

# Content Recommendation for Attention Management in Unified Social Messaging

Hongxia Jin

IBM Research Almaden  
San Jose, California 95120  
hongxia@acm.org

## Abstract

With the growing popularity of social networks and collaboration systems, people are increasingly working with or socially connected with each other. Unified messaging system provides a single interface for users to receive and process information from multiple sources. It is highly desirable to design attention management solution that can help users easily navigate and process dozens of unread messages from a unified message system. Moreover, with the proliferation of mobile devices people are now selectively consuming the most important messages on the go between different activities in their daily life. The information overload problem is especially acute for mobile users with small screen to display. In this paper, we present PAM, an intelligent end-to-end Personalized Attention Management solution that employs analytical techniques that can learn user interests and organize and prioritize incoming messages based on user interests. For a list of unread messages, PAM generates a concise attention report that allows users to quickly scan the important new messages from his important social connections as well as messages about his most important tasks that the user is involved with. Our solution can also be applied in other applications such as news filtering and alerts on mobile devices. Our evaluation results demonstrate the effectiveness of PAM.

## 1 Introduction

Over the past few years, we have witnessed the rapid development of online social networks and collaboration systems. People are increasingly collaborating with or socially connected to more and more people. For example, users may follow and respond to updates from social platforms such as Twitter and Facebook. Users may also subscribe to online newspapers and get alerts on what they care about everyday. Business users react to notifications from enterprise collaboration systems such as Jive Clearspace and Lotus Connections. Past research (Whittaker and Sidner 1996; Fisher et al. 2006) showed that users are already overwhelmed with only traditional email messages, not to mention the addition of more messages from social networks and collaboration systems. Unified messaging system provides a single interface for users to receive and process information from multiple sources. It is highly desirable to design attention management solution for unified message systems that

can help users easily scan and process the most important unread messages from email, social networks, collaboration tools, and customer relationship management (CRM) tools.

With the proliferation of smart phones, people are increasingly using multiple devices and choose to selectively consuming the most important messages on the go between different activities in their life. For example, when waiting in a line or during short breaks, a user may quickly process a small number of important messages using her smart phone; when back to office or home, she may deal with the remaining less important messages using her desktop or laptop computers, which have larger screens and more comfortable keyboard. As one can imagine it is painstaking to browse and process (reply to) a large number of new messages on the small screen of mobile devices. Therefore it is exceedingly desirable to organize and prioritize the messages for mobile users to enable this *selective processing* strategy.

In this paper, we present PAM, an intelligent end-to-end Attention Management solution that employs analytical techniques to help users mitigate information overload problem by enabling users to quickly scan the most important messages. We not only automatically categorize messages into different tasks but also prioritize messages in each task.

Specifically, PAM builds a rich user profile automatically from the user's historical data. In an unsupervised way it learns the task that the user is involved with, each task's importance level to the user and who is important in each task.

Secondly, given a list of unread messages, our system organize/cluster and prioritize messages based on the learned user profile. We categorize messages along three facets. The main facet that we use is "task", followed by "contact importance" and "time sensitivity". We designed a novel prioritization algorithm that can determine the order of the messages that the user should pay attention to within that task. The determination of the order of the messages within a task aims to maximize the benefits of user paying attention to that message. In this paper the benefits of paying attention to a message is determined based on multiple factors including the message's relevance to this task, the importance of its sender, the recency of the message and also, very importantly, what other messages the user has seen in this task. As one can imagine, reading one message may affect the subsequent messages that the user should read. We determine the order of the messages one by one in a greedy fashion.

We have evaluated our solution through a user study on real-world data and tested the accuracy of our categorization and prioritization algorithm. Experimental results demonstrate the effectiveness of our algorithms.

The remainder of this paper is organized as follows. We first survey some related work. We describe our overall system design in Section 3. We describe in details in Section 4 and 5 the underlying analytical techniques to learn rich user profile and our categorization and prioritization algorithms. After that, we present the evaluation results from a user study on PAM, and we conclude with future work.

## 2 Related Work

A lot of work has been done in the research community to tackle the information overloading problem.

