has not been any large study of email search ranking mechanisms in major Web email services. As an example, it is not clear why by default, all existing services display search results in reverse chronological order, and how this impacts precision and recall, as well as the user search behavior. Note that this "sorting by time" default view is critical since it has been shown that email searchers almost never re-sort search results [11].

In this paper, we challenge the prevalent ranked-by-time search result view in Web email and investigate whether an email-specific ranked-by-relevance view could bring value to users. Note that relevance is not totally absent from Web email search. Yahoo Mail for instance allows users to change the search results sort view from time to relevance. Yet, the

CIKM'15, October 19–23, 2015, Melbourne, Australia.

DOI: http://dx.doi.org/10.1145/2806416.2806471.

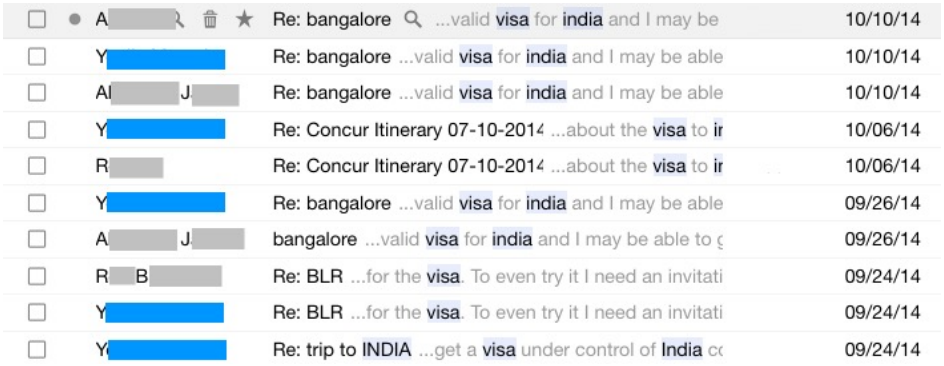

Figure 1: Time-sorted results for the query "visa to india"

sort preference toggling is quite hidden, and it most probable that the majority of users never changes the default order. Google exposes relevance ranking in small results sets: one example is the Gmail autocomplete search box [19], which dynamically predicts a few relevant messages as the user types the query; a more recent example is Inbox for Gmail[1], which lists a few "top results" ranked by relevance above the usual ranked by time of "all results". None of the major Web email services have divulged how their search mechanisms operate. However, one can safely assume that a two-stage approach is applied in which a pool of results that pass a certain relevant threshold is first identified, and then, in a second stage, these results are sorted by recency so as to rank fresher messages higher.

The main drawbacks of the chronological ordering of the results are twofold. First, it makes the discovery of older messages even harder. The older the message, the more difficult it is to remember characteristic attributes (such as its senders or differentiating terms). More specifically, Elsweiler et al. [10] have demonstrated that re-finding older email is notably difficult and that "specific support is required when the user is looking for older information". Second, chronological ordering imposes quite strict constraints for messages to qualify as relevant in the first phase. This is necessary in order to avoid embarrassing, non-relevant yet recent results from being displayed at the top of the list. As a result, recall is often degraded, contradicting the requirement for high recall, possibly at the cost of lower precision in a re-find scenario. Finding the "perfect query" for older messages becomes even harder.

Consider this motivating example taken from one of the authors' mailboxes. The searcher, to whom we refer as Y, is trying to recover the visa application form that he sent to his travel agency the last time he traveled to India. He does not remember which travel agent he worked with, nor his exact time of travel. When he issues the query "visa to india", using the default mail search ranking of his Web mail, he gets the results shown in Figure 1. Note that all personal information was obfuscated in this screen capture, as well as the mail system logo, for obvious privacy and anonymization reasons. These results match the terms "visa" and "india", in the body of the message, as indicated by the snippet match in grey, yet none satisfy his intent. They are all recent inquiries from friends and colleagues who are discussing with Y their own travel to India. In order to recover his old form, Y will need to try several reformulations, or advanced structured operator such as, "from":" or "has attachment", etc. We will revisit this example in Section 4.

In this work, we aim at demonstrating that while freshness remains indeed important in email search, ranking by time is not the optimal solution for satisfying users' needs. We argue that with a proper time and email-domain aware relevance estimation, relevance ranking will achieve better results. We investigate whether relaxing the match requirements in the first phase can improve recall, while improving precision during the second phase via a novel Learning-to-Rank (LTR) approach that considers a rich set of email-specific features. The latter range from features related to the query, the message, and the sender, to more traditional text-similarity features between candidate messages and the query.

More specifically, we pay special attention to the following types of features. First, we recognize email messages as structured documents, composed of a body and a header, where the header includes a subject line, and other fields such as sender, recipient(s) and date, as per the standard Internet message format[2].. Textual similarity between the query and a message is obviously key to relevance, yet query terms might appear in several textual fields of the message. Relevance should be affected by the type of field in which the query terms appear. To this effect, we take into account the message structure by using the BM25f score [22], which has been specifically designed for structured documents, in addition to other traditional similarity scores. Second, we propose to fully integrate in our relevance estimation the recency of a message, which we refer to as *freshness*, rather than using recency as a sorting mechanism. In addition, in the same way that usage data such as page clicks and dwell time has been shown to be critical in Web search [3], we propose to consider email actions performed by the user on each message (e.g., read, reply, forward, mark, etc.). Finally, we investigate whether more static signals, such as sender global email activity and the strength of the sender correspondence with the user, bring value to message ranking.

The contributions of our paper are three-fold:

1. We revisit email search, considering it as an independent search domain that requires its own ranking model.

[1]See the "Find emails" section in Google Support `https://support.google.com/inbox/answer/6067584`

[2]See Internet Message Format, RFC2822 and RFC6854

We challenge the traditional chronological ranking commonly used in email search, and argue that using a new email relevance model is better adapted to the user "refind" needs.

