Promoting Relevant Results in Time-Ranked Mail Search

David Carmel, Liane Lewin-Eytan, Alex Libov, Yoelle Maarek, Ariel Raviv
Yahoo Research, Park MATAM, Haifa 31905, Israel
{dcarmel, liane, alibov}@yahoo-inc.com, yoelle@yahoo.com,arielr@yahoo-inc.com

ABSTRACT

Mail search has traditionally served time-ranked results, even if it has been shown that relevance ranking provides higher retrieval quality on average. Some Web mail services have recently started to provide relevance ranking options such as the relevance toggle in the search results page of Yahoo Mail, or the "top results" section in Inbox by Gmail. Yet, ranking results by relevance is not accepted by all, either in mail search, or in in other domains such as social media, where it has even triggered some public outcry. Given the sensitivity of the topic, we propose here to investigate a mixed approach of promoting the most relevant results, to which we refer as "heroes", on top of time-ranked results. We argue that this approach represents a good compromise to mail searchers, supporting on one hand the time sorted paradigm they are familiar with, while being almost as effective as full relevance ranking view that Web mail users seem to be reluctant to adopt. We describe three hero-selection algorithms we have devised and the associated experiments we have conducted in Yahoo mail. We measure retrieval success via two metrics: MRR (Mean Reciprocal Rank) and Success@k, and verify agreement between these metrics and users' direct feedback. We demonstrate that supplementing time-sorted results with hero results leads to a higher MRR than the traditional time-sorted view. We additionally show that MRR better reflects users' perception of quality than Success@k. Finally, we report on online results following the successful launch of one of our hero-selection algorithms for all Yahoo enterprise mail users and a few million Yahoo Web mail users.

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

Keywords: Mail Search, Rank-by-time, Heroes

1. INTRODUCTION

Mail, the most traditional medium for digital communication, sees its traffic continuously increasing, mostly due to the increase in volume of machine-generated messages [14]. As inboxes keep growing, and most users have been shown never to clean up their inboxes [9], search capabilities become even more critical. Unfortunately, mail search has not observed the same pace of progress as Web search and is still perceived as being somehow broken [13].

Some Web mail systems have recently started to offer options for sorting results by relevance, in the hope to address this issue. One example is the Date/Relevance toggle in the Search Results view of Yahoo Mail, and another one is the "top results" section[1] that displays a few relevant results on top of the usual chronological list in the "Inbox by Gmail" variant of Gmail. Yet, these examples are rather the exceptions than the rule, and most mail search systems still rank results by time.

One possible interpretation for this somehow surprising anachronism, is that, like in desktop search, users expect perfect recall as they are trying to re-find "stuff they've seen" [11]. To this effect, they apply different search strategies. In some cases, they formulate very clear multiple-word queries, and in many cases they "play it safe", entering only a contact name and then exhaustively browsing search results in order to make sure they are not missing the relevant messages, [4]. While on average, ranking by relevance is still superior on average as demonstrated in [3], it might frustrate the searchers who are used to their own search strategy, and even strengthen the perception that mail search is broken.

Users preferring chronological ranking is not a phenomenon unique to mail. In social media for instance, some users strongly objected relevance ranking in the Twitter timeline[2] and more recently in Facebook and Instagram[3].

We propose here to re-build the user's trust in mail search by offering a mixed approach, that offers a combination of relevance and time-sorted results for easier adoption. We study several algorithms for adding a few most relevant results, to which we refer as "heroes", on top of the traditional time sorted view, as illustrated in Figure 1. One key novelty of our approach consists of optimizing the quality of results in a fixed *display window* of size k, where the first h heroes are the top relevant results and $k-h$ remaining results are the top time results. Another benefit of this approach is that the $k-h$ top time results transparently merge into the

WWW 2017, April 3–7, 2017, Perth, Australia.
ACM 978-1-4503-4913-0/17/04.
http://dx.doi.org/10.1145/3038912.3052659

[1] http://gmailblog.blogspot.co.il/2016/01/inbox-by-gmail-find-answers-even-faster.html
[2] http://www.theguardian.com/technology/2015/dec/09/twitter-courts-dislike-by-reordering-tweets-on-relevance
[3] http://www.latimes.com/business/technology/la-fi-tn-instagram-feed-20160315-story.html

rest of the traditional time sorted results, allowing the users to fall back into their regular search paradigm.

Figure 1: Display window of size 6, where 2 heroes are presented on top of the time ranked list.

We discuss three distinct algorithms for identifying such heroes and evaluate the quality of the top-k results, where the h first results are served by the relevance-based ranking algorithm deployed today in Yahoo Mail search backend [3], and the remaining $k - h$ are the most recent results. These three algorithms mostly differ in terms of (1) allowing/forbidding duplication of results between time and relevance in the top k results, and (2) using either a fixed number H or a variable number $h \leq H$ of heroes. We also consider two measures of quality for evaluation, Mean Reciprocal Rank (MRR) and Success@k[4] in a series of offline and online experiments.

The main contributions of this work are two-fold: (1) we present the first public study, to the best of our knowledge, investigating the benefits of augmenting time-ranked results with relevant results and,(2) we introduce and evaluate various algorithms for promoting hero results on the top of time results, optimizing the top-k display window that contains both heroes and time results for various values of k.