Topic/task browsing has been widely studied by researchers in the literature. Researchers (Dumais, Cutrell, and Chen 2001; Käki 2005) have found advantages for topic browsing interfaces in a variety of user scenarios. In (Bernstein et al. 2010), Bernstein et al. proposed a system called Eddi, which allows users to browse social status streams (such as those from Twitter) by topics. They take advantages of search engines to discover topics in a set of status messages. Our work performs the task discovery without using any third-party resource help. More importantly, we prioritize messages inside each task. This is not supported in Eddi.

The categorization of email messages is not a new concept. Researchers (Mock 2001; Dredze, Lau, and Kushmerick 2006; Xiang 2009) have proposed a number of techniques to cluster email messages. Dredze, Lau, and Kushmerick in 2006 proposed several algorithms to automatically classify emails into activities based on multiple attributes. However, their algorithm requires users to manually assign the weight of each attribute. They also require users to manually label each activity. Our clustering algorithm automatically learn and categorize user activities as well as their importance levels. Furthermore their work did not support prioritization of messages but we do.

Researchers (Chirita, Diederich, and Nejd1 2005; Yoo et al. 2009) have studied prioritizing email messages to reduce email overload. In (Yoo et al. 2009), Yoo et al. proposed to mine social networks for personalized email prioritization. In (Horvitz, Jacobs, and Hovel 1999), Horvitz et al. proposed an email prioritization algorithm that takes diversity into account. However, they did not categorize messages into task-based categories. More importantly, they only evaluate the priority of individual emails independently while we determine the importance of a message sequentially one by one by taking into considerations on what messages have already been chosen.

There also exists work that assists users in task management in email (Freed et al. 2008; Faulring et al. 2010). A multi-agent system called RADAR was proposed in (Freed et al. 2008), which is capable of identifying task-relevant content from email messages and helping users to manage and execute tasks. The objective of our work is different from theirs. Our objective is to prioritize message so as to enable users to scan the most important messages in all tasks.

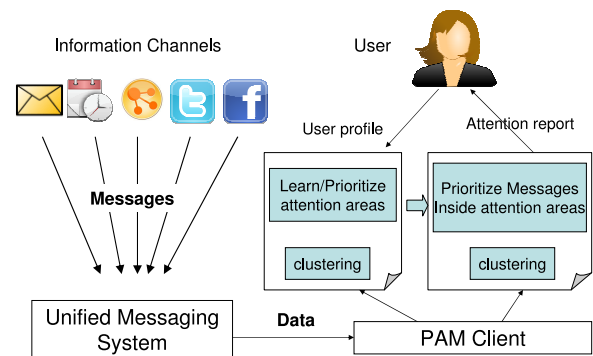


Figure 1: Dataflow among different components

## 3 Overall System and Preliminaries

The dataflow among different components in the overall system is shown in Figure 1. PAM is installed as a plug-in of the client of a unified messaging system on a user’s machine, for example a mobile device. The unified messaging system periodically retrieves new messages (feeds) from various information channels. When the user checks her unified message inbox on the mobile device, PAM creates an attention report from the current feeds and shows the report to the user.

Our system consists of two main designs, namely, modeling user interest from historical messages to build user profile, creating attention report for new messages based on the learned user profile. We will describe these designs in more details in the following sections.

We introduce a concept called *attention area* in our system. An attention area represents a task that may deserve the user’s attention. It consists of information on what the task is about and who the user is working with on that task. We utilize data clustering techniques to derive and organize messages into attention areas. Since this is a shared step in both profile building and attention report generation, we will devote rest of this section to introduce the concept of *attention area* and the clustering technique to derive attention areas.

### 3.1 Attention Areas

For convenience, we represent each unit of information about a user from various sources, such as a message or an online activity, as a *data item*.

**Definition 1 (Data Item)** An *data item* is represented as a tuple  $\langle W, U, t, r \rangle$ , where  $W$  is its textual content after stemming and removing stop words and is represented by a vector space;  $U$  is the people involved (e.g. the sender and the receivers of a message);  $t$  is its time-stamp; and  $r \in \{0, 1\}$  is a reaction flag indicating if the user is actively involves in the item.

For example, if the user composes or replies a message, the reaction flag is 1; if she ignores the message or clicks without responding, the reaction flag is 0. Intuitively, items with reaction flag 1 are more likely to be perceived as important by the user than those with reaction flag 0.