2. We introduce REX (Relevance Extended), a novel email relevance ranking algorithm, that makes use of a wide variety of features relevant to email search, including freshness, under a learning-to-rank approach.

3. We describe a full evaluation system and report our offline as well as online experiments conducted on the Yahoo Web mail system. We report on qualitative judgments conducted by professional editors on their own mailboxes as well as automatic evaluation results in a large scale privacy-preserving experiment. This is to the best of our knowledge the first experiment of that scale.

2. RELATED WORK

Email search in general, and email ranking in particular, have not received much attention by the IR research community, probably due to the lack of publicly available benchmarks of email data, which is too private and too sensitive to be widely exposed. One exception is the TREC Enterprise track dataset [6, 23], which contains about 200K email messages crawled from the W3C public mailing list archive `lists.w3.org`. In the known-item search task of the TREC 2005 enterprise track, participants were challenged with (query, message-id) pairs where their task was to rank the given message at the top of their search results. Participants took different approaches at integrating the message text with the message meta-data, as well as balancing between the various message fields. Macdonald et al. [17] combined Web features with email features using a field-based weighting model, investigating the retrieval performance attainable by each field and whether field evidences should be combined or not. Craswell et al [7] applied the BM25f formula [22] in order to account for the message meta-data in the final message score. Ogilvie and Callan [18] took a language model approach in order to combine evidences from the text of the message, the subject, the text of the thread in which the message occurs, and the text of messages that are in reply to the message. A comprehensive overview of this track can be found in [6].

The same dataset was also utilized in the email discussion search task of the TREC 2006 enterprise track [23]. In this task, participants searched for email discussions that are relevant to a given topic and added pro or con arguments.

The W3C dataset was also used in a few more studies. Yahyaei and Montz [27], showed that maximum entropy can be successfully applied in order to estimate feature weights in known-item email retrieval, leading to statistically significant improvements over several baselines. Weerkamp et al.[24] studied the usage of contextual information to improving email search. They expanded queries using several sources, such as threads and mailing lists. Additionally, their experiments showed that using query-independent features (email length, thread size, text quality), implemented as priors, resulted in further improvements.

Abdelrahman et al. [1] experimented with email search over the Enron corpus[3], an enterprise email corpus dating from 2003 that was released by the Federal Energy Regulatory Commission after the Enron investigation. They proposed a scoring function that derived information from the email subject, content, and sender. However since the Enron dataset does not include queries and relevance judgments, the authors generated 300 synthetic queries (using topic words related to Enron) and used a sample of 35 queries for testing. Relevance was estimated by three judges according to given guidelines. This approach is clearly far from a real life scenario as both queries and relevance judgments were not originated from the mailbox owners.

None of the previous work described above investigate ranking methods in a real online setting, and in particular in commercial Web email services, which serve hundreds of millions of users. A rare exception to this, is the work by Aberdeen et al. [2] on Gmail priority inbox. The authors predict the "importance" of a message, which is defined as the likelihood that a user will act on it. Their learning procedure considers social features that are based on the degree of interaction between the sender and the recipient, content features that identify important terms in the message, thread features that measure the user's interaction with the message thread, and label features which examine the labels that the user applies to the message. Recently, Kooti et al. [13] tried to predict the reply time and length for email messages, based on the stage of the conversation, user demographics, and use of portable devices. They found that as users receive more email messages in a day, they reply to a smaller fraction of them, while using shorter replies. However, their responsiveness remains intact, and they may even reply to emails faster.

While these recent works do not directly pertain to email ranking, the message importance score [2], or the predicted reply time [13], could potentially be utilized as additional features for message ranking. Similar to these works, we do consider content, sender, and action features, but in a different flavor, as our task is drastically different. To the best of our knowledge, our work is the first that discusses email ranking in the context of a large Web mail service, and experiments with real queries and real messages, considering not only editorial judgments (conducted by professional editors on their own mailboxes) but also real-user implicit satisfaction derived from clicks.

3. RANKING MODEL

Our model is based on a standard two-phase retrieval process. In the first phase, that we refer to as the "matching phase", we retrieve a pool of messages qualified as potentially relevant to the query. We use here standard IR text matching mechanisms and do not differentiate between candidates within the pool. The second phase ranks these messages using a rich set of features. The goal of the first phase is to improve performance, so that not all messages in the inbox need to be considered by the more expensive relevance estimation of the second phase.

We will get back to the first phase at the end of the section, and focus in the meantime on the second phase, the ranking phase that introduces a new mail-specific relevance estimation. Our ranking model is based on a large set of features covering the message (body and meta-data), its similarity to the query, and characteristics of the participants involved in the mail correspondence (sender and recipients). We de-

[3] `http://www.cs.cmu.edu/~enron/`

scribe next the set of features used by the ranking function as well as our LTR approach.

3.1 Ranking features

We detail below the four types of features that we consider, which relate to an individual message, senders, recipients, and the message/query similarity. These features are listed in a summarized view in Table 1.

3.1.1 Message

The retrieval unit we consider here is a single message. We could not consider a thread as a retrieval unit as the current version of Yahoo Web mail service that hosted our experiments does not support such a feature. Note however that we do recognize whether or not a message is part of a thread and reflect this fact in a message feature as discussed later.

Message features describe message characteristics that are independent from the query. One critical type of message features is the message freshness that we account for in different time units. An important characteristic of our work is to consider message freshness as a set of features in our relevance model, rather than as a sorting criterion. As shown in many other domains, temporal aspects of the documents can significantly impact their relevance to many query types [16, 9].

More specifically, we consider the freshness of a message in days, weeks, months and years, using the formula given below. We define

$$fresh_g(M) = e^{-\tau_g \cdot age(M)}, \quad (1)$$

where τ_g is a tunable decay factor and $age(M)$ is the message age measured in seconds. The different values of τ_g are set according to the granularity g of the message age with $g \in \{days, weeks, months, years\}$.