The rest of this paper is organized as follows. Section 2 covers some related work. Section 3 describes a large-scale analysis performed over a large Yahoo mail query log. Section 4 introduces our three hero selection algorithms, which are then evaluated in Section 5. Section 6 details a manual evaluation conducted by professionally trained Yahoo evaluators. Finally, Section 7 presents our online evaluation following the launch of one of our hero selection algorithms in Yahoo mail. We conclude in Section 8 by discussing the counter-intuitive results we observed in terms of tolerance for duplicates in search results, as well as the alignment of results across all evaluation methods.

2. RELATED WORK

Incorporating time-based considerations into relevance-based ranking has been extensively studied, typically by adding time-based features into the relevance ranking model so as to favor recent documents (e.g., [16, 10, 12, 5]). In the context of mail search, Carmel et al. [3] followed this direction, considering not only freshness features but also mail-specific features such as user actions on a given message. While many features significantly contributed to the overall ranking, the time-based features were found to be the most significant in determining the message score.

[4]Commonly defined as the relative number of queries for which the clicked message appears in the top-k results.

Another direction for integrating document freshness into the relevance model predicts the time sensitivity of the query (e.g., [15, 8, 7]). For time-sensitive queries, such as those pertaining to breaking news, older result candidates, which become stale fast, should be penalized with lower relevance scores. In this line of works, queries are first classified according to their time sensitivity. Then, document freshness features are boosted for time-sensitive queries, and ignored or demoted for atemporal queries.

Another form of displaying search results that is related to our work allows the user to rank the results according to different criteria (recency, relevance, authority, etc.) [6]. This view allows users to sort results according to their needs. Several Web mail search systems (e.g., Yahoo mail) follow this model and allow users to re-rank the default time-based results by relevance. However, this form requires further user interaction with the search results, and therefore the default ranking criterion plays an important role as experience shows that most users stick with the default rank and do not bother to re-rank the results even when beneficial [3]. In contrast, our hero-based approach is designed for discoverability, saving the need for additional re-ranking interaction.

Time-based ranking is also popular in social media sites such as Twitter, in spite of that content exploration requires substantial effort of users who sequentially scan the streams of tweets. A great deal of work was focused on developing relevance-based ranking models for Twitter search (e.g., [17]), however, relevance ranking of the stream of tweets has not been welcomed by Twitter users and has still not been integrated into the product. Bernstein et al. [2] presented an interface which categorizes the tweets into topics and provides access to the tweet classes by means of tag clouds. Abel et al. [1] enriched Twitter rank-by-time by faceted search, inferring facets such as locations and persons of individual messages, thus allowing users to narrow the search results into a specific facet.

3. MAIL RANKING ANALYSIS

We use here a dataset of 100K queries randomly selected over a period of one month of Yahoo Web mail. We used the same type of offline analysis conducted by some of the authors of this work, as described in [3]. Namely we associated each query with the result message clicked by the searcher and obtained MRR scores[5] of 0.526 for simulated relevance ranking as compared to 0.37 for the time ranking approach, that was in production in Yahoo Mail at the time. We computed the relevance ranking by using the relevance-based scoring function presented in [3], and verified again the superiority of relevance over time ranking, on a much larger dataset than the $10K$ log used in [3].

Additionally, we measured the fraction of queries for which the clicked message appeared in the top-k results (Success@k) or not (denoted by Failure@k) for $k \in \{6, 10\}$ in each ranking approach, as detailed in Table 1. The columns correspond to the performance of time ranking, while the rows correspond to relevance ranking. Each inner cell in the tablerepresents the fraction of the queries where both time and relevance

[5]Note that due to the offline nature of the analysis, we did not include unsuccessful queries and thus obtained higher MRR than what we will observe in our online analyses in Section 7.

rankings perform according to their respective column and row. For example, when considering the top six results, we see that in 9.7% of the cases time ranking fails while relevance succeeds. The "Overall" columns/rows sum up the cases where either time or relevance always succeeds or fails.

Time / Relevance	Success@6	Failure@6	Time-Overall
Success@6	66.2%	9.7%	75.9%
Failure@6	5%	19.1%	24.1%
Rel-Overall	71.2%	28.8%	100%
Time / Relevance	**Success@10**	**Failure@10**	**Time-Overall**
Success@10	76.8%	**7.5**%	84.3%
Failure@10	**3.7**%	12%	15.7%
Rel-Overall	80.5%	19.5	100%

Table 1: Fractions of queries according to the occurrence of the clicked message in the top-k results, for $k \in \{6, 10\}$, ranked by time and by relevance.

We see that, when considering the top ten results, relevance ranking succeeds in 7.5% (in bold) of the cases in which time ranking fails, and conversely time ranking succeeds in 3.7% (in bold) of the cases, for which relevance fails. In other words, there exist results that one ranking paradigm returns while the other misses them. Our interpretation here is that users will use different strategies in mail search depending on their intent and trust in the system. As discussed in [4], about 55% of the queries are contact queries, where the user's search strategy is to simply enter a contact name without additional information, and then scan all messages received or sent by the contact in order to improve recall. In such cases the time ranking view will help him browse systematically. This encourages us to investigate a mixed approach that supports both search ranking paradigms, as discussed in the next section.

4. HERO SELECTION ALGORITHMS

Given the variety of existing devices, a key parameter is the number of search results, noted by k, that are shown "above the fold", without scrolling. The top-k results presented on the *display window* consist of both hero results, noted by $HList$, where $|HList| = h$, and the head of the time-sorted results, noted by $TList$, ($|TList| = k - h$), as illustrated in Figure 1. Our task here is to identify $h \leq H$ hero results; we impose that $H < k$.

