Calendar-Aware Proactive Email Recommendation

Qian Zhao*
University of Minnesota
Minneapolis, MN, USA
zhaox331@umn.edu

Paul N. Bennett, Adam Fourney, Anne Loomis Thompson, Shane Williams, Adam D. Troy, Susan T. Dumais
Microsoft, Redmond, WA, USA
{pauben,adamfo,annelo,shanewil,adtroy,sdumais}@microsoft.com

ABSTRACT

In this paper, we study how to leverage calendar information to help with email re-finding using a zero-query prototype, Calendar-Aware Proactive Email Recommender System (CAPERS). CAPERS proactively selects and displays potentially useful emails to users based on their upcoming calendar events with a particular focus on meeting preparation. We approach this problem domain through a survey, a task-based experiment, and a field experiment comparing multiple email recommenders in a large technology company. We first show that a large proportion of email access is related to meetings and then study the effects of four email recommenders on user perception and engagement taking into account four categories of factors: the amount of email content, email recency, calendar-email content match, and calendar-email people match. We demonstrate that these factors all positively predict the usefulness of emails to meeting preparation and that calendar-email content match is the most important. We study the effects of different machine learning models for predicting usefulness and find that an online-learned linear model *doubles* user engagement compared with the baselines, which suggests the benefit of continuous online learning.

CCS CONCEPTS

• **Information systems → Email**; **Information retrieval**; **Recommender systems**;

KEYWORDS

email recommendation; email re-finding; email search

ACM Reference Format:
Qian Zhao and Paul N. Bennett, Adam Fourney, Anne Loomis Thompson, Shane Williams, Adam D. Troy, Susan T. Dumais. 2018. Calendar-Aware Proactive Email Recommendation. In *Proceedings of The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR '18).* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3209978.3210001

*Work performed while at Microsoft Research.

SIGIR '18, July 8–12, 2018, Ann Arbor, MI, USA

ACM ISBN 978-1-4503-5657-2/18/07…$15.00
https://doi.org/10.1145/3209978.3210001

1 INTRODUCTION

Email has been an important web-based communication medium for more than 25 years. While email was first designed for asynchronous communication people have "overloaded" the use of email with other functions such as task management and personal archiving [22, 35]. As online services and the web grow, email not only continues to serve all of these purposes [25] but an ever-increasing number, *e.g.* as a receipt file cabinet for e-commerce purchases, as a standard part of identity/authentication flow, and as calendar management. With regard to the last, because meeting arrangement and time negotiation often happen through email, nearly every modern email service – both web email and client applications – offer fully featured calendar management. Despite this integration, the majority of feature development has focused on moving information from email into a user's calendar but little work has focused on the implications of calendar information for improving other functionality of an email service.

In this paper, we focus on one such problem and ask whether the contextual information offered in calendar items can be used for more than simply reminding people of upcoming appointments. In particular, calendar items are a strong indicator of a person's future intent and identify a time for the appointment (informing a likely horizon of need) as well as often including the people involved, a subject, and a description. This forms a rich contextual basis to explore proactively finding and recommending emails the user will likely need for an upcoming calendar appointment. We study a setting particularly focused on meetings and associated information finding for meeting preparation in an enterprise email setting.

Proactive email recommendation can both improve email search efficiency and reduce the difficulty of query specification by enabling "search by meeting". The ideal experience is that the system does not wait for users to issue the query, but forms and issues the query for the user based on what we know about the user's current and future calendar appointments, (*i.e.*, zero-query search experience). We envision this could be activated by simply clicking on an appointment or activated by default (predicting both calendar item in focus and associated information). In this paper we study the setting where a user chooses the focus calendar item.

To understand the interaction of key factors, we construct an artifact named **CAPERS**: a Calendar-Aware Proactive Email Recommender System. CAPERS specifically focuses on meetings based on the observation that in work environments, people heavily rely on both emails and calendars to organize their daily activities and necessary information – an intuition that we first investigate using a survey. When the system has access to users' calendars, it gains a rich context including the time, duration and the content of a person's upcoming information needs. CAPERS targets at learning

the *usefulness* of emails with respect to the tasks of preparing for or having meetings. Usefulness is a stricter criteria beyond content relevance (and also preference), because emails can be highly topically relevant to a meeting but have no utility for preparation for the meeting itself (*e.g.*, sent time and people involved need to considered as well). CAPERS bootstraps the predictive models through pilot studies and further updates the models through online learning algorithms. In this paper, we aim to answer the following five research questions:

- *RQ1: How much email access is related to meetings?*
- *RQ2: What factors affect whether people prepare for a meeting?*
- *RQ3: What factors affect the usefulness of emails to a meeting and how?*
- *RQ4: How do machine-learning-based recommendation models and their online learning algorithms affect user experience in this system-level cold-start scenario?*
- *RQ5: What is the role of email recency in email recommendation (given prior work has shown that recency is an specially important factor for email search [10])?*

To address the above research questions, we employed mixed methods including a survey for RQ1-2 and a task-based and field experiment using the CAPERS prototype for RQ3-5. In the following, we first show through a survey that a large proportion of email access is related to meetings. Then, we examine how group size and the organizational relationship between meeting participants impacts the likelihood of preparing for a meeting. Next, we focus on learning recommendation models that leverage four categories of factors, including: amount of email content (e.g. length, number of attachments, etc.), email recency, calendar-email content match and calendar-email people match. Our aim is to both understand the modeling choices in this domain and understand the impact of each of these factors on predicting email usefulness for meeting preparation. Finally, through both a short task-based explicit judgment experiment and a longitudinal (one-month) field experiment with a small set of users, we study how people perceive the usefulness of the results returned by the different types of models and the impact on engagement with the recommendation models.

2 RELATED WORK

This paper is related to work in contextual information retrieval, personal search, context-aware and personal recommendation, email information extraction and classification, as well as email search and recommendation. We briefly review each of these areas here.

Contextual information retrieval. Information retrieval systems have been successfully utilizing user contextual information to further improve search results. *e.g.*, Bennett *et al.* [8] show how location information can be incorporated into Web search ranking. Their results demonstrate that a substantial fraction of Web search queries can be significantly improved by incorporating location-based features. As they later pointed out [7], there is a growing focus on how knowledge of user interests, intentions, and context can improve aspects of search and recommendation.