**Definition 2 (attention area)** An *attention area* is represented as a tuple  $\langle G, f_w, f_u, tl, s_p \rangle$ , where  $G$  is a set of data

items clustered into this attention area,  $f_w$  and  $f_u$  are functions that return the weights of a given word or a given person in the attention area, respectively,  $tl$  is the label, and  $s_p$  is a real-number importance score.

An attention area contains information such as keywords, people, timeline, and so on. PAM selects a label for each attention area. It may also measure and store the importance of each attention area. We derive *attention areas* from clustering of a user’s available data items.

### 3.2 Clustering Data Items into Attention Areas

We employ the hierarchical clustering strategy, which recursively merges item sets until there is no pair of item sets whose similarity degree exceeds a certain threshold. An initial value of the threshold may be determined by training.

In our system it is possible to use a topic modeling approach such as LDA (Blei, Ng, and Jordan 2003) models together with a trained classifier such as Support Vector Machine (SVM) to categorize future messages into the created topics. We choose to use a dynamic clustering strategy over this kind of hybrid approach because it unifies the topic deriving with classifying new messages into the existing topics/tasks. We also want to take advantage of the dynamic process to facilitate the incremental/evolving updates on new tasks from new data.

Given a group  $G_i = \{e_1, \dots, e_m\}$  of data items, we concatenate the textual contents of  $e_1, \dots, e_m$  to create the textual content  $W_i$  of  $G_i$ . The multi-set  $P_i$  is the union of the people in  $e_1, \dots, e_m$ . Similarly, the timeline  $T_i$  of  $G_i$  is a multi-set containing all the time-stamps of the items in  $G_i$ .

Given two groups  $G_1$  and  $G_2$  of data items, the similarity between  $G_1$  and  $G_2$  is computed as a linear combination of the similarities between textual contents, people, and timelines. Most real-world events and projects do not last forever. If there is a long time span between two items, it is unlikely that they belong to the same topic.

$$\begin{aligned} sim(G_1, G_2) = & \beta_1 \cdot sim(W_1, W_2) + \beta_2 \cdot sim(P_1, P_2) \\ & + \beta_3 \cdot sim(T_1, T_2) \end{aligned}$$

where  $\beta_1, \beta_2, \beta_3 \in [0, 1]$  are the weights of textual content, people, and time, respectively.  $\beta_1 + \beta_2 + \beta_3 = 1$ .

We used cosine similarity to measure topic similarity and use Jaccard distance to measure people similarity. To measure the time distance between  $G_1$  and  $G_2$ , we have

$$sim(T_1, T_2) = \alpha^{d(tc_1, tc_2)}$$

where  $tc_1$  and  $tc_2$  are the means of the time-stamps in  $T_1$  and  $T_2$  respectively,  $d(tc_1, tc_2)$  returns the difference (number of days) between  $tc_1$  and  $tc_2$ , and  $\alpha \in (0, 1]$  is a decay factor. The larger the difference is, the smaller  $sim(T_1, T_2)$  is.

## 4 Build User Profile

In our system we learn about a user by mining user historical data and automatically clustering his data into a number of activity areas as described above. We not only want to know who the user is working with and on what tasks; we also want to know how important (i.e., the user’s interest level) is each task and who is important in each task.

### 4.1 Importance Measurement on Attention Areas

After the clustering algorithm clusters a set of data items  $G$  into an attention area, we need to measure the weights of the words and the people, compute the importance score of the attention area, and select a representative label. This way we derive the user’s interest level on this area and who are important inside this area.

First, given a word  $w_i$ ,  $f_w(w_i) = 0$  if  $w_i$  is a stop word or a common English word; otherwise,  $f_w(w_i)$  is the number of data items in  $G$  that contain  $w_i$  in their textual contents. Similarly, given a person  $u_j$ ,  $f_u(u_j)$  is the number of data items in  $G$  that contain  $u_j$ .