The size of the display window can significantly vary according to the device, from smart-phones to large monitors. Note that $TList$ is in fact the head of the original *Time* list. In our implementation, we highlight the heroes results, but transparently merge the elements of $TList$ with the remaining elements of the time results, allowing the user to gracefully fall back to the traditional chronological view if not satisfied with the top k results.

We present three different algorithms to compute $HList$ and accordingly derive $TList$. These algorithms differ by two main attributes: (1) the size of $HList$, that is, the number of heroes served, and (2) allowing or forbidding duplication between heroes and time results, when a highly relevant result is also a most recent one.

1. *Heroes-Dup* selects exactly H heroes, according to their relevance scores, with $TList$ being formed by the top $(k - H)$ results of the *Time* list which may lead to duplication in the display window. Note that this algorithm provides a user experience that will look similar to the one offered by *Inbox by Gmail*, yet, as we do not know whether the latter slots top relevant results on top of time results or rather uses a more global optimization approach, we cannot discuss additional differences or similarities.

2. *Heroes-Fixed* behaves like *Heroes-Dup* yet forbids these heroes from appearing in $TList$, thus eliminating duplicates in the top k results. We refer to them as "hidden heroes", since in a traditional time-sorted view, they might be hidden well below the fold.

3. *Heroes-Iter* picks a non-fixed number of $h \leq H$ heroes, depending on the query, and avoids duplication in the display window. This algorithm works iteratively, increasing at each step the average relevance score of the messages in the display window, as described in detail below.

Our goal is to maximize user's satisfaction, which is evaluated by the two following measures: Success@k, where a user is considered satisfied if the last message she clicks on belongs to the top k results, and MRR, where user satisfaction is considered higher if the rank of the clicked message is lower, with the minimal rank being 1 for the top result. We compare the hero algorithms, using these two measures, to the traditional *Time* algorithm.

Algorithm 1 *Heroes-Iter*

Input
k - window size
$H < k$ - maximum number of hidden heroes to select
$R_t = \{t_1, \ldots, t_n\}$ – *Time* message list
Output
$< TList, HList >$ - two message lists s.t. $|TList| + |HList| = k$
Init
$TList = \{t_i \in R_t | 1 \leq i \leq k\}$
$HList = \emptyset$, $h = 0$
while $h < H$ **do**
  $m_h = \text{argmax}_{m \in R_t - (TList \cup HList)} Rel(m)$
  if $Rel(m_h) > Rel(TList[k-h])$ **then**
    ▷ Hero found
    $HList \leftarrow HList \cup \{m_h\}$
    $TList \leftarrow TList - \{TList[k-h]\}$
    $h \leftarrow h + 1$
  else
    break
return $HList$

Algorithm 1 presents the *Heroes-Iter* algorithm for identifying the complementary (short) list of hero messages to be presented in addition to the *Time* list, while adhering to a total of k results for both lists. The maximum number of heroes to be selected is limited by $H < k$. We begin with the initial list of the top-k results from the *Time* list. At each step our algorithm identifies the message with the maximal relevance score, m_h, among the results that are not currently presented. If the relevance score of m_h is higher than the relevance score of m_t, the last result in the current time-based ranked list, then it is is added to the hero list, while m_t is taken out from the time list. The stopping condition is met if we reach H heroes, or if m_h is lower or equal to m_t, i.e., the candidate hero, if added, would replace a higher scored message in the top k results and thus might

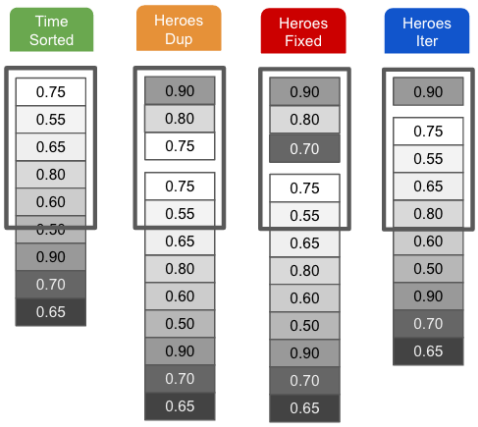


Figure 2: Example outputs of *Time*, *Heroes-Iter*, *Heroes-Fixed* and *Heroes-Dup* for a given query.The numbers in the cells represent the relevance scores of each message, while their brightness represent their recency: the fresher a message, the brighter the cell.

hurt precision. We denote the relevance score of result m by $Rel(m)$.

Note that the *Heroes-Iter* algorithm preserves three invariants:

1. The sum of the sizes of the two lists remains k, which corresponds to the top k results to be shown to the searcher. The *Heroes-Iter* algorithm keeps this invariant as it is initiated with k items and any addition of a hero message to $HList$ is followed by the removal of the last message from $TList$.

2. There are no duplicate messages across the two lists, since newly added hero messages are always selected from non presented results. Note that a message pushed out from $TList$ can return as a hero in a later stage of the algorithm.

3. The average relevance score of messages in the union of the two lists is monotonically increasing, since, at any stage we replace a lower-scored message with a higher-scored message.

Note that *Heroes-Iter* is conservative in its selection process, as it stops when the selection criterion fails. Based on our analysis, only 1-2 heroes will be typically selected per query. Moreover, it does not guarantee that the average relevance score of the messages in both lists is maximal. The *Heroes-Fixed* algorithm also preserves all invariants mentioned above. Additionally, in contrast to *Heroes-Iter*, it maximizes the average relevance score of the messages in the presented window, under the constraint of presenting $k - H$ messages ranked by time, hence it is expected to outperform *Heroes-Iter* according to the Success@k evaluation criterion. However, it *always* selects H heroes per query, an undesirable solution for queries that are already satisfied with the *Time* list, and when selected heroes have a low relevance score.