Personal search. Dumais *et al.* pioneered personal search in [12] through the "Stuff I've Seen" (SIS) system in 2003. The system provides a unified index of information that a person has seen, such as emails, web pages, documents, and appointments. The system then uses the rich cues of time and people to assist future re-finding. They demonstrated that SIS helps users find information more easily. Bendersky *et al.* [6] studied attribute parameterization algorithms to utilize user interaction data for further improving search over personal, rather than public, content. They show that attribute parameterization, which projects user queries and documents into a multi-dimensional space of fine-grained and semantically coherent attributes, enables effective usage of cross-user interactions for improving personal search quality. Zamani *et al.* [36] proposed context-aware ranking models to utilize situational context information of users, *i.e.*, the contextual features of the current search request that are independent from query content or user history, *e.g.*, the search time and location. Their results demonstrated the importance of situational context for personal search.

Context-aware and proactive recommender systems. Recommender systems are another broad domain of applications that greatly benefit from context-awareness. Since Adomavicius *et al.* [1] proposed the concept of context-aware recommender systems, various algorithms have been developed to address the problem of explicitly modeling user contextual preferences, *e.g.* the N-dimensional Tensor Factorization model by Karatzoglou *et al.* [19]. Later, Rendle *et al.* [28] developed and applied the Factorization Machine (FM) to model contextual information and empirically showed that the FM model outperforms the Tensor Factorization model in both prediction quality and runtime. We also used this state-of-the-art FM model for one of the recommenders in our experiments. Context-awareness is closely related to the concept of *proactive recommendation* in recommender systems where the system pushes recommendations to users when the current context seems appropriate [9, 16]. In proactive recommender systems, the appropriateness of a recommendation is sensitive to determining what the current context is and whether the context is the right trigger. In CAPERS, the system has fairly accurate information on the current user activity or the upcoming future activities (from the user's calendar) and hence primarily avoids the problem of sensing context in the first place.

Personal recommender. There is less prior literature on personal recommendation than personal search. The widely used collaborative filtering model [29] relies on social collective intelligence, *e.g.*, the system might recommend items to a target user that a similar set of users (neighbors) liked before. However, in a personal recommender setting, *e.g.* CAPERS, users only have access to and interact with their own items. PocketLens [24] is an early work in this domain by Miller *et al.* where they design a system that can make recommendations on top of a peer-to-peer architecture while only storing personal information locally. However, the algorithms essentially still rely on nearest neighbor computations.

Email re-finding: organization and search. The study of people's diverse use of email traces back to Mackay's work [22]. Whittaker [35] later proposed the concept of "email overload". Sixteen years later, Grevet *et al.* [17] showed with a qualitative analysis on Google's Gmail that email overload, both in terms of volume and function, is still a problem today. People typically rely on email organization or search when they need to access their previously received emails (*i.e.*, email re-finding). Whittaker *et al.* [34] showed that although some users spend considerable preparatory effort creating complex folder structures to help with re-finding, these preparatory behaviors are inefficient and do not improve retrieval success compared

with email search and threading which promote more effective finding. When users issue search queries in their email inbox, as stated by Carmel *et al.* [10], they usually have a high expectation on the recall of the results, *i.e.*, whether the success rate is guaranteed if the target is indeed in the search results. This makes the time-based ordering of the search results particularly important. In their recent work [11], they investigated a mixed approach of promoting the most relevant results, to which they refer as "heroes", on top of time-ranked results. They used a bifurcated list and found that Heroes-Dup in which the top relevant results are duplicated in the time-based results is the most effective approach compared with two other alternatives. This research inspired our work in which we also deployed a condition that promotes "heroes" while maintaining time-based ordering for the rest, although we made substantial modification specifically tailored to our recommendation setting.

Email extraction and classification. Researchers have been developing techniques to automatically extract important information from increasing amounts of machine-generated emails and classify them into different categories for users to help with the "overloaded" email usage. Wang *et al.* [33] designed an active learning model that considers collaborative filtering, implicit feedback, and time sensitive responsiveness features of broadcast emails for broadcast email prioritization. Avigdor-Elgrabli *et al.* [4] proposed a structural clustering method which leverages the HTML structure common to messages generated by the same mass-sender script. Ailon *et al.* [3] proposed to thread machine-generated emails by their causal structure, because a sequence of machine-generated emails are caused by a few related user actions.

Email recommendation. There is not much prior literature specifically on email recommendation to the best of our knowledge. Ning *et al.* [26] proposed Context-Aware Resource Recommendation Systems for meeting users' goals with a system component predicting user goals. That is, the context in their work mainly focuses on high-level professional goals. Dumais *et al.* [13] built a system called Implicit Query (IQ) that searches and presents related information (e.g. email messages, calendar appointments, contacts, web pages, news etc.) for the current email message being drafted. Rhodes [30] broadly defined and explored Just-In-Time Information Retrieval agents (JITIRs) back in 2000, but did not specifically study how to find useful emails based on people's upcoming calendar appointments.

In contrast to previous work, we investigate how contextual signals relying on people, time, and content impact proactive recommendation for email. While these factors have been investigated in an email search setting, the impact in a recommendation setting is generally under-explored. Furthermore, as both email recommendation and personal recommendation are relatively understudied, this expands work in those critical areas. Finally, through our exploration, we not only seek to understand the key factors that impact recommendation quality but also determine the impact of different modeling choices in this system cold-start scenario.

3 SELF-REPORTED EMAIL-MEETING HABITS

In order to understand the relationship between email access and meetings (RQ1 and RQ2) and to inform our study design and system deployment, we conducted a survey. For this, we randomly selected people from a large U.S.-based technology company (limiting the recruitment within U.S. only) and asked them survey questions on their meeting preparation and follow-up habits and how emails are used for their meeting preparation or follow-up.

In total, we gathered 592 complete responses for the survey. To ensure we have not gathered a narrow slice of behaviors specific to certain roles or demographics, we also asked for demographic/role information. Among our participants, 63.4% are male, 35.2% female (others prefer not to answer). In terms of age, 29.8% are 25-34, 29% 35-44, 22.6% 45-54, 10.6% 18-24. The participants have diverse roles in the organization, including: 21.7% program or project managers, 19.8% software developers, 8.9% sales, 6.6% IT support, 4.7% marketing and various others.

We found from the survey that 14.9% of participants have more than 5 meetings in a work day, 19.5% have 4-5 meetings, 41.5% have 2-3 meetings, 15.0% have one meeting, and 9% have less than one.