Second, we measure the importance of the attention area. Let  $L = \{e_1, \dots, e_m\}$  be the list of the data items in  $G$  sorted by recency, such that  $e_i$  is more recent than  $e_j$  when  $i < j$ . In reality, we may only consider a number of the most recent data items (say,  $m = 50$ ) instead of all the items in  $G$ . The Normalized Discounted Cumulative Gain (NDCG) is often used to measure the effectiveness of search and recommendation algorithms. In our work, we employ NDCG to estimate the importance of the attention area based on user reactions to data items that were clustered into this area. Let  $E_p$  is the set of data items with a positive reaction flag,  $r_i$  is 1 if the  $i$ th item of  $L$  is in  $E_p$  and  $r_i$  is 0 otherwise.

$$s_p = NDCG(L, E_p) = Z_x \sum_{i=1}^x (2^{r_i} - 1) / \log_2(i + 1)$$

where  $Z_x$  is chosen so that an all-positive list has NDCG value of 1. In the above measurement, the reactions to more recent items have greater effect on the importance score. The above formula intends to capture the trend of the importance of the attention area perceived by the user. For example, if a user recently loses interest on a topic and begins to ignore relevant messages, the importance score of the topic will decrease quickly, even if the user had a lot of positive feedback on the topic in the past. In general, PAM is adaptive when it measures the importance of a user’s attention areas.

### 4.2 Labeling Attention Areas

To select a indicative keyword to label the attention area, the word should not only be important to the attention area but also be distinguishable from other attention areas. We employ a TF-IDF-like approach to compute a representation score for each word in an attention area and select the one with the highest score. More specifically, given a word  $w_i$ , its representation score with regards to an attention area is

$$rs(w_i) = f_w(w_i) \log(|F|/|F_{w_i}|)$$

where  $f_w(w_i)$  is the weight of  $w_i$  in the attention area,  $|F|$  is the total number of the user’s attention areas, and  $|F_{w_i}|$  is the number of the user’s attention areas that contain  $w_i$  as one of their top  $x$  (say  $x = 20$ ) keywords.

## 5 Generate Personalized Attention Report

To generate a concise report, PAM utilizes three facets, namely task, important contacts and time-critical, to categorize incoming messages and enable users to browse them. The “time-critical” is similar to the “Urgent” category in current email clients. The “important contacts” category is only



Figure 2: Example attention report by PAM

for messages that come from important contacts but do not belong to any particular task. Our system optionally allows user to give us input on those people. Keep in mind, when the messages from important contacts belong to a particular task, our system already automatically identifies those types of important contacts. Therefore the user input on important contacts in our system is very little if not none. While these two facets are easy to understand and similar to existing email client interfaces, the main interface extension to current email client interface that our work can enable is the task-based category. The task-based categories are for messages that are clustered into attention areas. Each attention area is associated with a task that the user is actively involved with. In this work, a task is not equivalent to a topic. A task may comprise of one or more related topics; A topic may be split into several tasks, for instance, depending on different time lines. See Figure 2 for a sample report.

A task-based category is labeled by a representative keyword that is automatically learned after clustering. We employ a two-level interface for the task category. The first level interface lists all the tasks that the new messages belong to. As one can imagine, it is possible that there is no new message on certain categories since last check of the messages. In that case, that category is not displayed. This allows users to quickly learn if there is any new update on each category and the corresponding tasks. The task-based categories are displayed in descending order based on the learned user interest level on each task. The second level interface displays representative and important messages in descending order within each task category. The non-representative and less important messages in each category are not displayed in the first screen and can be read when users explicitly click a button, such as "see more".

---

### Algorithm 1 Ranking Messages inside a Category

---

```

1: Rank( $\mathcal{F}, \mathcal{S}$ )
2:  $\mathcal{S} = \emptyset$ 
3: if  $\mathcal{F} == \emptyset$  then
4:   return  $\mathcal{S}$ 
5: loop
6:   select  $k \in \mathcal{F}$  such that  $k = \operatorname{argmax}_{i \in \mathcal{F}} B(\mathcal{F}, \mathcal{S}, k)$ 
7:    $\mathcal{F} \leftarrow \mathcal{F} - k$ 
8:    $\mathcal{S} \leftarrow \mathcal{S} + k$ 
9:   if  $\mathcal{F} == \emptyset$  then
10:    return  $\mathcal{S}$ 

```

---

## 5.1 Categorizing Messages into Task Categories

Given a list of messages, our system first matches those messages with attention areas in the user profile and then creates an entry for each attention area that has a matching message. All the matching messages for a task-based attention category belongs to that category. Again, some attention areas are empty and empty attention areas are not displayed.