Figure 2 depicts an example corresponding to a query with the results returned by each algorithm for $k = 5$ and $H = 3$. The left side bar corresponds to time-based ranking[6], followed by *Heroes-Dup* and *Heroes-Fixed* and the right bar corresponds to *Heroes-Iter*. Heroes appear on top of the time sorted list. As can be seen from the figure, *Heroes-Dup* picks $H = 3$ most relevant messages as heroes, leading to a duplication of the message with score 0.75 in the window. The *Heroes-Fixed* algorithm picks $H = 3$ most relevant messages out of all the messages excluding the $k - H = 2$ most recent ones (i.e. those with scores 0.75 and 0.55), since they already appear in the window. Thus, it eliminates duplicates in the display window and chooses the older but relevant message with a score of 0.70 as the third hero result. The *Heroes-Iter* algorithm picks the most relevant message not currently in the window, scored 0.90, as the first hero, since it is more relevant than the last message scored 0.60, in the current window. The second most relevant message outside the updated window, scored 0.70, is tested against the current last message, which scores 0.80. As the relevance of the second potential hero is lower, it is not chosen and the *Heroes-Iter* algorithm stops with only one hero result selected.

[6]Note that two different messages can get the same relevance score.

5. OFFLINE EXPERIMENTS

We now present the results of an experiment on the same dataset mentioned in Section 3. Each query is associated with up to 100 retrieved messages, ranked by time, and the message that was clicked by the searcher[7]. In all experiments and for all the Hero algorithms, we use the relevance-based scoring function presented in [3].

5.1 Overall Quality Evaluation

We evaluate the quality of the three hero algorithms, and compare them with the time-based and relevance-based sorted results according to the MRR and Success@k measures. Figure 3 shows the Success@k and MRR results for the heroes algorithms and for *Time* and *Rel* (with $k = 6$, and with $H \in [0..k]$).

The leftmost point of the curves corresponds to $H = 0$ (no heroes), where all algorithms return the top-k results of the *Time* list. The rightmost point corresponds to $H = k$, where both *Heroes-Dup* and *Heroes-Fixed* return k heroes, which are the top k results of *Rel*, with no *Time* results, while *Heroes-Iter* returns $h \leq k$ heroes with $k-h$ time-based ranked messages (h is determined per query). Typically, the higher the number of heroes, the higher the average relevance score in the display window, and consequently, the higher the Success@k and MRR scores. This is consistent with the finding that relevance ranking achieves better quality than time-based ranking [3].

The *Heroes-Dup* algorithm achieves the lowest Success@k value, due to the duplication often occurring in the window. Obviously, if the top relevant result is also recent, the same result may appear twice in the window, both as a hero and within the time results. In this case, the window is thus not fully exploited and the success rate decreases. The potential number of duplicates is maximal when the two lists have a similar size. When H gets closer to the window size, k, the success rate of *Heroes-Dup* increases as well, as it converges to the relevance-ranked list.

In terms of MRR, *Heroes-Dup* is superior to *Heroes-Iter* and *Heroes-Fixed*, with an MRR value that increases as the

[7]If more than one message was clicked, we consider the last one.

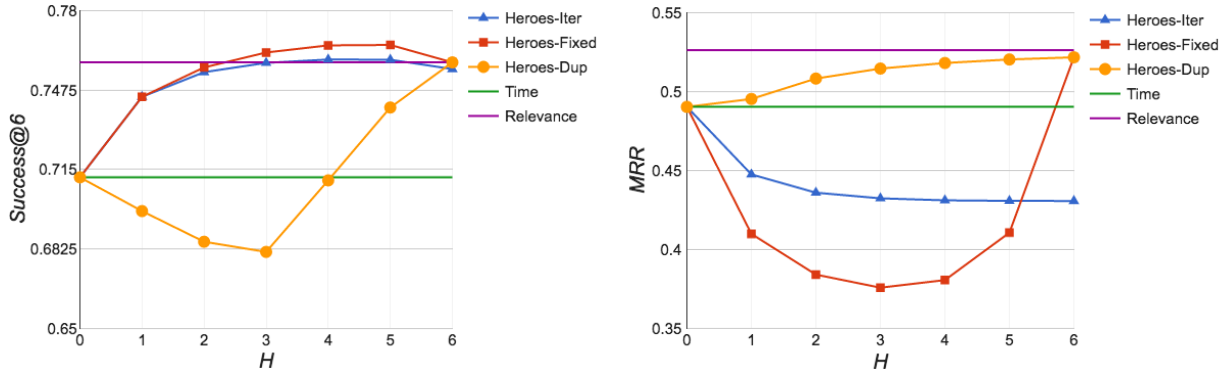


Figure 3: MRR and Success@k of the hero algorithms and of the time and relevance ranking. Measures of the hero algorithms are presented as a function of number of heroes (H). The maximal number of heroes is set to be k for $k = 6$.

number of heroes increases. since it converges to the relevance ranking view. The MRR of *Heroes-Iter* and *Heroes-Fixed* decreases as the number of heroes increases (for *Heroes-Fixed*, up to half the size of the window). This happens as heroes are put on top and the most recent results (*Time*) are pushed further down the list. Recall that when $H < k$, the top $k - H$ time results cannot be selected as heroes as *Heroes-Iter* and *Heroes-Fixed* do not allow duplicates. For *Heroes-Fixed*, the MRR increases for high values of H, as when H gets closer to k, it converges to relevance ranking. In addition, the Success@k shows better results for the hero algorithms than for *Rel* at high H values according to Success@k.