3.1 Email Access Related to Meetings

The following are two key survey questions that we ask regarding **RQ1**: *how much email access is related to meetings?*

- *Frequency of Email Access for Meetings*: How often do you prepare for meetings by reading, replying or forwarding relevant emails? (with single-selection options: "always", "frequently", "regularly", "occasionally" or "never")
- *The Last Email Access*: What was your last email access about? (options given for meeting related and non-meeting related behaviors)

We found that 68.4% of the participants prepare for meetings by accessing emails regularly or more often than regularly. Specifically, 11.5% reported "always", 20.2% "frequently", 36.7% "regularly", 30.0% "occasionally" and 1.6% reported "never". In terms of last email access as a proxy for overall email behavior (a multi-selection question: "What was your last email access about?"), responses indicated that a large proportion (32.0%) of email access was meeting-related. This both indicates that meeting-related email access is frequent while being consistent with the intuition that the majority of the email access is checking new unread emails (79.0%) or writing new emails (56.5%). In summary, we find that meetings occur frequently for many role types in an enterprise setting and that both meeting preparation and access of emails related to meeting preparation are frequent behaviors. While the survey is limited to one single enterprise, the diversity of roles and demographics make it likely that the results generalize to many enterprise settings (at least where information work is prevalent).

3.2 Meeting Types which Affect Preparation

To answer **RQ2**, we asked participants a multi-selection question in the survey: "What factors might affect whether or how far in advance you prepare for a meeting?" Almost all participants (96.8%) agreed that the role (whether organizer or presenter) is an important factor. Other factors include the amount of prior information/knowledge needed (80.3%), the organizational relationship with meeting attendees, *e.g.* direct report or manager (79.4%) and the number of attendees, *e.g.* 1:1 or group meeting (44.8%).

To ground their responses in a concrete scenario, we additionally asked participants to recall their most recent previous meeting in

which they are not the organizer or presenter and which lasted at least half an hour. For those that recall such a meeting, we ask details such as whether they prepared for the meeting and what type of meeting it was, *e.g.*, how many people were involved and who to meet with (a manager, direct report or colleague).

We found that 93.7% of participants can recall such a meeting. Among them, almost half (48.7%) responded that they prepared for the meeting, which shows that meeting preparation is not limited to organizers or presenters. These meetings varied from small group meetings with 3-6 attendees (48.2%), large group meetings with more than 6 attendees (35.7%), 1:1 with a colleague other than the supervisor, manager or direct reports (7.1%), 1:1 with the supervisor or manager (5.4%) and 1:1 with a direct report (1.4%). We found that people are significantly more likely to prepare for small group meetings (log odd-ratio change: 0.520, p=0.006 from a logistic regression model predicting whether the participants prepare with the meeting types as input) or 1:1 meetings (log odd-ratio change: 0.771, p=0.004 from the same model) compared with large group meetings. We did not find significant differences for meetings involving manager or direct reports.

Taken together, these results imply a person's role with respect to the meeting impacts when and whether a person prepares for a meeting, but moreover, meeting preparation is not limited to simply meetings for which a person is a presenter or organizer. In our system design, we take the results in this section into account in two ways. Our people matching features in Section 5.2 can differentially weight emails from the organizer versus attendees of a meeting. Also, the system design described in Section 4 defaults to the next upcoming meeting to enable proactive recommendation by default but also allows the user to select the meeting to allow user-initiated contextual recommendation. Given that the survey results in this section indicate predicting whether or not a person will prepare for a meeting may in itself be difficult, this enables separating the challenges and focusing on recommending relevant emails given a meeting context.

4 CAPERS PROTOTYPE

To conduct experiments driven by actual user behavior (answering RQ3-5), we developed CAPERS as an add-on to Microsoft Outlook. Outlook provides public APIs so that the general public can develop additional features on top of the standard Outlook without any access to client code. In addition, we can ask users to install the add-on without publicly releasing it into the Outlook add-on store by sending an offline installation file to organizational users with the organization's permission and the user's permission. After installing the add-on, users can click to activate the email recommendation pane (which is illustrated in a mock-up in Figure 1).

The interface design was informed based on the survey and a *pilot study*, which refers to the first 20 trials of the task-based experiment (see Section 6.1). The recommendation pane has two panels where the bottom panel lists the upcoming meetings of the user ordered by time and the top panel presents a list of email recommendations for the meeting selected in the lower panel. In desktop or laptop screens with standard sizes, around 2-3 emails and meetings can be displayed and users can scroll down in either panel to see more emails or meetings.

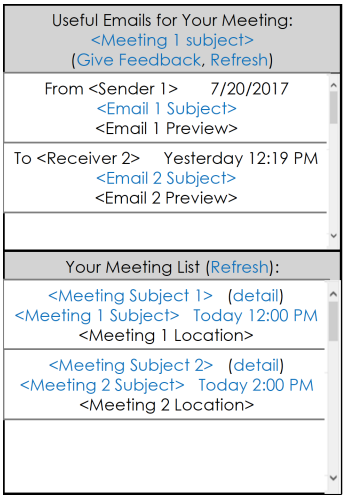

Figure 1: The CAPERS interface mock-up.

By default, the current or the first upcoming meeting is selected and users can switch focus to a different meeting by clicking the meeting subject (clicking a separate link "detail" opens a pop-up with the details of the meeting, *e.g.* content). Both the meeting and email lists are refreshed with a fixed time interval (five minutes in our system). Users will not perceive the refresh unless the list of meetings changes or the list of recommendations changes. Users can also manually refresh the meeting list or email recommendations by clicking the "Refresh" link.

Users can click the subject of the emails to open a pop-up with the full email. A shortened version of the email content is displayed along with the email subject and users can preview the full content of the email by hovering on the shortened preview. Because of this design, we log both clicking and hovering actions of the users. For hovering, we only trigger the logging when the hover lasts for more than a second.

5 CAPERS RECOMMENDATION

5.1 The Recommendation Flow

Algorithm 1 illustrates the recommendation flow in CAPERS when a meeting is selected by the user or by default. The input of the algorithm asks for an additional two weeks of meetings in the current user's calendar and it assumes a back-end email search service which has been provided by the email client. Note that for reproducibility, our entire implementation uses only public functionality of the email client and search APIs. Likewise we have selected a small but rich feature set that can be completely described here as well as being common in recommendation algorithms. Thus, our entire study can be replicated by any researcher or organization.

5.2 The Four Categories of Factors

Regarding **RQ3**, there are many factors that can potentially affect an email's usefulness to preparing for a meeting. First, it is intuitive to hypothesize that the amount of email content is an important factor. Second, prior work has emphasized the importance of recency in both email search [10] and personal search [12], *e.g.*, Dumais *et*

Algorithm 1: The CAPERS recommendation algorithm.