The second set of message features reflects the user's actions on a message. More specifically, we consider *replied, forwarded, draft, flagged, seen (read), spam, ham*, as binary features that denote whether or not the user conducted the respective action on the message.

An additional set of features corresponds to message attachments when they exist. The attachment features reflect the type of the attached object (e.g. document, image, audio, video, calendar invitation, etc.) and size range. An additional set of feature, that we refer to as "folder", indicate the type of folder to which the message belongs, where the message can belong to a predefined system folder such as inbox, sent, draft, trash, spam, chats, etc., or to a personal user-defined folder (we do not differ here between different user-defined folders as this data was not available in our experiment settings). Finally, the last set of message features represents its "exchange type", reflecting whether the message is a reply or a forwarded message, and whether or not it is part of a thread correspondence (clearly, a message can be both a reply/forward, as well as part of a thread).

3.1.2 Sender

We detail here features that relate to the sender of the message. The sender features are divided into two sub-types, that we refer to "vertical" when the feature pertains to the sender activity with respect to the user's mailbox, and as "horizontal" when it pertains to the sender activity across all users' mailboxes. We use a single vertical sender feature that represents the volume of communication between the user and the sender. The more frequent and the more recent (as per a decay factor we discuss later) the communication is, the higher the value of this feature. We compute it per user, as a same sender will have different exchange patterns with different users. We define it as:

$$P_u(s) \cong \frac{M^T_{u,s}}{M^T_u} \cdot \frac{M^O_{u,s}}{M^O_u} \quad (2)$$

The score $P_u(s)$ to sender s is thus relative to user u, where $M^T_{u,s}$ counts the total number of messages between u and s (both inbound and outbound), M^T_u counts the total number of messages between u and all its contacts (both inbound and outbound), $M^O_{u,s}$ counts the total number of outbound messages from u to s, and M^O_u counts the total number of outbound messages from u. The intuition here is that the user's outbound messages is a valuable signal in estimating the importance of a sender. Taking it into account prevents us from assigning an overwhelmingly high score to a mass sender (e.g. a robot sending a hotel newsletter for instance).

In addition, we decrease the influence of a message on the different counters in Equation 2, as it gets older, by using a decay factor. A recent message will contribute a score of 1 to the relevant counters, however, an old message will contribute only $\alpha^{age(M)}$, where $0 < \alpha < 1$ is a tunable parameter (we empirically set α to 0.92 in our experiments).

A special case of sender-user connection is reflected in the so-called "self correspondence" that indicates whether the sender is the user himself. Note that such cases are not rare, since as mentioned before, mailboxes have been used for long for data archiving.

The horizontal sender features are computed across all users, and relate to the global activity of the sender. Features in this set account for the volume of the sender inbound and outbound traffic (where the sender traffic is measured with respect to the entire user population in the system), the average number of URLs appearing in the sender messages, and the average number of recipients of these messages. In addition, we aggregate mail actions performed on the sender's messages over all users. For each action type, we provide the ratio of the sender messages triggering this action. Examples of action types include *delete, read, reply, forward, move-to-folder, mark-ham/spam, flag/star,* etc. We note that a large part of the horizontal sender features are good indicators as whether the sender is a robot sending a machine-generated message or a human actually composing the messages. For example, messages addressed to a very large number of recipients, or containing a high amount of URLs in their content, are likely to be machine generated, while messages triggering reply actions are commonly sent by humans.

3.1.3 Recipient

Recipient features aim at capturing the strength of the connection between the recipient and the message. These features differentiate between messages that are targeted to the user as the sole recipient, or as a part of a larger distribution list, under the intuition that a message sent only to the user might be more important to him/her. We parse the message header fields, namely `To:, CC:, BCC:` and reflect this information in the following recipient features: *isInTo*, *isInCC* and *isInGroup*.

Type	Sub-Type	Feature (or features set) Name	Description
Message	Freshness	time by days/weeks/months/years	message age by respective resolution
	User Actions	replied	message was replied
		forwarded	message was forwarded
		draft	message is saved as draft
		flagged	message is flagged by star
		seen	message was read
		spam	message was marked as Spam
		ham	message was marked as Ham
	Attachment	has attachment	message has an attachment
		attachment type	set of binary features for attachment type
		attachment size	set of binary features for attachment size range
	Folder	folder type	set of binary features indicating the folder of the message (specific system folder or a personal folder)
	Exchange Type	reply/forward	binary features indicating if message is a reply or a forward
		in thread	message is part of a thread correspondence
Recipient		in To/Cc/Group	set of binary features corresponding to wether recipient is in To/Cc/Group (group is Bcc or mailing list)
Sender	Vertical	sender-user connection	strength of correspondence between sender and user
		self correspondence	sender is the user (binary, message was sent from the user to himself)
	Horizontal (over all users)	sender outbound/inbound traffic	set of binary features for sender's outbound/inbound traffic range
		sender urls	set of binary features for range of urls num in sender's messages
		sender recipients num	set of binary features for range of recipients num in sender's messages
		sender-users actions	ratio of the sender's messages on which a specific action was performed
Query Similarity		BM25f	BM25f similarity between message and query
		tf-idf	tf-idf similarity between message fields and query
		coord	fraction of query terms found in the message

Table 1: Set of features used by the ranking function.

3.1.4 Message-Query Similarity

We consider three main features for representing the connection between the message and the query, all evaluating the textual similarity between the two: $BM25f$, $tf.idf$, and *coord* as defined below.