Among remaining messages, PAM will check if a message comes from a person in the list of "Important Contacts". If such a message exists, there will be a category called "Important Contacts" in the attention report.

After excluding those messages from important contacts, for the remaining messages, PAM employs the same dynamic clustering process discussed earlier to generate new task-based categories. This is for special case when the new messages are not associated with any existing tasks and can create new attention areas. The user profile can be incrementally updated with new categories learned. The dynamic nature of our clustering algorithm makes it natural to perform incremental updates. It is also easy to re-calculate the importance levels of the tasks.

## 5.2 Prioritize Messages inside a Task Category

On the attention report, for each task category we want to show some important representative messages that belong to that category. There are two steps involved in this process. The main step for this is to determine the order of the new messages that the user should pay attention to in that task.

**Determining the Importance Order of the Messages Inside a Category** Algorithm 1 shows the pseudocode of the algorithm. Given a list of messages in  $F$  that are clustered into a task-based category from the previous organization step, we take a greedy approach to order messages and move them to  $S$  one by one. In every loop step between line 6 to 8, we always choose the message that can maximize the user benefits of knowing that message. The benefit is calculated by a function  $B$  that we will explain below. Once such a message is chosen, it is moved from  $F$  to  $S$ .

It is natural to understand that the benefit of knowing a message depends on multiple factors. Below shows some factors used in our work. The list of factors is expandable.

- The similarity between the topic of the message and the topic of the attention area. The closer the better.

- The similarity between the message and the messages that are chosen previously. If there is already a similar message chosen, it is less critical for the user to see this one.
- The closeness between the sender of the message and user's important people in this task. A user prefers to see messages coming from an important person for that task. Recall we measured weight for each person when we derive the attention areas to build user profile.
- The recency of the message. A user prefers to see messages that received most recently if possible.

Based on the above observation, we quantify the user benefits of knowing about message  $k$  when the user already knows all the messages in  $S$  in category  $C$  as follows:

$$B(F, S, k) = \text{sim}(k, C) - \text{sim}(k, S) + \text{sim}(k, PR)$$

$\text{Sim}(k, C)$  and  $\text{sim}(k, S)$  measure the similarity between message  $k$  and the attention area  $C$  and the messages already chosen in  $S$  respectively. For clustering, we discussed the normalized similarity measurement between two groups of data items. Each group is represented by a list of topical keywords and people involved as well as its time line. We apply the same similarity measurement here.

$\text{Sim}(k, PR)$  measures the closeness between the message  $k$  against user's preference. We assume a user prefers to see messages from an important person for a task. We also assume a user prefers to use more recent message if possible.

$$\text{sim}(k, PR) = \frac{(P_k - P_{min})}{(P_{max} - P_{min})} + \frac{(T_{now} - T_k)}{(T_{now} - T_{earliest})}$$

Let  $P_k$  be the importance weight of the sender of message  $k$ , and  $P_{min}$  and  $P_{max}$  be the least weight and the max weight of people in this attention area as indicated in user's profile. Let  $T_k$  be the time when the message arrived. We measure the normalized similarity between  $k$  and the user's preference on people and recency as follows, assuming time value goes bigger as time goes.

While we used the above factors in our experiments, we believe it is also possible to use other factors. For example, it is possible to consider factors that depends on the device on which we implement this system. On a smartphone, a user may prefer to read short messages over long messages. Therefore one may use the length as a user preference factor in the above calculation. As another example, if this system is implemented for news recommendation, a user may prefer to get updated with news happening on a particular location. In this case it is possible to use location as a user preference factor.

### Select Representative Messages in a Task Category

Once the order of the messages inside the task category is determined, our system will display the ordered message list from top down on a desktop machine. On small screen mobile devices, our system chooses to display a certain number of representative messages from the ordered message list from top down. The number of representative messages can be specified by the user. As a default value, our system determines the boundary of the representative message list

when the value of the above benefit function is below some threshold. Intuitively it means knowing any of the remaining messages bring little value to the user and therefore it is fine to delay paying attention to those messages until later. The users can always see all the messages in this category by clicking the "see more" button on mobile devices.