Data: m: the target meeting with the information of subject, body and people involved. M: the upcoming two weeks of meetings with the same information from the current user.

1 Trigger Generation: Extract key words from the meeting m's subject and body text (taking the top three key words ranked according to TF-IDF [31] with respect to the content of M.).

2 Candidate Generation: Assuming an email search service provided by the email client, construct a query with the disjunction of the extracted key words and the people involved in the meeting m (including attendees and the organizer) and query the search service. Denote this set of retrieved emails as the Candidate Set S.

3 Candidate Ranking: Rank the emails in S according to different ranking algorithms as illustrated for the different conditions of recommenders in Section 5.3. Denote this ranked set of emails as R.

Result: R

Algorithm 2: The online learning algorithm for learning the linear blending weights of the four categories of factors predicting whether an email is "Useful" (labeled as 1.0), "Maybe Useful"(labeled as 0.5) or "Not Useful" (labeled as 0.0).

Data: A sequence of (x_t, y_t) for $t = 1, ..., T$ where x represents the features in Table 1 and y represents the user's usefulness feedback.

Parameters: $\beta = 1$; d is the dimension of x_t.

Initialization: $A = \beta I_d$, $b = 1_d$, *i.e.*, equal weights for all input features.

1 **for** $t = 1, 2, ..., T$ **do**

2 $\quad \theta_t = A^{-1}b$

3 $\quad$ Let the user's feedback on the usefulness be y_t

4 $\quad A \leftarrow A + x_t^T x_t$

5 $\quad b \leftarrow b + x_t y_t$

6 **end**

Result: θ_T and the linear model is represented as $f(x, \theta) = \theta^T x$

al. found that in the personal search system SIS, users accessed recent items more frequently and issued more queries in which they sorted the results by date. They also found that queries in the system involving a name accounted for 25% of the queries. Harvey *et al.* in [18] showed that people are very important to email search. Similarly, we hypothesize that both recency and the match of who are involved in the email and meeting may play a significant role. As demonstrated in the survey, the role of the involved person might be important as well, *e.g.*, whether the email is sent by the organizer or attendee. Last but not least, content match is probably important for emails to be relevant and useful similar to the match between query and search results in a search system.

We summarize the four categories of factors in Table 1 – including how we operationalize each of them in CAPERS. These features are non-exhaustive examples of some of the ways these four factors could be captured, and it is interesting future work to explore many other ways of capturing these factors.

From a machine learning point of view, it is possible to learn low-dimensional vector representations for words or sentences over the target domain of emails or meetings and then train a model to take in those embedding features as input to predict usefulness, *e.g.* similar to the word2vec model [23]. This approach on email data however is subject to private information leaking [6]. It is also possible to directly learn how likely an email will be useful in general, *e.g.*, Elsweiler *et al.* [14] showed that the same emails tend to be re-found again over time. However, our factor design targets at gaining an initial understanding of how different categories of factors potentially affect usefulness. These features are global across users without revealing personal or sensitive information in emails. This enables us to satisfy the privacy policy constraint of the organization while not introducing sophisticated modeling or featuring approaches.

5.3 The Six Recommenders

In order to answer our last two research questions (**RQ4** and **RQ5**), we examined six recommenders. The first two (*Time* and *Search*) provide non-learning baselines (*i.e.*, not based on machine learning models), the next two (*Static Linear* and *Static Hero*) provide a learned baseline that does not update with further interaction and the same but with a recency-based re-ranking inspired by the literature. The final two (*Online Linear* and *Online FM*) provide information on the impact of online learning in this limited data scenario as well as second-order interaction modeling. In all cases, both sent and received emails are considered as candidates since a person may both find utility in the emails they have sent or received. We now describe these in more depth.

For the **Time** condition, the email recommendations consist simply of the most recent emails ordered from most recent to least recent. This provides a simple baseline only based on recency. It is a weak baseline but mimics the default presentation of emails in a typical email client without search.

In the **Search** condition, we aim to simulate a simple search a person might perform to find items for a meeting but without the benefit of learning which factors most contribute to relevance. To construct the query, three key words are picked according to the TF-IDF criterion [31] from the TF of the meeting's subject and IDF from among all the meeting subjects of the user's upcoming two weeks of meetings. It then proactively issues the query (concatenating the three key words with spaces) to the back-end email search service. This is trying to mimic the results of user search but may differ from actual user search, *e.g.*, users do not have the exact information in their mind for upcoming meetings but they can search more effectively than using three key words. We consider this as an experimental condition, instead of a baseline, because it is proactive. Note that no people information is used for the search in this condition.

The remaining conditions are learning the effects of the four categories of factors (Table 1) in predicting the usefulness of emails

Table 1: The extracted features operationalizing four categories of factors in predicting an email's usefulness to a meeting.

Factor Category	Feature Name	Definition
Amount of Content	*EmailLength*	#words in both the subject and body; converted to quantile in the candidate set
	HavingAttachment	whether the email has attachments
Email Recency	*RecencyScore*	the time difference between the sent time of the email and the current time; converted to quantile in the candidate set
Content Match	*ContentMatchScore*	the cosine similarity between the word count vectors of the email and meeting's content including both the subject and body
	SubjectMatchScore	the cosine similarity between the word count vectors of the email's content (both the subject and body) and the meeting's subject
People Match	*PeopleMatchScore*	the Jaccard index of the two sets of people involved in the email and the meeting
	FromAttendee	whether the email was sent by one of the attendees in the meeting
	FromOrganizer	whether the email was sent by the organizer of the meeting

to meetings. They differ in how an initially retrieved email candidate set S is ranked, i.e. the Step 3, in Algorithm 1

We are faced with a system-level cold-start problem for which no historical explicit or implicit usefulness labels are available. We approach this problem through a pilot study (which refers to the first 20 trials of the task-based experiment in Section 6.1) to collect initial usefulness labels for email-meeting pairs from potential users.

With the initial labels, we first train a model linearly blending the factors in Table 1 through Algorithm 2 by scanning the labeled data points once to predict the usefulness value of an email to a meeting (see Section 6.1 for the scale of the usefulness labels). This algorithm is adapted from LinearUCB [21] to learn a linear regression model. The learned blending weights do not change any more after this initial training (see Table 2, the second column for the weights). This condition is referred to as **Static Linear**.

Faced with the system cold-start, we also study the *effects of online learning to improve the user experience (*i.e. *user perception and engagement).* Therefore, we design an **Online Linear** condition that continues learning by running Algorithm 2 whenever users add more labels into the system (see Table 2, the third column for the final learned weights).

