Which to View: Personalized Prioritization for Broadcast Emails*

Beidou Wang[♯†], Martin Ester[†], Jiajun Bu[♯], Yu Zhu[‡], Ziyu Guan[⋆], Deng Cai[‡],
[♯]Zhejiang Provincial Key Laboratory of Service Robot, College of Computer Science, Zhejiang University, China
[†] School of Computing Science, Simon Fraser University, Canada
[⋆]College of Information and Technology, Northwest University of China
[‡]State Key Lab of CAD&CG, College of Computer Science, Zhejiang University, China
[†]{beidouw,ester}@sfu.ca, [♯‡]{bjj,zhuyu_cad,dcai}@zju.edu.cn,[⋆]{ziyuguan}@nwu.edu.cn

ABSTRACT

Email is one of the most important communication tools today, but email overload resulting from the large number of unimportant or irrelevant emails is causing trillion-level economy loss every year. Thus personalized email prioritization algorithms are of urgent need. Despite lots of previous effort on this topic, broadcast email, an important type of email, is overlooked in previous literature. Broadcast emails are significantly different from normal emails, introducing both new challenges and opportunities. On one hand, lack of real senders and limited user interactions invalidate the key features exploited by traditional email prioritization algorithms; on the other hand, thousands of receivers for one broadcast email bring us the opportunity to predict importance through collaborative filtering. However, broadcast emails face a severe cold-start problem which hinders the direct application of collaborative filtering. In this paper, we propose the first framework for broadcast email prioritization by designing a novel active learning model that considers the collaborative filtering, implicit feedback and time sensitive responsiveness features of broadcast emails. Our method is thoroughly evaluated on a large scale real world industrial dataset from Samsung Electronics. Our method is proved highly effective and outperforms state-of-the-art personalized email prioritization methods.

Categories and Subject Descriptors

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

General Terms

Theory

*This work was inspired by the author's internship at Samsung Research Canada

WWW 2016, April 11–15, 2016, Montréal, Québec, Canada.
ACM 978-1-4503-4143-1/16/04.
http://dx.doi.org/10.1145/2872427.2883049 .

Figure 1: Gmail Uses a Yellow Icon to Mark the Important Emails

Keywords

Email Prioritization, Active Learning, Recommendation System

1. INTRODUCTION

With nearly 200 billion emails sent and received per day, email is undoubtedly one of the most prevalent personal and business communication tools[29]. However, along with the great benefits come significant drawbacks. According to previous research, 58% of emails are unimportant or irrelevant, and a person on average spends nearly 380 hours every year to handle those emails, which causes trillion-level economy loss in productivity[4][16]. This phenomenon is called email overload, and there is an urgent need to develop a system that can automatically learn the personal priorities of the emails to mitigate the problem.

Different from spam filtering which has been widely explored in the literature[24][25], *personalized email prioritization* aims at making a personalized prediction of the importance label of non-spam emails. Many efforts have been done in both industry and academia to solve this problem[39][36]. For instance, Google has proposed an email importance prediction algorithm for Gmail[1] and it has been used in the Gmail Priority Inbox, in which every important email is marked with a yellow icon next to the sender's name (Figure 1).

However, broadcast email, one important type of email with interesting and challenging characteristics, has been overlooked in previous personalized email prioritization systems. A *broadcast email* is an email message that are sent to a group of receivers, usually by organizations, companies and web services. The size of the group is typically large, commonly containing thousands or even millions of receivers. A typical email user may be members of dozens of broadcast mailing lists. For instance, as a student, we are in the broadcast mailing list of graduate students; as an employee we are in the company's developer department list; as a con-

sumer we are in the promo lists of various products. Even though there are important and related broadcast emails, a large portion of them are irrelevant and unimportant and one can easily get swamped by them. Moreover, it makes us more likely to miss the really important broadcast emails since we are used to neglecting them. Thus, personalized email prioritization is even more important for broadcast emails. However, broadcast emails are significantly different from normal emails. The following challenging characteristics of broadcast emails could fail traditional personalized email prioritization methods:

The Same Sender One of the most indicative features from previous works [1][39] for personalized email prioritization is the social feature based on the interactions between the sender and the receiver. For instance, if a high percentage of a sender's emails were read by the receiver, we can deem the sender important and predict his following emails are also important to the receiver. However, for a broadcast mailing list, there is usually only one sender (e.g. mailing list admin) and a receiver may get hundreds of different emails from the same broadcast sender.