## 6 Evaluation

We have implemented an end-to-end prototype of our attention management solution and evaluated our system performance. In an ideal attention report: (1) all the messages categorized into a task category indeed belong to the intended task so that the user does not have to manually pick out relevant messages for a task, (2) the task is labeled reasonably well capturing what the task is mainly about, (3) the importance of the task is determined correctly, (4) inside each task category all the representative messages are indeed the important messages in that category that the user wants to know. We have evaluated the accuracy of our attention report along these four dimensions.

Moreover, we are interested in measuring how well our system handles the evolving set of tasks over time.

### 6.1 User Evaluation Methods

We conducted a user study on about 15 people in a research institute. The subjects were in different roles, including management, research, and engineering. Some are technical people, some are not. They were not part of this project team. We acquired users' data through a unified information gathering platform. The types of user data include email, calendar events, and activities on an enterprise collaboration system. The collaboration activities include shared files, Wiki posts, blogs, status update messages and so on. The collected data is a good mixture of messages as motivated in Figure 1. For example, the status update messages are very short and almost identical to tweets. For each user, we harvest data in an incremental fashion. We first harvest about two weeks' emails, calendar events, and all of his/her collaboration events/activities we can collect. The number of emails does not vary much among users. On average, the number of emails is 1300 and about 80 calendar events. The numbers of collaboration activities vary among users, ranging from around 20 to over 200. We call this the first data set. With this data set, our system learns user profile by creating attention areas and determining its label and importance levels to the user. This profile is denote as  $P1$ .

Without seeing our system-generated  $P1$ , the users were asked to organize the first data set into task categories by themselves. Due to the large number of messages in the first data set, all users are only willing to come up with the names/labels of the important tasks. We evaluate the accuracy of  $P1$  against the user-generated task lists.

The next day we ask users not read emails and other messages in the unified message system until the end of the day. We collect all of this day's messages. We call this the second data set. We create an attention report  $R2$  for the user based on this data set. At the end of the day we interview the user. Before we show the attention report for the day to the

user, we ask users to read the messages and record the order of their readings. The users are not required to read all the messages. They stop when they feel they have already had a good grasp on what is happening from this day’s messages. We also ask users to manually categorize the messages they have read into the different task categories. The messages they have read in each task category by this point become the representative message list for the category. Then we ask users to finish reading and categorizing all remaining messages. Note it is possible to generate new task categories that have never been seen from the previous two weeks’ data. The users are asked to mark the important tasks from all the current tasks. We compare our result  $R2$  with what the user manually comes up with to measure the accuracy of our attention report in terms of categorization accuracy, labeling accuracy and importance level accuracy as well as the accuracy on the representative list for each task category. We update the user profile with the new tasks generated in  $R2$ .

Lastly, the user can proceed to process this day’s messages including replying to emails, posting new blogs and so on. With user’s such actions our system updates the importance level of the tasks in the updated user profile.

We continue to collect new data for another day. We call this the third data set. We generate another attention report  $R3$  based on this third data set and repeat the same interview. Again  $R3$  may contain new tasks that were never seen before. Our system updates the user profile and also updates the importance level of the tasks in the updated user profile.

The number of messages in the second data set and third data set do not vary much among different users, averaging at 100 emails and 8 calendar events per day. The number of messages from other social collaboration tools for users varies from 2 to 10 per day.

**Questions and Metrics** We sort the attention areas in our system-generated user profile  $P$  and attention report  $R$ . Let  $A[1 : k]$  be the list of the  $k$  most important attention areas in a profile or an attention report. In one experiment, we used  $k = 10$ ; in another experiment we used  $k = 20$ . For both  $P1$  and  $R2, R3$ , we asked the following questions and used the following criteria to measure the overall quality of the profile and attention reports.