Note that the full LinearUCB has an exploration parameter to tune the Explore & Exploit process. Here we use zero exploration in Algorithm 2 to focus solely on the value of updating the model. We leave studying the impact of exploration as future work.

The **Static Hero** condition (for **RQ5**) is inspired by the recent work of Carmel *et al.* [11] in email research. Time is a critical factor in email ranking because people have a strong preference for time-based ordering of their emails even in the context of a given search query [10]. In the study, they demonstrate the benefits of a condition where the top two relevant search results with respect to the search query are promoted to the top, following with the complete set of relevant results ordered based on time. Similarly, we want to answer whether recency (*i.e.*, ordering by time) plays a similar critical role in email recommendation. Therefore, this condition post-processes the ranking of the Static Linear condition by picking the top two emails according to predicted utility first and sorting the rest by their sent time with more recent at the top.

Lastly, we study a generalized version of the classical SVD model ([20]), **Factorization Machine (FM)** [28], taking the same factors in Table 1 as features but additionally modeling the second-order interactions between all pairs of the input features compared with linear models as shown in Equation 1. $W = (w; U)$ are the model parameters. w is a parameter vector with the length the same as the number of features, d. U is a $d \times k$ matrix where k is the dimensionality of the latent factors representing each feature i ($i = 0, ..., d$). By convention, we use $x_0 = 1$ so that w_0 represents the learned intercept or global bias of the model. Let y be the users' usefulness feedback, the model minimizes a L_2 norm (least-squared) loss function through the widely used Stochastic Gradient Descent (SGD) algorithm.

Since we focus on studying the effects of online learning on user experience, the FM model is trained by running the SGD algorithm once for each label obtained in real-time, for which we name as **Online FM** (parameters: $k = 10$, learning rate $\eta = 0.01$). We conduct post-experiment analysis to understand the performance gap between the current online learned model and the optimal one trained until convergence. Future work is necessary to study other online optimization algorithms with better optimality guarantees [32].

$$f(W, x) = f(w, U, x_i) = \sum_{i=0}^{d} w_i + \sum_{i<j, 1<=i,j<=d} U_i^T \cdot U_j \quad (1)$$

6 EXPERIMENT DESIGN

6.1 The Task-Based Experiment

We sent out email invitations asking potential participants (randomly selected from the same company) to install the email recommendation add-on. Right after the user installs the add-on, an icon button (with the text "Recommend Emails") will show up at a prominent position of the Outlook. If the user clicks it, the email recommendation pane as shown in Figure 1 gets displayed at the right of Outlook occupying around 1/3 of the screen width (this add-on is designed for desktop, laptop or web versions of Outlook, not mobile devices). In the first-run of the add-on, it asks for consent from the user on the service agreement and the logging and privacy policies. If the user agrees, it starts the following task-based

within-subject experiment[1], which includes explicit judgment of email usefulness to meetings and subjective evaluation of the six email recommenders.

- *Step 1*: We need your input to help the system learn to recommend useful emails for your meetings. Please select a meeting that you are most likely to prepare for. (The user's upcoming meetings are displayed after this prompt.)
- *Step 2*: For each of the displayed emails, please label them as "Not Useful", "Maybe Useful" or "Useful" for you to prepare for the selected meeting. (The list of displayed emails are generated by Algorithm 1 and the Static Linear ranker. Note that during the pilot study, *i.e.*, the first 20 trials, since no labels were collected yet, the weights for the input features in the Static Linear ranker are in an initialized state, *i.e.*, the input features have equal weights as shown in the initialization part of the Algorithm 2).
- *Step 3*: Based on your input, we have built six email recommenders. Please tell us your preferences on them. To give feedback, first select another meeting that you are most likely to prepare for. (The user's upcoming meeting list is displayed after this prompt excluding the previously selected meeting.)
- *Step 4(to 10)*: Here are the emails that Recommender k (k=1 to 6) thinks useful for you to prepare for the selected meeting. Please answer three questions about the recommended email list. (See below for the three questions.)

In Step 4-10, the order of the recommenders being presented is randomly shuffled. For each recommender, we measure three subjective metrics by using the following three likert-scale questions (with answers from "strongly disagree", "disagree", "neutral" to "agree" and "strongly agree").

- *Perceived Precision*: The recommendations contain some useful emails for me to prepare for the selected meeting.
- *Perceived Recall*: The recommendations contain all needed emails for me to prepare for the selected meeting.
- *Intention to Use*: I would like to use this recommender in my daily work.

At any step, users have the option to skip all the remaining steps and directly go into the following field experiment.

6.2 The Field Experiment

After users finish the task-based experiment (or skip it), each user is randomly assigned into one of the five recommenders persistently (to avoid dropout we exclude the Time condition because it performed poorly based on the feedback of the pilot study). We measured the following measurements in a given time period (one month) for each user since they joined the field experiment.

- *numRequests*: The number of email recommendation requests sent out by the user using the email recommender add-on during the measured time period. Each user has one observation for this metric and higher values suggest higher utilities (*i.e.*, users find it useful to keep requesting email recommendations for more meetings; Note that asking for more email recommendations for the same meeting does not count as a recommendation request because all recommendations are preloaded for the same meeting).
- *actionRate*: The percentage of email recommendation requests that attract any clicks or hovers among all the email recommendation requests sent out by the user during the measured time period. Each user has one observation for this metric and higher values suggest higher precision.

[1]In the add-on, we provide a link for users to revoke consent and delete all their data whenever they want to by first clicking the link and then confirming the deletion.

Table 2: The weights for the four categories of factors. The second column is used by the Static Linear condition. The third column is the final weight of the Online Linear condition. The fourth column is the coefficient and standard error from an analytical mixed-effect cumulative link model (treating user ID as a random intercept and usefulness response as ordinal value). Significance codes: * p<0.05, ** p<0.01, *** p<0.001.