BM25f [22], is a state-of-the-art ranking function that measures the textual similarity between a query Q and a structured document D that consists of several textual fields. It is defined as

$$BM25f(Q,D) = \sum_{q \in Q} idf(q) * \frac{\hat{tf}(q,D)}{k + \hat{tf}(q,D)}$$
$$\hat{tf}(q,D) = \sum_{f \in F} w_f \frac{tf(q,D_f)}{(1-b_f) + b_f \frac{l_{D_f}}{avl_f}}, \quad (3)$$

where $idf(q)$ is the inverse document frequency of the query term q in the user mailbox, and k is a tunable parameter. F is the set of document fields, $tf(q,D_f)$ is the term frequency of the query term within the field D_f, l_{D_f} and $avgl_f$ are the document field length, and the average document field length, over all documents respectively. Finally b_f and w_f are the field parameters. We consider both unigrams and bigrams as query terms. We extend here the definition of a bigram, and consider it both as two consecutive query terms, as well as two consecutive query terms found (unordered) within a window of 5 terms in the respective field. Each of these cases, consecutive or within a window, is reflected as a separate term in the sum.

In our context, D represents as message, and the fields f are its textual message fields. More specifically, we consider the header fields (i.e., `Subject, From, To, CC`), the attachment name and the attachment content when appropriate, and the body of the message. The scoring function also leverages, for each field f, its relative importance weight w_f and the normalization parameter b_f. For each message field f, we tune the pair (w_f, b_f). In addition, the k parameter is used to normalize the term frequency scores $\hat{tf}$. All parameters of the $BM25$ formula were tuned using a small random subset of the datasets we experimented with.

The second feature is **tf.idf** that we use under the following variant:

$$tf.idf(Q,D_f) = \frac{\sum_{q \in Q} tf(q,D_f) \cdot idf(q)}{l_{D_f}} \quad (4)$$

where l_{D_f} is the length of the message field.

There is a major difference between $BM25f$ and $tf.idf$. The first weights the query terms based on their distribution over the message fields, while the second measures the similarity of each field to the query independently of the others. We provide the $BM25f$ score of the whole message, and the independent $tf.idf$ scores of all message fields, as input to the LTR learning module that we describe below.

Lastly, we consider the **coord** feature, reusing Lucene (`http://lucene.apache.org/`) terminology, which defines $coord(Q,D)$ as the fraction of query terms in query Q that occur at least once in the message D, in any of its textual fields.

3.2 Learning to Rank

Our LTR procedure is an online variant of SVMRank [4], which searches for a linear weight vector that aims to rank, for each training query, the clicked message higher than other top scored messages of the query. Our training dataset consists of a large scale random sample of real user queries issued on Yahoo Web mail service, as described in Section 4. All users voluntarily opted-in for being part of

such a research experiment. For each query, we retrieve in the first phase of our model, up to 100 candidate messages from the searcher's mailbox, as well as the messages clicked by the searcher[4].

The training algorithm begins with a zero-weight vector and updates it for each example using the AROW online learning procedure [5], which showed comparable performance to SVMRank. Specifically, for each example, our algorithm first ranks the retrieved messages using the current scoring function. Then, it selects message pairs consisting of the feature vector v_c of the clicked message and of the feature vector v_m of each of the top K ranked messages. Following the original SVMRank algorithm, for each such pair, the algorithm generates the difference vector $v_c - v_m$, as a training example for the linear ranker.

The optimized parameters we used, based on an independent validation set, are the following: 5 training rounds, K set to 10, and the AROW hyper-parameter r set to 1.

3.3 First Matching Phase

The search process begins with the first phase, which retrieves a pool of candidate messages that match the user's query. We consider two approaches for the first matching phase. In the first approach, we demand a strict match between the query and the message: all query terms are required to appear in at least one of the message fields. This is also the default approach that is currently used by the live mail system on which we conducted our experiments. In the second approach we relax the matching constraints by requiring only a partial match between the query and the message: at least one query term should appear in the message content.

The strict match seems to be preferred when a chronological sort is applied in the second stage. Otherwise, partial match may end with non-relevant messages containing only a few insignificant query terms that will be ranked high due to their recency. Our hope here is that with an appropriate second stage ranking based on relevance estimation, no embarrassing cases would surface to the top of the list, even when applying the relaxed first matching phase.

In the second phase, the messages are ranked and returned as results to the user. The ranking method we experimented with is based on the trained ranking function learned by our LTR process over the training data. Our ranking score, which we coin REX (Relevance Extended), uses the tuned weight vector in order to linearly combine all message feature scores into a final message score used to rank the list of messages. We will discuss later the effectiveness of REX in both types of first-phase approaches.

4. EXPERIMENTS

In this section we present the large scale experiments we carried on Yahoo Web mail service. We describe the experiments setup, the datasets we used, as well as the results we achieved. We divide our experiments in two parts. The first part consists of a quantitative automatic evaluation, comparing our ranking to the current time-based ranking in the production system which we use as baseline. For both systems, the same strict match approach is used to generate the pool of candidates, which allows us to evaluate the contribution of the second phase of reranking. This automatic evaluation was performed using two datasets extracted from Yahoo email service query log; the Web dataset contain email queries of regular users while the Corporate dataset contain queries of Yahoo employees.

The second experimental part consists of a qualitative editorial evaluation that was performed on the Corporate dataset by professional editors who conducted judgments on their own corporate mailboxes. In this second set of experiments, we relaxed the matching constraints of the first phase before applying our REX ranking.

[4]When several messages were clicked by the searcher we only consider the most recent clicked one while ignoring the rest.

4.1 Automatic evaluation

The automatic evaluation was performed in order to evaluate the performance of REX, our relevance ranking algorithm, as compared to the default chronological ranking view deployed in production of the Yahoo Web mail service. Both ranking methods (REX and chronological) were run on the same system, where the matching process conducted in the first phase required a strict match between the query and the message as mentioned at Section 3.