- For each attention area, we first check if the area is in the user-generated list. If yes we consider the area as coherent. If not in the list we ask the user to mark whether it represents a coherent real-world task for him/her. If it does, we also ask users if it is feasible to add this area into the task list the user generated. Recall that users felt difficulty to come up with the complete task list by themselves and sometimes are only willing to come up with the names/labels of the important tasks. We use the user-marked “coherent” tasks to augment the user-generated task list when the user agrees. An attention area is incoherent if it is a mixture of items belonging to different tasks or what it represents is unclear to the user. Let  $A_c[1 : k]$  be the set of attention areas that have been marked as “coherent”. The *coherence precision* of  $A[1 : k]$  is computed as  $|A_c[1 : k]|/k$ . With the augmented more complete task list that user

generates or agrees to, the users are allowed to adjust the list of the important tasks in that complete list.

- For each attention area, if it corresponds to a task in the user-generated list, we compare their labels. If the labels are different, we ask the user if they prefer to change our label to theirs. If the label needs to be changed, our label is considered incorrect. If the area is not in the user-generated list, we ask the user to state whether its label is *indicative* or not. In other words, whether the user is able to tell what a task is about by just looking at its label. The user may also propose a better word to label the area. We use the *label precision* of  $A[1 : k]$  to measure the accuracy of the labels our system selected.
- We also would like to evaluate how accurately we have categorized data into an attention area. We ask users to determine which items are categorized incorrectly. Given an attention area  $a_i \in A[1 : k]$ , the *categorization precision* of  $a_i$  is computed as  $|I_c(a_i)|/|I(a_i)|$ , where  $I(a_i)$  is the set of data items in  $a_i$  and  $I_c(a_i) \subseteq I(a_i)$  is the set of items that are classified correctly. The *categorization recall* of  $a_i$  is computed as  $|I_c(a_i)|/|I_i|$ , where  $I_i$  is the set of data items that should be classified into  $a_i$ . The categorization precision and recall are the average values of the attention areas in  $A[1 : k]$ .
- With the augmented user-generated task lists and the user-adjusted important task list, we measure our task ranking accuracy. The *ranking precision* of  $A[1 : k]$  is computed as  $|A_p[1 : k]|/|A_c[1 : k]|$ , where  $A_p[1 : k]$  is the set of attention areas in  $A[1 : k]$  with correct relative importance estimation. We say that PAM has estimated the relative importance of an attention field  $a_i$  correctly when one of these conditions holds: (1) the user marks  $a_i$  as “important”, and  $a_i$  is among the top 10 attention fields in the user’s profile; (2) the user marks  $a_i$  as “important”, and there does not exist any “unimportant” attention field that is ranked higher than  $a_i$ ; (3) the user marks  $a_i$  as “unimportant”, and the ranking of  $a_i$  is lower than 10; (4) the user marks  $a_i$  as “unimportant”, and there does not exist any “important” attention field that is ranked lower than  $a_i$ . The *ranking recall* is computed as  $|A_p[1 : k]|/|A_u[1 : k]|$ , where the  $A_u[1 : k]$  is the augmented task list.
- For each attention area, we compare our system-generated representative message list against the user-generated list. We use precision and recall to measure the accuracy for each area. The *representativeness precision and recall* of  $A[1 : k]$  are the average values over all the attention areas in  $A[1 : k]$ .

Our evaluation results are given in Figure 3. We list results for the top 10 and 20 attention areas. The values in Figure 3 are the average values over the 15 users we interviewed. As we can see, our system achieves pretty good accuracy across all dimensions we measured. As a note, we did not show the categorization accuracy because none of the participants singled out a particular message that is categorized incorrectly.

Top $k$	Profile	Coherent Prec.	Label Prec.
10	$P1$	0.88	0.89
	$R2$	0.87	0.87
	$R3$	0.89	0.90
20	$P1$	0.79	0.89
	$R2$	0.86	0.86
	$R3$	0.83	0.88

(a) Coherent, Label Precisions

Top $k$	Profile	Rank Precision	Rank Recall	$F_1$
10	$P1$	0.74	0.86	0.79
	$R2$	0.75	0.88	0.81
	$R3$	0.74	0.87	0.80
20	$P1$	0.81	0.97	0.88
	$R2$	0.82	0.96	0.88
	$R3$	0.81	0.96	0.88

(b) Ranking precision, recall and  $F_1$ 

Top $k$	Profile	Repres. Prec.	Repres. Recall	$F_1$
10	$P1$	0.83	0.90	0.86
	$R2$	0.87	0.92	0.89
	$R3$	0.85	0.91	0.88
20	$P1$	0.82	0.90	0.86
	$R2$	0.86	0.94	0.90
	$R3$	0.85	0.93	0.89

(c) Representativeness precision, recall and  $F_1$ 

Figure 3: Evaluation results

## 7 Acknowledgments

The author likes to thank Qihua Wang’s help in this work.