Feature Name	Weight	Final Weight	Coefficient (Std.)
Intercept	-0.486	-0.166	N.A.
SubjectMatchScore	0.212	0.650	4.75 (0.601) ***
RecencyScore	0.684	0.164	0.892 (0.244) ***
EmailLength	0.152	0.102	0.737 (0.239) **
HavingAttachment	0.100	0.105	0.428 (0.252)
FromOrganizer	0.206	0.084	0.357 (0.164) *
FromAttendee	0.023	0.014	0.233 (0.216)
PeopleMatchScore	0.486	0.289	0.185 (0.237)

7 EXPERIMENT RESULTS

7.1 The Pilot Study and Bootstrapped Models

In the pilot study (running the task-based experiment initially with 20 participants), we gathered 366 labeled data points in total. Table 2 lists the weights that are learned by Algorithm 2, which are used by the Static Linear condition directly and also used as an initial state for the Online Linear condition before online updating. The table also shows that the final weights of the Online Linear condition, which updates using all collected labeled data points so far, are slightly different from the Static Linear condition. The Static Linear condition emphasizes most the *RecencyScore* while the Online Linear condition changes to emphasize *SubjectMatchScore* the most, which is consistent with the coefficients of a more sophisticated ordinal regression model (built for offline analysis) as shown in the fourth column.

7.2 The Data and Analysis

As of September 28 2017, we have had 141 installs of the add-on in total. This gives us 1,518 labeled data points from 84 participants. There are 381 field recommendation requests from 75 users, which gives roughly 15 participants for each of the five recommenders. We received 641 responses from 44 participants for the six recommenders regarding the three subjective metrics, which gives us roughly 36 responses to each question and condition. Note that participants could opt out from the task-based experiment or even stop using the add-on after installation, which explains the smaller number of participants who had labels and field activities compared with the number of installs. When the response data are ordinal, we use the cumulative link mixed-effect regression model [2] to

conduct the analysis. For *numRequests*, we used negative binomial regression while for *actionRate*, we used linear regression analysis.

7.3 Factors that Affect Usefulness

Based on the pilot study, we develop the following two hypotheses regarding how the four categories of factors might affect the email usefulness (**RQ3**).

H1: Matching by the meeting subject is better than matching both the meeting subject and the body text. This results from the observation that the meeting body text contains a much broader set of information and hence adds noise to the content matching process. Instead, the meeting subject is a naturally filtered set of key words by users regarding the main content of the meeting.

H2: There are interactions between the amount of email content or email recency and people or content match. Intuitively, when the match between the emails and the meeting is low, it does not matter how much information is contained in the email or how recent the email is.

Table 3: The coefficients and standard errors of the four categories of factors from a similar ordinal regression model as in Table 2. Significance codes: * p<0.05, ** p<0.01, *** p<0.001.

Feature Name	Coefficient (Std.)
EmailLength	1.72 (0.326) ***
RecencyScore	1.10 (0.333) ***
SubjectMatchScore	3.53 (0.638) ***
PeopleMatchScore	1.58 (0.421) ***
RecencyScore:LowContentMatch	-0.319 (0.382)
RecencyScore:LowPeopleMatch	-0.167 (0.384)
EmailLength:LowContentMatch	-1.21 (0.379) **
EmailLength:LowPeopleMatch	-1.01 (0.388) **

We tested these two hypotheses by analyzing all the usefulness-labeled data points we collected. Table 2 (the fourth column) and 3 list the results. Our hypotheses are supported by the data. Specifically, the match score between an email with the meeting content (including both subject and body text) is not statistically significant when controlling for the match score between the email with the meeting subject only, which itself is statistically significant. Our second hypothesis is partially supported as well but the results are more nuanced. We find significant interaction effects between the email length and content/people match, while email recency is a strong positive factor no matter when the content or people match score is high or low.

Besides these two hypotheses, Table 2 (the fourth column) and Table 3 show that both ContentMatch (specifically SubjectMatchScore) and PeopleMatch have strong positive effects. Among all the factors we examined, SubjectMatchScore, *i.e.*, how well an email's content matches the subject of a meeting, is the most important one.

7.4 Linear, FM and Hero Models

To compare the six recommenders (**RQ4-5**), we first analyze the participants' responses to the three subjective metrics that we deployed in the task-based experiment excluding responses collected in the pilot study (because at that time, the models are not bootstrapped yet). The first three columns of Table 4 show the results of the analysis. As we see, the Search, Static Linear, Static Hero, Online Linear, and Online FM conditions are significantly better than the baseline Time condition in terms of perceived precision, recall and intention to use although they are not significantly different from each other. The Time condition performed poorly with 12% positive and 62% negative (others neutral) responses for the *intention-to-use* metric (note that this is not in conflict with the strong positive effects of Recency in Table 3 because all four categories of factors are taken into account there). Overall, we found that recommending some useful emails (precision; the best condition has 56% positive, 25% negative, others neutral) or having users develop intention to use the recommender in their daily work (intention to use; the best condition has 46% positive, 20% negative, others neutral) are easier problems than retrieving all the necessary emails (recall; the best condition has 31% positive, 49% negative, others neutral).

The last two columns of Table 4 show the results of the metrics *numRequests* and *actionRate*. We see that the Online Linear condition is significantly higher in terms of *numRequests* than all the other four conditions (which are not significantly different from each other), which suggests that it provides the most utility for its users compared with other recommenders. In terms of *actionRate*, the data shows a strong trend that the Search condition is lower than the other four conditions (because of the high standard errors, the comparison does not achieve statistical significance). The Online FM condition shows a trend of having a lower value than the other three conditions (the Static Linear, Online Linear and Static Hero).

To further understand the differences of the recommenders, we compute nDCG@k (Normalized Discounted Cumulative Gain for top k=5,10,15; higher values indicate better recommendation accuracy) for the four learning-based rankers (the Search condition can not be computed because it relies on the external search service from the email client) based on the users' ratings in the task-based experiment Step 2 (relevance values are "Useful"=1.0, "Maybe Useful"=0.5 and "Not Useful"=0.0 for computing nDCG). For the Online Linear and Online FM rankers, their models are updated with all labeled data points of all users before the first rating time of the list on which to predict. This is a proxy metric measuring the accuracies of the recommenders being used by users in the field experiment.

In order to know how online updating the FM model with previous labels only once (the case in the experiments) affects the model performance compared with training until the FM model converges with the same set of parameters ($k = 10, \eta = 0.01$; the convergence criteria is that the loss on the evaluated labels is not decreasing), which we name as **Batch FM**, we also compute the nDCG for Batch FM. Note that this is an expensive approach because it fully re-trains the FM model for each labeled email list to be evaluated on. We also evaluated the Batch FM varying parameters $k = 10, 50, 100, 150, 200$ to understand how sensitive the models are to specific parameter choices.