Limited Types of Interaction Traditional methods often exploit a user's interactions with emails for importance prediction. Compared to normal emails, nevertheless, the types of interaction with broadcast emails are limited. For instance, we usually will not reply, forward or cc a broadcast email; we would not manually label an irrelevant mail from a broadcast mailing list as spam either, since we still wish to receive other mails from the mailing list. The common user action on broadcast emails is viewing.

Hence, many key features of traditional methods cannot be extracted for broadcast emails, which significantly deteriorate their performance. Despite these new challenges, broadcast emails also bring us new opportunities. The most prominent one is that each broadcast email is sent to thousands of users and other users' responses (view or not) can be very helpful in predicting a target user's preference. In other words, we could generate priority predictions by *collaborative filtering*: for a user, if other users with similar interests have considered the email important (i.e. viewed it), he should be very likely to also consider it as an important email. In this paper, we propose the first personalized email prioritization framework for broadcast emails, in which we exploit collaborative filtering features by considering other users' responses to broadcast emails. However, there exists one key challenge. Each email waiting for priority prediction is completely *cold*. That's to say no view-email action has been observed, since the email has not yet been sent to any users, which makes it impossible to exploit the collaborative filtering features directly.

We propose a novel active learning framework to solve the cold-start problem. The intuition is simple. For a new email, we first send it to a small portion of the users in the mailing list (e.g. 5%) without priority labels and wait for a short period of time (e.g. half an hour) to collect their feedback (whether the user has read the email). Then based on these users' feedback, we predict the priority for the remaining majority of users. Our personalized email prioritization problem can thus be naturally divided into two sub problems. First, how to sample the small portion of users whose feedback can help us the most in determining the email priority for the remaining users. Second, once user feedback are gathered, how to use them to accurately predict personalized priority for the remaining users.

Our problem can be considered as a type of active learning for recommendation. However, due to the unique characteristics of the broadcast email prioritization task, several challenges exist which are different from those for the traditional active learning recommendation methods.

Implicit Feedback To the best of our knowledge, the literature on active learning in recommender systems focuses on explicit feedback [22][20][35][21][15]. For example, works on the MovieLens and Netflix datasets deal with 1-5 star user ratings. The underlying assumption is that once we query a user about an item and get his feedback, we can know his preference on the item based on his rating. While in our task, we only consider users' view-email actions, which are one-class implicit feedback. If the user viewed the email (positive feedback) we infer the email is important. However, if the user did not view the email (negative feedback), there is no way we can infer the importance of the email since we could attribute a user not reading an email to a lack of interest or a lack of awareness of the email.

Timely Response In the active learning recommendation literature, there exists an underlying assumption that users queried for feedback always provide feedback in time. However, this is not the case in our task. When we send the email to a small portion of the subscribers for feedback, we can only wait for a short period of time for the responses, due to the real-time nature of emails. Hence, our problem requires us to sample users who can provide responsive feedback in time.

Completely Cold Item In previous works on active learning for recommendation [12, 17, 31], the methods require that even for cold start items, there are a small number of initial ratings. However, in our problem, every email waiting for prioritization is completely cold with zero user response.

Fairness in User Querying For every email requiring prioritization, we need to query some informative users for feedback. The choice of informative users need to be fair. That is to say we cannot always pick the same set of users for feedback querying, because it will result in them losing the chance of benefiting from the service of email prioritization and may even generate user frustration.

To cope with the above mentioned challenges, we propose a novel active learning framework, in which we sample a small set of informative users considering both the preference of a user and his tendency of giving responsive feedback, by exploiting features including text and attributes of emails and users' email-view interactions. After gathering the feedback, we use a weighted regularized matrix factorization method specifically designed for implicit feedback to learn the preference scores for the remaining users and use the scores as the key feature to predict the final priority of the email.

Since there is no publicly available dataset that contains personal importance judgments by real users for broadcast emails due to the privacy concern [39], we collect an industrial dataset from Samsung Electronics. The dataset contains thousands of broadcast emails from one of Samsung Electronics's company broadcast mailing lists with thousands of Samsung employees as subscribers. These employees are from all over the world and with diverse demographic features. We collect both the text features (e.g. email title, content) and attributes features (e.g. sender, receiver, timestamp) for the broadcast emails. We also collect user's view logs of these emails in a 9-month time window. We conduct extensive experiments and demonstrate that our method outperforms all the baseline algorithms in terms of prediction accuracy.