Comparing the results for the two evaluation measures, it is clear that *Heroes-Dup* improves MRR, while the two other algorithms, *Heroes-Iter* and *Heroes-Fixed* improve Success@k. The *Heroes-Dup* algorithm simply puts the most relevant results on top, and therefore follows *Rel* which is optimized to maximize MRR. On the other hand, the *Heroes-Iter* and *Heroes-Fixed* algorithms, by avoiding duplicates, increase the diversity of the messages, and the average relevance score in the window, thus optimizing Success@k.

MRR is hurt by *Heroes-Iter* and *Heroes-Fixed* when heroes are added to the window, since for many queries the most recent results, which are pushed down, and are not duplicated in the hero list, are very likely to contain the clicked result (see Table 1). In addition, a hero can often be selected from a very low rank according to *Time*, hence it was neither exposed nor clicked by the searcher.

Interestingly, *Heroes-Iter* and *Heroes-Fixed* lead to their best Success@k value when H approaches half the size of the display window. In addition, when H is slightly above half of the display window, the *Heroes-Dup* algorithm achieves an almost optimal MRR. When comparing *Heroes-Fixed* with *Heroes-Iter* for higher H values, *Heroes-Iter*'s Success@k rate is slightly lower than *Heroes-Fixed*, the optimal algorithm for this measure, possibly due to the lower number of heroes it selects. However, for almost all values of H, *Heroes-Iter* is superior to *Heroes-Fixed* with respect to the MRR measure (apart from $H = 6$, where *Heroes-Fixed* fills the window with the top k relevant messages).

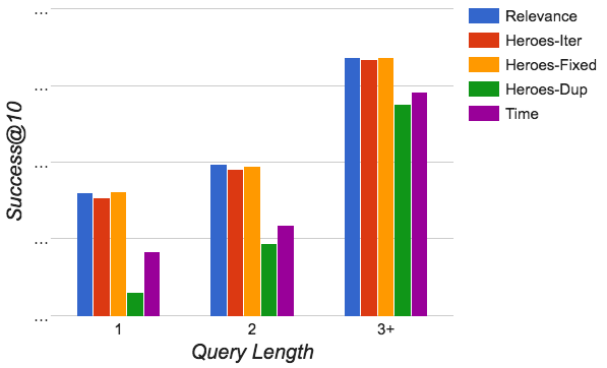
It is also interesting to analyze the relationship between Table 1, and Figure 3. *Time*, with no heroes, is of high quality for about 71% of the queries, for $k = 6$ (See Figure 3, when $H = 0$). The results definitely show that when replacing some of the results in the window with heroes, we increase this portion of success by bringing more clicked messages to the window. However, by doing so we also hurt some of the queries for which time-based ranking is superior to relevance-based ranking. Thus, while overall *Heroes-Fixed* and *Heroes-Iter* achieve a nice increase in the average Success@k measure, there is still room for improvement for specific "temporal sensitive" queries [10], for which time-based ranking is preferred.

5.2 Query-Dependent Quality Evaluation

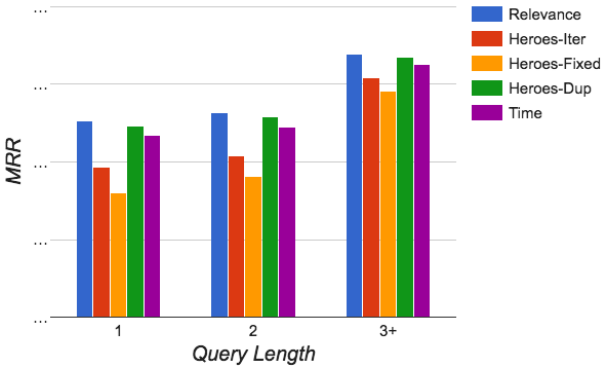
We evaluate the quality of the Heroes algorithms for different subsets of queries, and derive respective insights as to the added values of our algorithms with respect to the nature of the search queries. For all experiments presented in this section, we use as maximal number of heroes $H = 3$ and $k = 10$.

Query Length. Figure 4 presents the quality of our algorithms with respect to query length. The distribution of the query length is presented in Table 2, where it can be seen that more than 90% of the queries contain less than three terms, and almost 70% of them contain one term only. Figures 4(a) and 4(b) give the Success@k and MRR scores of the different algorithms as a function of the number of query terms. For all algorithms, it can be seen that the longer the query, the better the quality, which is good property that validates the underlying relevance measure, REX2, independently on how results are sorted. Paradoxically, users still don't trust the search mechanism and favor short queries. This led us to investigate mail query suggestion mechanisms, out of the scope of this paper, to encourage users to issues longer, more specific, queries.