## 8 Conclusion

We have described a novel intelligent attention management solution called PAM that employs analytical techniques to mitigate information overload for users. Given a list of unread messages, PAM generates a concise attention report by clustering, labeling, and prioritizing those messages. The attention report enables users to selectively read the most important messages first. This is exceedingly useful for mobile users who has a small screen and also very limited time in processing messages. Our evaluation results from user study on real-world data demonstrated the effectiveness of PAM. As future work we want to conduct extensive user studies on the usability of our system. We also like to automatically learn user preferences from message click-through such as to reduce the need of user explicit input as well as improve the accuracy of our system. It is also very desirable to make our system adaptive to the learned user preferences.

## References

Bernstein, M. S.; Suh, B.; Hong, L.; Chen, J.; Kairam, S.; and Chi, E. H. 2010. Eddi: interactive topic-based browsing of social status streams. In *Proceedings of the 23rd annual ACM symposium on User interface software and technology*, UIST ’10, 303–312. New York, NY, USA: ACM.

Blei, D. M.; Ng, A. Y.; and Jordan, M. I. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.* 3:993–1022.

Chirita, P.-A.; Diederich, J.; and Nejdil, W. 2005. Mail-rank: using ranking for spam detection. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, CIKM ’05, 373–380. New York, NY, USA: ACM.

Dredze, M.; Lau, T.; and Kushmerick, N. 2006. Automatically classifying emails into activities. In *Proceedings of the 11th international conference on Intelligent user interfaces*, IUI ’06, 70–77. New York, NY, USA: ACM.

Dumais, S.; Cutrell, E.; and Chen, H. 2001. Optimizing search by showing results in context. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI ’01, 277–284. New York, NY, USA: ACM.

Faulring, A.; Myers, B.; Mohnkern, K.; Schmerl, B.; Steinfeld, A.; Zimmerman, J.; Smailagic, A.; Hansen, J.; and Siewiorek, D. 2010. Agent-assisted task management that reduces email overload. In *Proceeding of the 14th international conference on Intelligent user interfaces*, IUI ’10, 61–70. New York, NY, USA: ACM.

Fisher, D.; Brush, A. J.; Gleave, E.; and Smith, M. A. 2006. Revisiting whittaker & sidner’s ”email overload” ten years later. In *Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*, CSCW ’06, 309–312. New York, NY, USA: ACM.

Freed, M.; Carbonell, J.; Gordon, G.; Hayes, J.; Myers, B.; Siewiorek, D.; Smith, S.; Steinfeld, A.; and Tomasic, A. 2008. Radar: a personal assistant that learns to reduce email overload. In *Proceedings of the 23rd national conference on Artificial intelligence - Volume 3*, 1287–1293. AAAI Press.

Horvitz, E.; Jacobs, A.; and Hovel, D. 1999. Attention-sensitive alerting. In *Proceeding of UAI’ 99: Conference on Uncertainty and Artificial Intelligence*, 305–313.

Käki, M. 2005. Findex: search result categories help users when document ranking fails. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI ’05, 131–140. New York, NY, USA: ACM.

Mock, K. 2001. An experimental framework for email categorization and management. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR ’01, 392–393. New York, NY, USA: ACM.

Whittaker, S., and Sidner, C. 1996. Email overload: exploring personal information management of email. In *Proceedings of the SIGCHI conference on Human factors in computing systems: common ground*, CHI ’96, 276–283. New York, NY, USA: ACM.

Xiang, Y. 2009. Managing email overload with an automatic nonparametric clustering system. *J. Supercomput.* 48:227–242.

Yoo, S.; Yang, Y.; Lin, F.; and Moon, I.-C. 2009. Mining social networks for personalized email prioritization. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD ’09, 967–976. New York, NY, USA: ACM.