Our main contributions are as follows:

1. We present the first in-depth discussion of personalized prioritization for broadcast emails and propose an active learning based framework to solve the problem. In particular, we exploit the collaborative filtering features in email prioritization.

2. We propose a novel active learning model that can handle one class implicit feedback, and considers users' time-sensitive responsiveness for active learning based recommendation.

3. Our method is thoroughly evaluated on a large scale real industrial dataset, and is demonstrated to be highly effective compared against a large number of baselines.

2. RELATED WORK

2.1 Active Learning for Recommendation

In recommendation systems, active learning methods have been proposed to acquire those ratings from users that will best help in predicting the ratings for the unknown user and item pairs. Active learning methods for recommendation can be classified into three big categories, attention-based methods, uncertainty reduction methods and error reduction methods[37].

Attention-based strategies are simple and easy to implement, and are usually used as the initial attempts to solve the cold start problem. The Popularity strategy [9, 10] selects items that have received the highest number of ratings and thus users are more likely to be able to rate them. The Coverage strategy [9] selects items that are highly co-rated with other items by users and thus are prone to improve the prediction accuracy for the other items. However, these methods are not personalized and do not consider the fairness requirement listed in the Introduction.

Uncertainty reduction aims at reducing the uncertainty of rating estimates, decision boundaries and model parameters. For uncertainty reduction of rating estimates, some researchers [23] propose to select training points in a local (greedy) manner by rating items with high uncertainty. The uncertainty of an item's rating can be measured by its variance, entropy [9, 32] or confidence interval [30]. However, labeling uncertain items does not necessarily reduce the rating uncertainty for the other items. [31] proposes a method to solve this problem. Decision Boundary-based approaches select training points closest to the decision boundary, in order to obtain a more accurate decision boundary [5]. In Model Uncertainty-based methods, training points are selected so as to reduce the uncertainty about the model's parameters. It assumes that the accuracy of output values will improve when we improve the accuracy of the model's parameters [12, 17]. There are also combined strategies [32, 26] considering both popularity and uncertainty, which can be done in various ways to achieve different objectives. However, these methods either require a small amount of initial ratings which is not available for completely cold items/users or are non-personalized.

Error reduction refers to reducing the predictive error by 1) Greedy Extent [9, 10], namely optimizing the performance measure (e.g. minimizing RMSE) on the training set or 2) utilizing the relation between the predictive error on the testing set and other factors. Output Change-based approaches [33] assume that most changes in the output estimates lead to more accurate estimates on the testing set. Thus they favor points that are likely to cause many estimates to change. Parameter Change-based methods [34] choose points that will change the model's parameters the most. Many works [2, 19] propose sampling methods aiming to reduce the variance of the model's parameters estimates in terms of different measures. These methods all require a small amount of initial ratings and cannot handle the completely cold start problem. The Personality-Based Binary Prediction method [6] tries to utilize attributes information to handle the completely cold start problem and transforms the item selection problem into a common matrix factorization problem. However, all the above mentioned methods are designed for the explicit rating prediction problem. The challenges of one-class implicit feedback in our task (e.g. the uncertainty in negative feedback and the informativeness difference between positive and negative feedback) has not been discussed and cannot be effectively handled by previous methods.

Some works also consider the trade-off of the exploration-exploitation problem [32, 30]. Most recommendation methods tend to focus on exploitation, that is, provide accurate predictions to the user. However, recommending items that will expand users' interests (exploration) is of great importance as well, especially for newly sign-up users or long-term users. The exploration-exploitation criteria also holds true in the active learning phase of our task. Many of the promising solutions come from the study of the multi-armed bandit problem [7]. Random sampling, ϵ-greedy method, the upper confidence bound (UCB) algorithms and Thompson sampling have been explored in solving the problem [7]. However, the setting of these solutions requires that several arms (sampling strategies) perform parallelly and cannot be directly applied to our task.

2.2 Prioritization for Emails

Email prioritization focuses on making a personalized prediction of the importance label of non-spam emails [39, 18, 13].