Due to the different nature of the two datasets, learning was performed separately for each, using the same system and set of features. Furthermore, there is a major difference between the search activity of users in these domains. For the Corporate email dataset, we followed the corporate privacy guidelines and could collect a large dataset of 100,000 queries from a few thousands opt-in users. For the Web mail dataset, due to privacy limitations, we collected queries only from users who opted in for the research study. This limited us to 10,000 queries originating from these opt-in users. In both datasets the queries were collected over a period of one month.

For the Corporate dataset, LTR was performed over a training set of 75,000 queries randomly selected (leaving the rest for testing), while for the Web email dataset, LTR was performed over 10,000 queries, using 10-fold cross validation, due the smaller size of the dataset.

Our goal here is to evaluate the quality of our re-ranking process as compared to the default chronological ranking. Precision is evaluated using two performance measures: MRR and *success@k*. The MRR measure stands for the Mean Reciprocal Rank score across all queries, that is, $\frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{r_i}$, where Q is the set of all queries, and r_i is the rank of the clicked message for query i. The *success@k* measure is the percentage of the queries for which the clicked message was ranked within the top-k results.

The baseline established for our evaluation is the time-based ranking used in production by the leading Web mail service, referred as *Time*. In order to evaluate the contribution of the different features we used for relevance ranking, we divided them into different sets, adding them one after the other to the learning process. For each additional set of features added to the system, new weights were learned following the LTR procedure and performance measures were derived following an additional evaluation round. The feature sets were added by order of direct relation to the message, from its content, through actions it triggered, and up to features related to its sender. More precisely, the features were divided into four sets:

- *freshness* (see Table 1, message/freshness). Taking this set only results in a ranking very close to that of *Time*.

Algo	*MRR (+lift %)*	*success@1*	*success@3*	*success@5*	*success@10*
Corporate email dataset					
Time	0.3722	0.2238	0.4213	0.5416	0.7037
REX$(fresh. + sim.)$	0.4261 (+14.48%)	0.2748	0.4887	0.6028	0.7509
REX$(fresh. + sim. + actions)$	0.4550 (+22.24%)	0.2999	0.5253	0.6421	0.781
REX$(fresh. + sim. + actions + sender)$	0.4548 (+22.19%)	0.2994	0.5263	0.6419	0.7837
Web email dataset					
Time	0.3717	0.2282	0.4290	0.5264	0.6783
****REX***$(fresh. + sim.)$	0.3785 (+1.81%)	0.2316	0.4406	0.5419	0.6909
REX$(fresh. + sim. + actions)$	0.4238 (+14%)	0.2711	0.4925	0.6004	0.7436
REX$(fresh. + sim. + actions + sender)$	0.4258 (+14.55%)	0.2731	0.4959	0.6000	0.7427

Table 2: Automatic evaluation on the Corporate and Web email datasets: MRR and success@k values for search result sets ≥ 30. The MRR lift for REX is computed with respect to the chronological ranking baseline. All *REX* results, except those in the line marked by asterisk, are statistically significant better than Time results (two-tailed paired t-test, $p < 1.0e\text{-}8$).

- *message-query similarity*, which contains features based on the textual similarity between the query and the message, taking into account both the content of the message and its metadata fields (see Table 1, query similarity). Taking freshness and this feature set together results in a ranking model that is very close in spirit to previous work on email ranking [7, 17] which mostly integrated textual similarity and message freshness.

- *user actions*, which corresponds to the various actions the user performed on the message, which should reflect its importance to the user (see Table 1, message/user actions).

- *sender*, which characterizes the sender via horizontal features computed across the whole set of users (see Table 1, sender/horizontal), as well as the vertical features representing the correspondence between the sender and the user (see Table 1, sender/vertical).

The evaluation results of both datasets are summarized in Table 2. Each row for the REX algorithm takes into account an additional set of features, as denoted in the table. We experimented with only queries that generate a match pool of at least 30 results in the first phase, in order to verify the effect of reranking, which are clearly less visible in very short result sets.

In the next experiment we considered smaller as well as larger match pools. We used different thresholds τ for the minimal size of the search results, where τ ranges over the values $\{10, 20, 30, 40, 50\}$. That is, for each value of τ, we performed an evaluation round over 10,000 queries with search result sets of size $\geq \tau$. The performance of REX, with full feature set, and the chronological ranking, as a function of τ which controls the minimal size of the search results, are summarized in Figure 2.

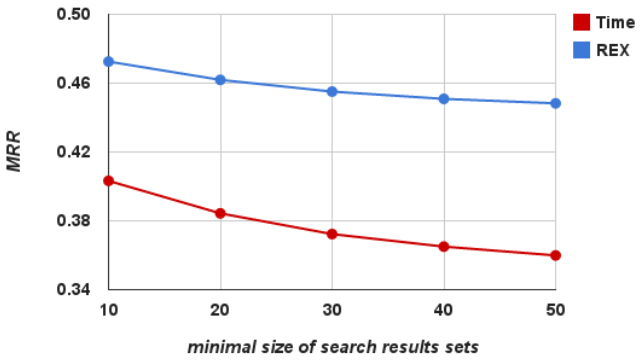

Figure 2: MRR of REX and Time, as a function of τ, in the corporate email dataset.

4.2 Discussion

There are several interesting insights that can be derived from the experimental results. At first, it is clear that the REX ranker significantly outperforms the existing chronological ranker for email search results, both in the Corporate and in the Web email datasets. The improvement in both domains, based on the automatic evaluation on a large-scale test set, is statistically significant, and is larger than 22% in MRR, in the Corporate domain, and larger than 14% in the Web domain. Moreover, as Figure 2 shows, the relative improvement increases as more messages in the user mailbox match the query, growing from 17% for small result sets of at least 10 messages, to 24% for result sets with at least 50 messages. Note that the MRR naturally decreases as the result set grows, yet it does not deteriorate as fast as with time-sorted results.