We consider the difference between the measures of the different heroes algorithms and time ranking. With respect to Success@k, it can be seen that the shorter the query (one or two terms), the higher the added value of the heroes algorithms as compared to the time-sorted view. For queries of length one or two, *Heroes-Iter* and *Heroes-Fixed* achieve about 4.5% increase, while they achieve only 2.3% increase for queries of length three and more. With respect to MRR, *Heroes-Dup* achieves an increase of 5.3% for single-term queries, while achieving an increase of 2.7% for queries of length three and more. Thus, the shorter (and less precise) the query, the

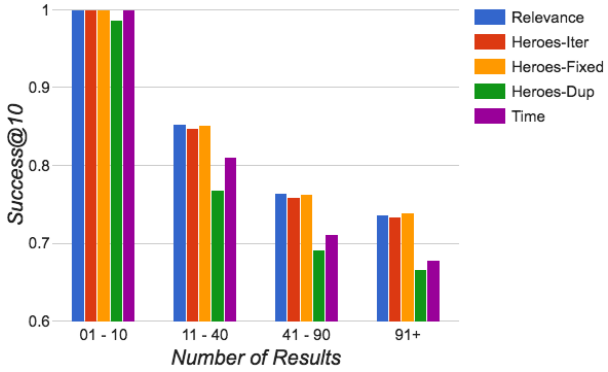


(a) *Success*@10

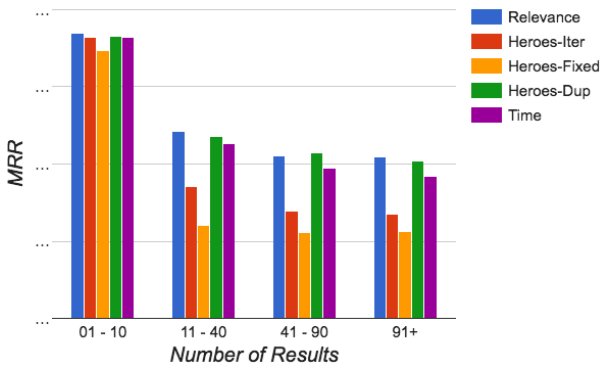


(b) *MRR*

Figure 4: Success@10 and MRR as a function of the query length, for $k = 10$ and $H = 3$.



(a) *Success*@10



(b) *MRR*

Figure 5: Success@10 and MRR as a function of the number of results, for $k = 10$ and $H = 3$.

higher the added value of the heroes algorithms as compared to the time-sorted view.

Query Length	Queries Percentage
1	69.5%
2	21.1%
3+	9.4%

Table 2: Fractions of queries wrt length

Results Number	Queries Percentage
1-10	26.9%
11-40	24.8%
41-90	21.8%
91+	26.5%

Table 3: Fractions of queries wrt number of results

Number of Results. Figures 5(a) and 5(b) present the Success@k and MRR measures of the algorithms as a function of the number of results matching the query. The distribution of the queries with respect to their number of results is presented in Table 3. For all algorithms, it can be seen that the higher the number of results, the lower the quality, as it becomes harder to identify the most relevant results.

The higher the number of results, the higher the added value of the heroes algorithms with respect to time-based ranking. With respect to MRR, *Heroes-Dup* is equal to *Time* for up to 10 results, while it achieves an increase of more than 10% for more than 40 results. With respect to Success@k, *Heroes-Iter* and *Heroes-Fixed* are equal to *Time* for up to 10 results, while they achieve an increase of more than 9% for more than 40 results. As before, the advantage of the heroes increases as the quality of *Time* decreases.

6. MANUAL EVALUATION

We conducted a manual evaluation, with the help of Yahoo professional evaluators who were given directives on searching their own mailboxes in the current and new settings. Our evaluation included 16 trained professionals, who were all asked to issue 20 queries on their personal mailboxes. Out of the 20 queries, 16 queries (80%) had to match a given pattern, which specified the number of terms as well as their type (e.g. <person name> <content word>). The patterns were defined to cover the most prominent query scenarios as found in our query log. The evaluators were directed to formulate their own queries, in full freedom, as long the latter matched one of the required patterns. Examples of query patterns are given in Table 4, while examples of real evaluators' queries instantiating these patterns with associated intents are shown in the three left columns of Figure 6. Additionally, 4 out of the 20 queries (20%) were entirely up to the evaluators to devise. Moreover, 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.

Each query was run on four systems: one using the existing chronological ranking deployed in the Web mail system we experiment with, and three novel systems, each using a

			TIME		HEROES 1		HEROES 2		HEROES 3	
Pattern	Query	Intent Description	Most relevant	Relevant	Most relevant	Relevant	Most relevant	Relevant	Most relevant	Relevant
<company name> <content word>	Uber receipt	Receipt from Uber from April	2	1	5	3	3	2	2	1
<content word>	resume	Leah's resume	2	1	1	5	5	2	2	1
<content word> <content word>	Chess tournament	Reminder about MVWSD chess tournament	3	1,2,4	1	4,5,6	5	2,3,4	3	1,2,4

Figure 6: Editorial evaluation - examples of evaluators' queries and ranking results.

Query Pattern Examples
<company name>
<person name> <content word>
<company name> <content word>
from:<person name> <content word>
<content word>
<content word> <content word>
<attachment name>
<company name> <content word> <content word>

Table 4: Manual evaluation - examples of query patterns.

different version of the hero algorithm, namely *Heroes-Iter*, *Heroes-Fixed*, and *Heroes-Dup*. For this evaluation, we set the maximum number of heroes, H, to 3, and the window size, k, to 6. The values correspond to a mobile setting where the window size is relatively small, as illustrated in Figure 1.

For each system, the evaluators had to identify the rank of the most relevant result, i.e. the message they wanted to re-find when formulating their query. The rank of a message is determined with respect to its position in the unified list of both relevance-based ordered hero results on top, and the time-based ordered recent results below. In addition they were asked to judge other relevant results, if any. Each message was labeled according to three relevance levels: most relevant (assigned to at most one result), relevant, and non-relevant. Examples of result assessments made by the evaluators for their own queries are presented in the right-side columns of Figure 6.