Douglas et. al [1] propose a simple linear logistic regression model to do prioritization for gmail, in which the final prediction is the sum of the global model and the user model log odds. Four categories of features are considered in the model, including social features, content features, thread features and label features. In [39], authors use personal social networks to capture user groups and to obtain rich features that represent the social roles from the viewpoint of

Table 1: Email Features

Feature Category	Features
Email Body	Email title, content of email body
Email Header	Sender ID (only one sender, email-admin), receiver ID, time stamp of email
User Attribute	Receiver ID, receiver country, receiver timezone

a particular user. They also developed a semi-supervised (transductive) learning algorithm that propagates importance labels from training examples to test examples through message and user nodes in a personal email network. In [36], authors summarized multiple classification and semi-supervised clustering methods on spam detection and email categorization tasks. A social clustering approach is proposed in [18] to predict the email prioritization based on the relations between its sender and induced social clusters. [27] defines metrics for measuring the social importance of users based on the email elements: from, to and cc, and user actions of replying and reading, which can potentially be used for measuring email prioritization. Horvitz et al. [13] regard the email prioritization prediction task as a classification problem. They use Support Vector Machines to predict that whether the utility of newly arrived emails is high or low. However, due to the previously described characteristics of broadcast emails, i.e. The Same Sender and Limited Types of Interaction, these traditional methods designed for normal emails cannot be effectively applied to our broadcast email prioritization task.

3. PROBLEM DEFINITION

The task of this paper is personalized prioritization for broadcast emails. That is to say we want to predict whether a broadcast email is important or not for a given user. The problem can be divided into two *sub problems*. First, sample a small portion of users whose feedback can best help us in predicting the email priority for the remaining users. Second, predict the priority for the remaining users based on the feedback collected from the sampled users. We define the problem formally as follows.

For user set $\mathbf{U}$ and email set $\mathbf{E}$, we define a binary email importance label set $\mathbf{I}$ based on users' email-view interactions. That's to say, for user $u \in U$ and email $e \in \mathbf{E}$,

$$I_{u,e} = \begin{cases} 1 & \text{if } u \text{ has viewed } e \\ 0 & \text{if } u \text{ hasn't viewed } e \end{cases} \quad (1)$$

For each email e, we record its text and attribute-based features, e.g. title, content, sender and receiver of the email. For each user u, we record user attribute features, e.g. country and timezone. Details of the features are provided in Table 1. Given a new email e_{new}, we define the two sub problems mentioned above as:

Sampling Users for Feedback Given $\mathbf{U}$, $\mathbf{E}$, $\mathbf{I}$, e_{new} and time interval $T_{feedback}$ in which we collect users' feedback, select the subset $\mathbf{S}$ of k users from $\mathbf{U}$ whose feedback in $T_{feedback}$ maximizes the prediction accuracy of $\mathbf{I}_{\mathbf{U}-\mathbf{S},e_{new}}$.

Prediction For Remaining Users Given $\mathbf{U}$, $\mathbf{E}$, $\mathbf{I}$, e_{new}, $\mathbf{I}_{\mathbf{S},e_{new}}$, predict the importance label set $\mathbf{I}_{\mathbf{U}-\mathbf{S},e_{new}}$.

4. THE FRAMEWORK

We propose an active learning framework for the broadcast email prioritization problem. There are two parts of our framework, sampling informative users for feedback and making priority prediction for the remaining users based on the collected feedback, which will be described in details in the following subsections.

4.1 Sampling Users for Feedback

To the best of our knowledge, none of the previous works using active learning for recommendation focuses on how to handle one-class implicit feedback data and most of them, especially the personalized active learning for recommendation methods require at least a small portion of initial ratings for each cold start user/item[1][39][36]. None of the previous works considers the time cost of obtaining users' feedback and none of them considers the fairness issue when sampling users for feedback. However, all of the above mentioned challenges are very important in sampling users for feedback for our task and are carefully handled in our work. Next, we first introduce important criteria of sampling users considered in our work and then discuss our sampling strategy in detail.

4.1.1 Sampling Positive Feedback

To make our method general, we only consider email-view interaction for our broadcast email prioritization task. There are two kinds of feedback we can receive. First, *positive feedback* which means the user has viewed the email, is a relatively clear evidence that this email is important. Second, *negative feedback* which means the user fails to view the email in time, is a mixture of cases where the user is unaware of the email or the user thinks the email is unimportant. Our proposed sampling method aims to sample more positive feedback from the users for the following reasons.

More Informative Positive feedback give us clearer and more confident information of users' preference to the email and decreases the overall uncertainty of the predicted email importance.