From Table 5, we see a consistent trend with the comparison of the metric *actionRate*, *i.e.* the Online FM model has lower accuracy than the linear models, which themselves are not substantially different from each other. However, the full-trained Batch FM model achieves similar accuracy in terms of nDCG as the linear models,

Table 4: The comparison of the six recommenders. The first three columns are the effect sizes (and std.; in the log-odd-ratio scale) of all other conditions relative to the Time condition from three cumulative link models. The other two columns have the means and standard errors of *numRequests* (from a negative binomial regression model and varying baselines for comparison) and *actionRate* (from a linear regression model). Note that Time condition was not deployed in the field experiment. Significance code: * p<0.05, ** p<0.01, *** p< 0.001.

Recommender	Perceived Precision	Perceived Recall	Intention to Use	Mean (std.), median of numRequests	Mean (std.) of actionRate
Search	1.66 (0.492) ***	1.18 (0.521) *	1.81 (0.516) ***	3.80 (1.39), 2.0	0.104 (0.176)
Static Linear	1.88 (0.492) ***	1.68 (0.499) ***	1.60 (0.494) **	5.05 (0.880), 3.0 <Online Linear**	0.518 (0.078)
Static Hero	1.93 (0.485) ***	1.32 (0.501) **	1.59 (0.509) **	4.87 (1.35), 4.5 <Online Linear*	0.417 (0.124)
Online Linear	2.03 (0.499) ***	1.63 (0.518) **	1.86 (0.514) ***	**11.1 (2.78), 7.0** >Search*	0.485 (0.124)
Online FM	1.67 (0.466) ***	1.51 (0.502) **	2.25 (0.511) ***	3.14 (0.716), 3.0 <Online Linear***	0.343 (0.094)

Table 5: The nDCG@k(k=5,10,15) for the Static Linear, Static Hero, Online Linear, Online FM and Batch FM rankers.

Recommender	nDCG@5	nDCG@10	nDCG@15
Static Linear	0.837	0.807	0.804
Static Hero	0.832	0.810	0.785
Online Linear	0.835	0.805	0.802
Online FM	0.793	0.768	0.753
Batch FM, k=10	0.819	0.795	0.794
Batch FM, k=50	0.829	0.800	0.798
Batch FM, k=100	0.833	0.800	0.797
Batch FM, k=150	0.826	0.800	0.796
Batch FM, k=200	0.828	0.798	0.795

which suggests that a better trained FM model could achieve similar (although probably not better) *actionRate* or *numRequests* as the linear models. The best parameter k is 100 but the accuracy is not substantially different from $k = 10$.

8 DISCUSSION

As shown in the survey results, a large proportion of email access is related to meetings, which renders CAPERS a worthy effort not only in terms of scientific understanding but also of practical value in improving email information access. We also find that a large proportion of email access is related to user tasks in their daily jobs, which suggests that it would be useful to be able to predict user tasks and proactively recommend emails accordingly in future work. We also find that participants are more likely to prepare for small-group or 1:1 meetings than large-group meetings. An interesting thread of future work is to build similar predictive models to rank the meeting list by how likely the user is to want to prepare or follow up for the meeting, *e.g.*, utilizing the recently proposed method by Bahrainian *et al.* [5].

We find that all four categories of factors have significant positive effects on email usefulness for preparing a meeting, *i.e.* emails that are more recently received, more informative, with better content and people match are more likely to be useful. We also found an interaction between the email length and the content and people match between the email and meeting. Specifically, the email length is more important when the people or content match is high than when the match is low.

We show that an online learned linear model through an online learning algorithm (adapted from LinearUCB [21]) provides the most utility (reflected by the *2x* positive engagement metric *numRequests*) for users compared with a linear model without online updating. We did not find significant differences in terms of either perceived precision, perceived recall or post-experiment accuracy metric nDCG between the two conditions. We think the increased engagement could be an effect of the dynamic nature of the recommender which is hard to capture by evaluating on static recommendations. It is interesting future work to test this hypothesis through a field survey asking how users perceive the recommendation change. This work suggests that designing better online learning algorithms for more complex state-of-the-art models *e.g.* Factorization Machine [27] (and Gradient-Boosted Decision Trees [15]) is a promising research direction.

We limit ourselves to online updating the models with historical labels only once because of our research focus. Post-experiment analysis showed that this choice negatively affects the performance of the FM model. However, even if we fully train the FM model until convergence, the accuracy is not better than linear models (given the same set of features). It is likely because the small scale of training data in this system cold-start setting is not enough to gain benefits from the FM model.

We did not find a significant difference between the Static Hero condition and the Static Linear condition. We see a slight decrease in terms of nDCG effectiveness in the Static Hero condition – which manipulates the Static Linear condition by sorting all but the top two recommendations by recency – but no improvement in engagement. This suggests email recency may be captured well in a recommendation setting as a normal factor rather than treating it specially in presentation. However, we recognize that in the original design of "heroes" by Carmel *et al.* [11], the promoted top relevant results are separately displayed from the time-based "all" results, which could be an important design factor. Future work is necessary to address this difference and look at other measurements like users' perceived readability because of breaking the temporal order.

9 LIMITATIONS AND FUTURE WORK

Our findings are based on an initial set of 141 users from a large technology company. Future research is necessary to study the system across different work environments and on a larger set of users. For the sake of user experience in the first-run of the add-on, we did not ask participants to rate the usefulness of each email recommendations generated by different algorithms (which could be a better design in future work).

10 CONCLUSION

In this work, we propose a Calendar-Aware Proactive Email Recommender System to address the email information overload problem. The system recommends potentially useful emails to users based on their upcoming calendar schedule with a special focus on meetings. We study the key factors by constructing and deploying a prototype named CAPERS and through the methods of a survey, a task-based experiment and a field experiment. We ask people survey questions on how they utilize emails to prepare meetings and find that a large proportion of email access (32.0%) is related to meetings. We ask users to rate email usefulness with respect to meetings that they are likely to prepare for and find that email recency, the amount of email content, calendar-email content match and calendar-email people match all have significant positive effects on usefulness. We compare a recommender based on a static linear combination of the four categories of factors and a recommender based on a dynamically online updated linear model and find that online updating significantly increases (2x) positive engagement and hence the utility. We did not see improvement when online learning a more complex Factorization Machine model in this cold-start scenario in terms of both accuracy and engagement. Unlike email search, we did not find significant improvement by further sorting emails by their recency when displaying the recommendations after ranking based on the static linear model which suggests that recency can be captured well by treating it similar to other factors in email recommendation. More generally, CAPERS demonstrates the rich possibility of incorporating contextual recommendation directly into productivity workflows and illustrates the potential for exploring other scenarios where contextual intelligence directly assists with information access.

REFERENCES

[1] Gediminas Adomavicius and Alexander Tuzhilin. 2011. Context-aware recommender systems. In *Recommender systems handbook*. Springer, 217–253.