The relative contribution of the feature sets to ranking can be seen in Table 2. While the similarity features give an increase in MRR of more than 14% in the Corporate domain (on top of the freshness features), they almost do not contribute to the Web mail dataset, probably due to noisier content that is typical to that domain.

The user actions features contribute an additional increase of 8% in the Corporate domain, and of 12% in the Web domain. This can be easily explained by the large difference in the activity levels of the users in both sets. For example, more than 50% of the messages in the Corporate email dataset are reply messages, compared to less than 30% in the Web email domain. This is partly due to the larger number of threads in Corporate dataset compared to the Web dataset: about 30% of the messages in the Corporate dataset were part of threads, compared to less than 20% in the Web dataset.

Surprisingly, the sender features do not bring further improvement in both domains. This is counter-intuitive as we expected these features, which represent the sender char-

acteristics and its relation with the user, to have a significant impact on relevance estimation. As noted in Section 3, many of the sender features are good indicators as whether the message is machine generated or personally composed (i.e., the sender is a robot or a human). One interpretation could be that some senders send a mix of important and not important messages. Yet, we believe it is worth continuing exploring other models for sender representation, maybe at finer levels of granularity, and reserve this for future work.

Other features listed in Table 1, that did not bring much value, include the message exchange type, message attachment and message folder features, as well as the set of recipient features. Considering for example the recipient features, one could have expected that user appearance in the `To:`, rather than in the `CC/BCC:` fields of a message, would impact the message importance. However, this is not the case, as it seems that important information that the user would like to re-access can equally be found in messages that were not addressed directly and personally to him. An interesting difference arises from this feature, with respect to the nature of the two domains. While in the Web domain the user appears in more than 60% of the messages in the `To:` field (and only in 5% in the `CC:` field), in the Corporate domain the user appears in only 40% of the messages in the `To:` field in (and in 14% in the `CC:` field).

Looking at the REX ranking models learned in both domains, we can see that both models are very similar in terms of the relative weights assigned to the model features. In both models the freshness features are the most important, as expected, then user actions, followed by the textual similarity to the query, and finally the sender features. For freshness, the most important feature is the message age in years, then in months, weeks and days. For the user actions features, the most important one is *Seen* (whether the message was opened), then *Forwarded*, *Spam*, *Flagged*, *Replied*, *Draft*, and *Ham*. For the similarity features, *Coord* is the most significant one, then the $tf.idf$ similarity of the fields *From*, *Body*, *Subject*, *Attachment*, and the *To* fields. Finally the $BM25f$ similarity score.

Interestingly, different natures of search usage can also be observed when looking at the fields similarity in both domains. For example, the `From:` field was relevant to the query in almost 30% of the messages in our Web dataset, while being relevant in less than 20% of the messages in the Corporate dataset. This could indicate that in Web email, users use more often the name of a contact to retrieve a message from the respective sender. On the other hand, we observe a larger number of hits on the subject field in the Corporate dataset compared to the Web email dataset (22% compared to 17%). Finally, the low significance of the sender features for ranking is also reflected by their low relative weights in both domains.

4.3 Editorial evaluation

Given that the first matching phase has a direct impact on recall, we decided to explore a less strict matching policy. As discussed earlier, relaxing matching constraints, when using pure chronological ranking, would have negative effects on quality – non-relevant recent messages containing only a few query terms may be pushed on top of the list. However, our hope is that we can allow softer constraints and still not hurt quality thanks to our REX ranking method.

To this effect, we conducted a manual evaluation, with the help of professional editors who were given directives on searching their own mailboxes in the corporate settings. Our editorial evaluation included 20 professional editors. They were all asked to issue 25 queries on their own mailboxes. Each of the 25 queries had to match a given pattern, which specified the number of terms as well as their type (e.g., `<sender name>, <subject word>, <body word>`). The patterns were defined so as to cover most possible query scenarios as found in our query log. Editors were directed to formulate their own queries, in full freedom, as long they match one of the required patterns. In addition, they were asked to add a description of the associated intent, in order to describe the message they wish to re-find, before issuing their queries.

Using searcher-generated queries is a key requirement in the context of email search, as only the mailbox owner knows what his or her mailbox might cover. This differentiates our work from previous work such as [1], which uses synthetic queries over the Enron corpus. Examples of query patterns are given in Table 3, while examples of real editorial queries instantiating these patterns with associated intents are shown in the three left columns of Figure 3. Note that the editor queries may include misspelled or redundant words. We allowed such queries as they do happen in mail search and are a common source of frustration when no results are returned.

Query Pattern Examples
<sender name>
<sender name> <body word>
<subject word>
<subject word> <subject word>
<subject word> <subject word> <body word>
<sender name> <subject word> <body word>
to:<recipient name> <subject word>
from:<sender name> <body word>
<sender name> <subject word> <redundant word>

Table 3: Editorial evaluation - query patterns

Each query was run on two systems: one using the existing chronological ranking deployed in the Yahoo Web mail system we experiment with, and the other using REX, on top of the relaxed first phase match we evaluate here. Note that due to environment constraints in our Web mail system, we could not change the ranking in the live system. Instead, we compared the two systems as a whole, the existing live time-based ranking, vs. REX on top of a relaxed phase one.

For each system, the editors had to identify the rank of the most relevant result, if any, hopefully the message the editor had in mind at query time. In addition they were asked to judge other relevant results, if any. Each message was labeled according to three relevance levels: most significant (assigned to at most one result), related, and unrelated. Examples of result assessments made by editors for their own queries are presented in the right-side columns of Figure 3.