Overall, 320 personal mail queries were collected for this experiment, where each query is associated with top-6 messages, judged by the evaluator who submitted the query, for each algorithm we experimented with. The ranking algorithms were evaluated using both MRR and Success@k measures, as before. Furthermore, the identification of the relevance level of the judged results allowed us to compute their $NDCG@k$ (Normalized Discounted Cumulative Gain) values. The latter was computed over the top-6 results, using three relevance scores $0, 1, 7$ for non-relevant, relevant and most relevant results, respectively. Table 5 summarizes the results of the manual evaluation.

The table shows that the algorithms that avoid duplications, namely *Heroes-Fixed* and *Heroes-Iter*, outperform *Time* and *Heroes-Dup* algorithms with respect to the Success@6 measure, which complies with their goal of optimizing recall within the chosen window (with a slight benefit to *Heroes-Fixed* that outperforms *Time* by 9.9%). However, considering MRR and $NDCG@6$, there is a clear advantage to *Heroes-Dup*, followed by the traditional Time-based ranking model, where *Heroes-Dup* outperforms *Time* by 6.2% and 5.7% for MRR and NDCG respectively. These results are aligned with our offline evaluation results, where *Heroes-Fixed* and *Heroes-Iter* (hero algorithms with no duplications) perform clearly better according to the Success@k measure, and *Heroes-Dup* (hero algorithm with duplication) performs better according to the MRR measure.

In addition to providing relevance judgments, the evaluators were asked to rate several qualitative statements describing their overall experience with the hero model, comparing it to the traditional time based model. They were offered a range of options, ranging from 1 (lowest satisfaction) to 7 (highest satisfaction). Table 6 presents the evaluators' feedback as well as the average ratings for each statement. As can be seen, most answers were positive with respect to the hero model. For example, the statement "*With the hero model, I felt that I could easily find the message I was looking for*" got an average ratings of 5.2 (out of 7).

Lastly, we asked evaluators which of the four systems they liked the best. Out of the 16 participants, 11 (73%) indicated that *Heroes-Dup* provided the best overall experience. Their qualitative feedback thus demonstrates a clear preference of *Heroes-Dup*. As demonstrated in both the offline and manual evaluation, *Heroes-Dup* outperforms the other hero algorithms in terms of MRR. There seems to be a clear agreement between expressed users' satisfaction and MRR, indicating that users prefer to see their most relevant results at top positions, even if duplicated, rather than optimizing the overall success rate.

7. ONLINE EVALUATION

Based on the promising results achieved in our offline and manual evaluations, *Heroes-Dup* was launched in both Yahoo enterprise mail and part of Yahoo Web mail. Figure 7 shows an example of the integration of *Heroes-Dup* in Yahoo Mail, where the relevant results are older than the top of the sorted-by-time list and do not appear as top results in the traditional time-only view.

We compared *Heroes-Dup* with the current time-based ranking in both enterprise and Web mail settings, each having their own characteristics, [3]. Our online evaluation is based on a sample of $50K$ queries originating from general Web mail users and $180K$ queries issued by Yahoo employees in our enterprise environment. In both cases, half of the queries were served the traditional time sorted results, while the other half were served a hero-based treatment. Note that given that this experiment was conducted on live traffic, we had to chose a single new treatment and elected to pick the one that did best in our manual evaluation, namely *Heroes-Dup*.

Table 7 summarizes the results of the online evaluation, considering both MRR and Success@6 measures. As can be seen, both measures achieve a significant increase, where the

	Time	*Heroes-Fixed*	*Heroes-Iter*	*Heroes-Dup*
Success@6	0.82	**0.90 (+9.9%)***	0.89 (+9.2%)*	0.86 (+5.4%)*
MRR	0.65	0.59 (-8.9%)	0.61 (-6.1%)	**0.69 (+6.2%)***
NDCG@6	0.73	0.71 (-2.6%)	0.73 (-0.9%)	**0.77 (+5.7%)***

Table 5: Manual evaluation: MRR, Success@6 and NDCG@6 values of *Time* and hero algorithms. All gains marked with (*) are statistically significant (two-tailed paired t-test, $p < 0.05$).

User Experience Feedback	**Avg**
With the hero model, I felt that the results were related to my query	**4.87**
With the hero model, I felt that I could easily find the message I was looking for	**5.20**
I prefer the hero model way of looking at results	**4.80**

Table 6: User experience with the hero model compared to the current time-based model

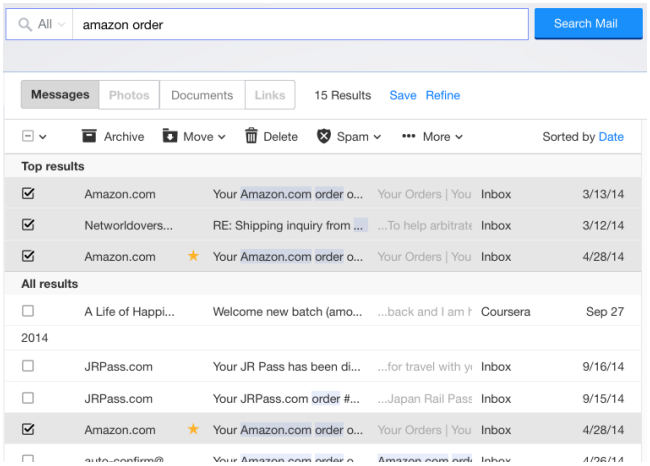


Figure 7: Integration of hero results (with H=3) in Yahoo Mail. The heroes are presented in the "Top results" section followed by "All results" section, which corresponds to the traditional time-ordered view of the results.

highest increase was achieved for the MRR measure, in alignment with the offline and manual evaluations for *Heroes-Dup*. More specifically, we observed an increase of 18.1% in MRR, in enterprise mail and of 12% in Web mail. One possible interpretation for this difference could be that enterprise users, especially in a Web company, are more trained at expressing discriminating queries.