Data Sparsity The collected feedback will be used in a matrix factorization framework to predict the email priority for other users. Each new email waiting for prioritization is completely cold and we can only sample a small portion of users for feedback. Moreover, only a small portion of them will give positive feedback in time. Thus, the positive feedback data for the new email can be very sparse if we do not employ a sampling strategy favoring positive feedback.

It is worth noting that collecting too much positive feedback may introduce bias to the system, which will have a negative impact on priority prediction. We will discuss in section 4.2 about how to cope with this issue. There are two factors considered in our work in order to sample more positive feedback: users' preference and users' responsiveness, which will be explained in details as follows.

4.1.2 Predicting Users' Preference

The first way to get more positive feedback for an email is to predict users' preference to the email and sample users who are predicted to be interested in it. However, since

the email waiting for prioritization is completely cold with zero email-view interaction, we have to rely on additional information. Fortunately, the text features of emails provide us a natural way to link the new email to old emails. We use a hybrid recommendation algorithm to predict a user's preference towards the new email by combining the ideas of content based recommendation and item based collaborative filtering.

The text features of an email e are represented as $< et, eb >$, where et and eb correspond to the term vectors for the title and body of e and each dimension of a term vector corresponds to the tf-idf value of a term. Similar to [38], stopword filtering, stemming and part of speech tagging are performed on the text features and only nouns are kept in the vocabulary.

We define the similarity between 2 emails e_i and e_j as the weighted average of the cosine similarities of their titles and bodies.

$$sim(e_i, e_j) = cos(et_i, et_j) + \alpha cos(eb_i, eb_j) \tag{2}$$

α is a constant weight and is learned by cross validation. For a new email e_{new} requiring prioritization, we first find the top k similar emails $\mathbf{E}_{sim}$ based on the similarity defined above. Then we predict user u_j's preference on e_{new} using an item based collaborative filtering idea by aggregating u_j's responses on $\mathbf{E}_{sim}$.

$$Score_{e_{new},u_j} = \frac{\sum_{e_i \in \mathbf{E}_{sim}} I_{u_j,e_i} sim(e_{new}, e_i)}{\sum_{e_i \in \mathbf{E}_{sim}} sim(e_{new}, e_i)} \tag{3}$$

$Score_{e_{new},u_j}$ is used later in the user sampling phase. It is worth noting that other methods like the method of [8] can also be used in our framework as long as we adapt it to estimate users' preference on completely cold items with the help of text features. We stick to the above described approach for simplicity and efficiency.

4.1.3 Predicting Users' Responsiveness

Different from previous works in active learning, the time cost of waiting for users' feedback cannot be ignored in our work since we can only afford to wait for a short time period for a user to respond. Thus, another crucial way to increase positive feedback is to sample responsive users who return their feedback in time. There are two factors that need to be considered to predict whether a user is responsive or not.

Users' Activity Time Users are only active (available to check emails) in a certain time window. For instance, a user usually only checks broadcast emails from his company during working hours and some users may start working early and leave early while others may prefer the opposite. Moreover, users are located in different countries and different timezones with various local work routines. It is important to generate personal activity time probability model and only query users who are active.

Email Checking Frequency Different users have different email checking habits. Some users may only check their inbox twice a day while others respond to the email in real time with the help of push notifications. To sample more responsive users, we always prefer users who check their inbox frequently.

We estimate the active time and email checking frequency based on the timestamps of a user's previous view-email behavior. All the timestamps are converted to the corresponding local time according to the timezone feature from user attributes. We define user u's temporal active profile as $D(u) =< v_{t_1}(u), v_{t_2}(u), ..., v_{t_{24}}(u) >$, where $v_t(u)$ is the number of days in which u was active in time interval t. Each day is divided into 24 time intervals. For example, t_1 is the time interval from 0:00 to 0:59. We regard u to be active in interval t in a day if at least one of u's view-email interactions is observed in interval t on that day.