The ranking algorithms were evaluated using MRR and *success@k*, as before. Furthermore, the identification of the relevance level of the results allowed us to use two additional measures: NDCG@k (Normalized Discounted Cumulative Gain), and *p@k*. The NDCG@10 measure was computed over the 10 first results, using 3 relevance scores $\{0, 1, 3\}$ for, unrelated, related and most significant results, respectively. Tables 4 and 5 summarize the results of the editorial

Pattern	Query	Intent Description	Algo A Most relevant	Algo A Related	Algo B Most relevant	Algo B Related
<sender name> <body word>	Lila dress	Searching for the disscussion with Lila about the dress of the party	4	1,2,3,5	1	2,3,4,5,6
<subject word>	Reports	Trying to pull up my most recent weekly report to copy as a template	4	1,2,3,5,6,7	>10	1,2,3,4,5,6,7,8,9,10
<subject word> <body word>	Top Holiday	Looking for the spreadsheet for the Top Holiday	>10	1, 3, 4, 5, 6, 7, 8, 9, 10	3	1, 2, 4, 5, 6, 7, 8, 9, 10
to:<recipient name> <subject word>	to:spence KE	Invite to KE meeting	Not found	1,2,3	4	1,2,3,5,6,7,10
from:<sender name> <body word>	from:Gaylon highest-priority	Want to know which project was the highest priority	2	1	2	5
<sender name>	Christina	Looking for the most recent email from Christina	1	2, 3, 4, 7, 8	1	2, 3, 4, 6, 8, 9, 10
<sender name> <body word> <body word>	Marta terri schedule	Want to know when did Marta schedule training for LL	2	1	2	1,3,4,5,6

Figure 3: Examples of real editorial queries and ranking results

evaluation. All measures show significant and sustainable improvement of *REX* over *Time* (40.6% improvement in MRR and 34.6% improvement in NDCG@10).

Algo	*MRR (+lift %)*	*succ@1*	*succ@3*	*succ@5*	*succ@10*
Time	0.3629	0.2360	0.4140	0.5280	0.6560
REX	0.5105 (+40.65%)	0.3580	0.6000	0.7020	0.8260

Table 4: Editorial evaluation - MRR and success@k values of Time and REX. All *REX* results, are statistically significant better than Time results (two-tailed paired t-test, $p < 0.05$).

Algo	*NDCG@10 (+lift %)*	*p@1*	*p@3*	*p@5*	*p@10*
Time	0.4936	0.5540	0.4420	0.3920	0.2962
REX	0.6647 (+34.66%)	0.6960	0.6073	0.5352	0.4244

Table 5: Editorial evaluation - NDCG and p@k values of Time and Rex. All *REX* results are statistically significant better than Time results (two-tailed paired t-test, $p < 0.05$).

In addition to providing relevance judgments, the editors were given an opportunity to provide some feedback on their own impression from the two ranking systems. Most of the returned responses were very positive regarding the REX ranker. Examples include "... Sometimes, I had the feeling that Algo. B was really reading my mind to put in the first place exactly the email message I was thinking of", and "...Today, after I ran it again, it was not that much impressive, but still I have the feeling it was the type of search that gave me the best results...". Given how experienced these editors are with email search and with assessing search results, we found their feedback to be most encouraging.

4.4 Revisiting our Motivating Example

We revisit here our motivating example shown in Figure 1 in the Introduction. User Y issued the same query "visa to india" on his mailbox, but this time selecting our REX relevance ranking instead of the default time ranking. Note that at the time of writing, the REX ranking on top of the strict matching phase is fully implemented in production, but not deployed yet to all users, in our Web mail service. However it can be invoked on demand via a hidden parameter for experimentation purposes. A screen capture of the results Y obtained is shown in Figure 4. We note that older results from 2013 suddenly pop up. The third one, which contains an attachment as indicated by the paper clip icon, turns out to be the perfect one containing the actual visa form that Y was seeking. The signals that played a significant role here were the previous actions of Y on this message (reading, replying, folding), the textual similarity between the query and the subject field, and the existing attachment. Note also that there was no noticeable difference in response time between the default existing time-based search and the REX-based experimental search, as our ranking model naturally scales to Web mail requirements, thanks to its two-phase approach.

5. CONCLUSIONS AND FUTURE WORK

While some leading Web mail services return search results sorted by relevance, either in short "top results" like in Gmail, or as a non-too-visible feature in Yahoo mail, search results are still predominantly ranked by default by time in the leading Web mail services. In this paper, we challenge this chronological sort, as a sort of anachronism in light of the progress in relevance ranking. Yet, we do recognize the importance of freshness by integrating it in a mail-specific relevance model. While the set of features we introduce are not all novel (freshness for instance has been used in Web ranking), this is, to the best of our knowledge, the first time a mail-specific relevance model is described publicly, applied and evaluated, in production in a real Web mail service, under the hard performance requirements of Web-scale mail systems.

We detailed the experiments we conducted with real users and demonstrated that our model performs significantly better in terms of quality, as compared to the default existing time-based ranking in the Web mail service. We showed significant improvements in multiple settings, using the same system with Web mail users as well as corporate users of the same service, including professional editors. We shared the insights we derived from our extensive study and experiments. Some were expected such as the importance of textual similarity across message fields and of actions conducted on message. Others were more surprising such the poor influence of the message sender importance. We were particularly pleased by the evidence that the larger the result set, the more significant the improvement is.

This encourages us to believe that now is the time to depart from the old date sort in favor of more modern relevance ranking. We believe that Web mail users should be able to re-find past messages without having to play with advanced operators, which, like in Web search, are typically ignored by the overwhelming majority of Web mail users.