8. CONCLUSIONS

In this paper, we argued that exclusively showing sorted-by-time results in mail search is somehow an anachronism. We described three algorithms that promote relevant "hero" results on top of time-ranked results, and demonstrated that they outperform traditional time ranking. Using two measures of success, MRR and Success@k, we discovered that our simpler algorithm with duplicates, *Heroes-Dup*, achieves a higher MRR and a lower Success@k score, while the two other algorithms achieve a lower MRR but higher Success@k scores. In addition, while intuitively we would have anticipated that duplicate results would annoy mail search users, a qualitative evaluation conducted by professional evaluators clearly favored *Heroes-Dup*. This leads us to believe that in mail search, MRR is the better measure of success, and users can tolerate duplication as long as they find their result in the top ranks. Based on these findings, our *Heroes-Dup* algorithm has been fully launched in Yahoo enterprise mail and part of Yahoo Web mail. While top results might also appear now in other Web mail services, such as Inbox by Gmail, this work is, to the best of our knowledge, the first one that details associated algorithms, online experiments, and demonstrate the value of hero results, with a significant increase of 18.12% and 12% in MRR, for enterprise mail and Web mail respectively.

Corporate mail		
	Time	*Heroes-Dup*
MRR	0.239	**0.282 (+18.1%)**
Success@6	0.591	**0.638 (+8.4%)**

Web mail		
	Time	*Heroes-Dup*
MRR	0.243	**0.272 (+12%)**
Success@6	0.602	**0.624 (+3.8%)**

Table 7: MRR and Success@6 of *Time* and *Heroes-Dup* algorithms measured over Yahoo Corporate and Web mail.

Given the promising results we obtained, we hope that more and more mail search engines will invest efforts in improving results relevance so as to correct the user's perception of mail search being broken. We strongly believe that once mail search can be trusted, users will issue longer queries, ranking will keep improving and hopefully mail search will start following the same path of success as Web search.

9. REFERENCES

[1] F. Abel, I. Celik, G.-J. Houben, and P. Siehndel. Leveraging the semantics of tweets for adaptive faceted search on twitter. In *Proceedings of ISWC*, pages 1–17. Springer-Verlag, 2011.

[2] M. S. Bernstein, B. Suh, L. Hong, J. Chen, S. Kairam, and E. H. Chi. Eddi: Interactive topic-based browsing of social status streams. In *Proceedings of UIST*, pages 303–312. ACM, 2010.

[3] D. Carmel, G. Halawi, L. Lewin-Eytan, Y. Maarek, and A. Raviv. Rank by time or by relevance? revisiting email search. In *Proceedings of CIKM*, pages 283–292, 2015.

[4] D. Carmel, L. Lewin-Eytan, A. Libov, Y. Maarek, and A. Raviv. The demographics of mail search and their application to query suggestion. In *Proceedings of WWW*. ACM, April 2017.

[5] S. Cheng, A. Arvanitis, and V. Hristidis. How fresh do you want your search results? In *Proceedings of CIKM*, pages 1271–1280. ACM, 2013.

[6] E. Cutrell, S. T. Dumais, and J. Teevan. Searching to eliminate personal information management. *Commun. ACM*, 49(1):58–64, Jan. 2006.

[7] N. Dai, M. Shokouhi, and B. D. Davison. Learning to rank for freshness and relevance. In *Proceedings of SIGIR*, pages 95–104. ACM, 2011.

[8] W. Dakka, L. Gravano, and P. G. Ipeirotis. Answering general time sensitive queries. In *Proceedings of CIKM*, pages 1437–1438. ACM, 2008.

[9] D. Di Castro, Z. Karnin, L. Lewin-Eytan, and Y. Maarek. You've got mail, and here is what you could do with it!: Analyzing and predicting actions on email messages. In *Proceedings of WSDM*, pages 307–316. ACM, 2016.

[10] A. Dong, Y. Chang, Z. Zheng, G. Mishne, J. Bai, R. Zhang, K. Buchner, C. Liao, and F. Diaz. Towards recency ranking in web search. In *Proceedings of WSDM*, pages 11–20. ACM, 2010.

[11] 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. Proceedings of SIGIR, pages 72–79, 2003.

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

[13] D. Elsweiler, M. Harvey, and M. Hacker. Understanding re-finding behavior in naturalistic email interaction logs. In *Proceedings of SIGIR*, pages 35–44. ACM, 2011.

[14] 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*, pages 869–878, 2014.

[15] R. Jones and F. Diaz. Temporal profiles of queries. *ACM Trans. Inf. Syst.*, 25(3), July 2007.

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

[17] K. Tao, F. Abel, C. Hauff, and G.-J. Houben. Twinder: A search engine for twitter streams. In *Proceedings of ICWE*, pages 153–168. Springer-Verlag, 2012.