[2] Alan Agresti and Maria Kateri. 2011. Categorical data analysis. In *International encyclopedia of statistical science*. Springer, 206–208.

[3] Nir Ailon, Zohar S Karnin, Edo Liberty, and Yoelle Maarek. 2013. Threading machine generated email. In *Proceedings of the sixth ACM international conference on Web search and data mining*. ACM, 405–414.

[4] Noa Avigdor-Elgrabli, Mark Cwalinski, Dotan Di Castro, Iftah Gamzu, Irena Grabovitch-Zuyev, Liane Lewin-Eytan, and Yoelle Maarek. 2016. Structural Clustering of Machine-Generated Mail. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. ACM, 217–226.

[5] Seyed Ali Bahrainian and Fabio Crestani. 2018. Augmentation of Human Memory: Anticipating Topics that Continue in the Next Meeting. In *Proceedings of the 2018 Conference on Human Information Interaction&Retrieval*. ACM, 150–159.

[6] Michael Bendersky, Xuanhui Wang, Donald Metzler, and Marc Najork. 2017. Learning from user interactions in personal search via attribute parameterization. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 791–799.

[7] Paul N Bennett, Kevyn Collins-Thompson, Diane Kelly, Ryen W White, and Yi Zhang. 2015. Overview of the special issue on contextual search and recommendation. *ACM Transactions on Information Systems (TOIS)* 33, 1 (2015), 1e.

[8] Paul N Bennett, Filip Radlinski, Ryen W White, and Emine Yilmaz. 2011. Inferring and using location metadata to personalize web search. In *Proceedings of SIGIR'11*. ACM, 135–144.

[9] Matthias Braunhofer, Francesco Ricci, Béatrice Lamche, and Wolfgang Wörndl. 2015. A context-aware model for proactive recommender systems in the tourism domain. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct*. ACM, 1070–1075.

[10] David Carmel, Guy Halawi, Liane Lewin-Eytan, Yoelle Maarek, and Ariel Raviv. 2015. Rank by time or by relevance?: Revisiting email search. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 283–292.

[11] David Carmel, Liane Lewin-Eytan, Alex Libov, Yoelle Maarek, and Ariel Raviv. 2017. Promoting Relevant Results in Time-Ranked Mail Search. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1551–1559.

[12] Susan Dumais, Edward Cutrell, Jonathan J Cadiz, Gavin Jancke, Raman Sarin, and Daniel C Robbins. 2016. Stuff I've seen: a system for personal information retrieval and re-use. In *ACM SIGIR Forum*, Vol. 49. ACM, 28–35.

[13] Susan Dumais, Edward Cutrell, Raman Sarin, and Eric Horvitz. 2004. Implicit queries (IQ) for contextualized search. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 594–594.

[14] David Elsweiler, Morgan Harvey, and Martin Hacker. 2011. Understanding refinding behavior in naturalistic email interaction logs. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. ACM, 35–44.

[15] Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics* (2001), 1189–1232.

[16] Daniel Gallego, Wolfgang Woerndl, and Gabriel Huecas. 2013. Evaluating the impact of proactivity in the user experience of a context-aware restaurant recommender for Android smartphones. *Journal of Systems Architecture* 59, 9 (2013), 748–758.

[17] Catherine Grevet, David Choi, Debra Kumar, and Eric Gilbert. 2014. Overload is overloaded: email in the age of Gmail. In *Proceedings of the sigchi conference on human factors in computing systems*. ACM, 793–802.

[18] Morgan Harvey and David Elsweiler. 2012. Exploring query patterns in email search. In *European Conference on Information Retrieval*. Springer, 25–36.

[19] Alexandros Karatzoglou, Xavier Amatriain, Linas Baltrunas, and Nuria Oliver. 2010. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering. In *Proceedings of RecSys'10*. ACM, 79–86.

[20] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009).

[21] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. 2010. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*. ACM, 661–670.

[22] Wendy E Mackay. 1988. More than just a communication system: diversity in the use of electronic mail. In *Proceedings of CSCW'88*. ACM, 344–353.

[23] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[24] Bradley N Miller, Joseph A Konstan, and John Riedl. 2004. PocketLens: Toward a personal recommender system. *ACM Transactions on Information Systems (TOIS)* 22, 3 (2004), 437–476.

[25] Kanika Narang, Susan T Dumais, Nick Craswell, Dan Liebling, and Qingyao Ai. 2017. Large-scale analysis of email search and organizational strategies. In *Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval*. ACM, 215–223.

[26] Ke Ning, Ruinan Gong, Stefan Decker, Yuliu Chen, and David O'sullivan. 2007. A context-aware resource recommendation system for business collaboration. In *E-Commerce Technology and the 4th IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services, 2007. CEC/EEE 2007. The 9th IEEE International Conference on*. IEEE, 457–460.

[27] Steffen Rendle. 2010. Factorization machines. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. IEEE, 995–1000.

[28] Steffen Rendle, Zeno Gantner, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2011. Fast context-aware recommendations with factorization machines. In *Proceedings of SIGIR'11*. ACM, 635–644.

[29] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. 1994. GroupLens: an open architecture for collaborative filtering of netnews. In *Proceedings of CSCW'94*. ACM, 175–186.

[30] Bradley James Rhodes. 2000. *Just-in-time information retrieval*. Ph.D. Dissertation. Massachusetts Institute of Technology.

[31] Gerard Salton and Michael J McGill. 1986. Introduction to modern information retrieval. (1986).

[32] Shai Shalev-Shwartz et al. 2012. Online learning and online convex optimization. *Foundations and Trends® in Machine Learning* 4, 2 (2012), 107–194.

[33] Beidou Wang, Martin Ester, Jiajun Bu, Yu Zhu, Ziyu Guan, and Deng Cai. 2016. Which to view: Personalized prioritization for broadcast emails. In *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1181–1190.

[34] Steve Whittaker, Tara Matthews, Julian Cerruti, Hernan Badenes, and John Tang. 2011. Am I wasting my time organizing email?: a study of email refinding. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 3449–3458.

[35] Steve Whittaker and Candace Sidner. 1996. Email overload: exploring personal information management of email. In *Proceedings of CHI'96*. ACM, 276–283.

[36] Hamed Zamani, Michael Bendersky, Xuanhui Wang, and Mingyang Zhang. 2017. Situational Context for Ranking in Personal Search. In *Proceedings of WWW'17*. International World Wide Web Conferences Steering Committee, 1531–1540.