We plan to continue improving our model considering more signals as they become available, investigating the handling of threads as full-fledge retrieval units. We would also like to verify whether personalizing the ranking models, either by user, or by type of user, in case of sparse data, would bring value. We hope that this work will open new research directions for mail research, especially around the impact of such ranking on user engagement. Chronological ranking

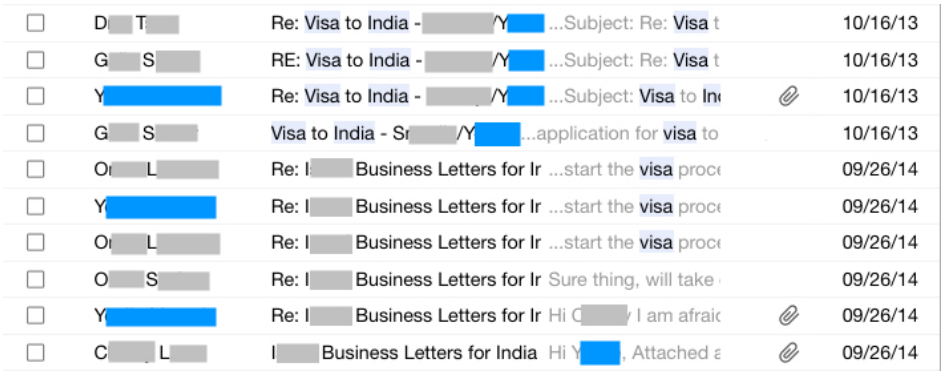

Figure 4: REX-sorted results for the query "visa to india"

dictates users to re-find their target in traversal mode that is based on the message submission time, while relevance ranking might educate users to better express their information needs. In the long term, after mail searchers get sufficiently used to such relevance ranking, we would like to verify whether it affects the way query formulation, users' search strategy, as well as the overall way users interact with the mail system.

6. REFERENCES

[1] S. AbdelRahman, B. Hassan, and R. Bahgat. A new email retrieval ranking approach. *International Journal of Computer Science & Information Technology*, 2(5), 2010.

[2] D. Aberdeen, O. Pacovsky, and A. Slater. The learning behind gmail priority inbox. In *LCCC: NIPS 2010 Workshop on Learning on Cores, Clusters and Clouds*, Vancouver, BC, Canada, Dec 2010.

[3] R. Baeza-Yates and Y. Maarek. (big) usage data in web search. In *Proceedings of SIGIR*, Portland, Oregon, USA, 2012.

[4] Y. Cao, J. Xu, T.-Y. Liu, H. Li, Y. Huang, and H.-W. Hon. Adapting ranking SVM to document retrieval. In *Proceedings of SIGIR*, Seattle,WA, Aug 2006.

[5] K. Crammer, A. Kulesza, and M. Dredze. Adaptive regularization of weight vectors. *Machine Learning*, 91(2):155–187, 2013.

[6] N. Craswell, A. P. de Vries, and I. Soboroff. Overview of the trec 2005 enterprise track. In *TREC*, volume 5, pages 199–205, 2005.

[7] N. Craswell, H. Zaragoza, and S. Robertson. Microsoft cambridge at trec 14: Enterprise track. In *TREC*, 2005.

[8] 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 *Proceedings of SIGIR*, Toronto, Canada, 2003.

[9] M. Efron and G. Golovchinsky. Estimation methods for ranking recent information. In *Proceedings of SIGIR*, pages 495–504. ACM, 2011.

[10] D. Elsweiler, M. Baillie, and I. Ruthven. What makes re-finding information difficult? a study of email re-finding. In *Proceedings of ECIR*, Dublin, Ireland, 2011.

[11] D. Elsweiler, M. Harvey, and M. Hacker. Understanding re-finding behavior in naturalistic email interaction logs. In *Proceedings of SIGIR*, Beijing, China, 2011.

[12] M. Grbovic, G. Halawi, Z. Karnin, and Y. Maarek. How many folders do you really need? classifying email into a handful of categories. In *Proceedings of CIKM*, Shanghai, China, 2014.

[13] F. Kooti, L. M. Aiello, M. Grbovic, K. Lerman, and A. Mantrach. Evolution of conversations in the age of email overload. In *Proceedings of WWW*, 2015.

[14] Y. Koren, E. Liberty, Y. Maarek, and R. Sandler. Automatically tagging email by leveraging other users' folders. In *Proceedings of KDD*, San-Diego,CA, Aug 2011.

[15] B. Leiba. Update to internet message format to allow group syntax in the "from:" and "sender:" header fields, RFC 6854, March 2013.

[16] X. Li and W. B. Croft. Time-based language models. In *Proceedings of CIKM*, pages 469–475. ACM, 2003.

[17] C. Macdonald and I. Ounis. Combining fields in known-item email search. In *Proceedings of SIGIR*, Seattle, WA, Aug 2006.

[18] P. Ogilvie and J. Callan. Experiments with language models for known-item finding of e-mail messages. 2005.

[19] D. Olanoff. Gmail search autocomplete gets smarter predictions, using past searches to help you find things. Techcrunch, April 2013.

[20] E. P. Resnick. Internet message format, RFC 2822, April 2001.

[21] S. Radicati and J. Levenstein. Email market, 2013-2017, Nov 2013.

[22] S. Robertson, H. Zaragoza, and M. Taylor. Simple BM25 extension to multiple weighted fields. In *Proceedings of CIKM*, pages 42–49. ACM, 2004.

[23] I. Soboroff, A. P. de Vries, and N. Craswell. Overview of the trec 2006 enterprise track. In *TREC*, 2006.

[24] W. Weerkamp, K. Balog, and M. De Rijke. Using contextual information to improve search in email archives. In *Advances in Information Retrieval*, pages 400–411. Springer, 2009.

[25] S. Whittaker, T. Matthews, J. Cerruti, H. Badenes, and J. Tang. Am I wasting my time organizing email?: A study of email refinding. In *Proceedings of CHI*, Vancouver, BC, Canada, 2011.

[26] S. Whittaker and C. Sidner. Email overload: Exploring personal information management of email. In *Proceedings of CHI*, Vancouver, BC, Canada, 1996.

[27] S. Yahyaei and C. Monz. Applying maximum entropy to known-item email retrieval. In *Advances in Information Retrieval*, pages 406–413. Springer, 2008.