Since for each user the view-email interactions can be very sparse, when generating personal active time probability model, we also rely on the view-email interactions of other users from the same country. We define the user set from country j as U_j. For a user u_i coming from country j, we define u_i's probability of being active at time interval t as

$$P_t(u_i) = \frac{\sum_{e \in E} I_{u_i,e}}{1 + \gamma \sum_{e \in E} I_{u_i,e}} \frac{v_t(u_i)}{ob(u_i)} + \frac{1}{1 + \gamma \sum_{e \in E} I_{u_i,e}} \frac{1}{|U_j|} \sum_{u \in U_j} \frac{v_t(u)}{ob(u)} \tag{4}$$

where $ob(u_i)$ refers to the total number of observation days for user u_i and $ob(u_i) = min($number of days u_i is registered , number of days of the experiment observation $)$. γ is a constant weight which can be learned by cross validation. The number of days of the experiment observation for our work is 270 (9 months). $\sum_{e \in E} I_{u_i,e}$ is the total number of u_i's view-email interactions. The intuition behind Equation (4) is that if u_i has few view-email interactions, the estimation of active probability for time interval t relies more on the average active probability in t of other users from the same country. As the number of view-email interactions increases, u_i's own interactive data will gradually become dominant in the probability estimation. $P_t(u_i)$ is used later in the user sampling.

As mentioned earlier, a user's email checking frequency also matters in giving a responsive feedback. We define the one-hour time window after one view-email interaction of a user as an *email checking session* and all the following view-email interactions within the one-hour time window belong to the same session. Denoting $\{\text{session}(u)\}$ as the set of all email checking sessions of user u, we define email checking frequency for u as:

$$frequency(u) = \frac{|\{\text{session}(u)\}|}{\text{ob}(u) + \zeta} \tag{5}$$

in which ζ is a constant used for smoothing and can be determined by cross validation.

4.1.4 Sampling Strategy

As discussed above, for a user u and a new email e_{new} sent at his local time t, the probability of u providing a positive feedback to e_{new} within the time limit is related with u's preference towards e_{new} ($score_{e_{new},u}$), u's probability of being active at t ($P_t(u)$) and u's email checking frequency ($frequency(u)$). We define the probability of u giving a positive feedback to e_{new} at t within the time limit as:

$$P(u, e_{new}, t) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 score_{e_{new},u} + \beta_2 P_t(u) + \beta_3 frequency(u))}} \tag{6}$$

It is a logistic regression model considering the above mentioned 3 factors. $\beta = \{\beta_0, ..., \beta_3\}$ is the model parameter which is trained based on the validation set.

For all users, we sort them in descending order of $P(u, e_{new}, t)$. We then sample the top k users to form the sampled user set $\mathbf{S}$.

4.2 Prediction for the Remaining Users

Since we tend to sample responsive positive feedback, bias could be introduced when we make predictions for the remaining users. In this section, we first discuss about how we use the weighted low-rank approximation technique to eliminate the bias and then propose the classification model for the final priority label prediction.

4.2.1 Weighted Low-rank Approximation

After receiving the feedback from the sampled users, we propose to use a matrix factorization based method to predict the preference for the remaining users. Since the feedback are one-class implicit and our sampling method introduces bias towards positive feedback, a weighted low-rank approximation method is developed to handle the implicit data and correct the bias.

The intuition behind the proposed method is to punish (add weight to) unexpected sampled feedback. That is to say, during the training phase, for all the feedback gathered by querying sampled users, we will punish negative feedback which we predict to have a high responsive positive probability in the sampling phase and positive feedback which we predict to have a low responsive positive probability in the sampling phase, by adding additional weights on corresponding training data points. For all the other training points, we will assign high weights to positive feedback and lower weights to negative feedback as described in [28][14].

Given the expanded importance label set $\mathbf{I}' = \{\mathbf{I}, \mathbf{I_S}, e_{new}\}$, our objective is to minimize the loss function

$$\mathcal{L}(\mathbf{P}, \mathbf{Q}) = \sum_{ij} W_{ij}(I'_{ij} - \mathbf{P}_{i\cdot}\mathbf{Q}^T_{j\cdot}) + \lambda(||\mathbf{P}||^2_F + ||\mathbf{Q}||^2_F) \quad (7)$$

in which $\mathbf{P} \in \mathcal{R}^{m\times d}$ and $\mathbf{Q} \in \mathcal{R}^{(n+1)\times d}$ stand for the latent vectors for $\mathbf{U}$ and $\{\mathbf{E}, e_{new}\}$. W_{ij} is a non-negative weight for u_i and e_j. Different from all the previous active learning works, we exploit the responsive positive feedback probabilities predicted in the user sampling phase of active learning in the weighting scheme to eliminate the bias from sampling. The weighting scheme of non-negative weight matrix $\mathbf{W}$ is summarized in Table 2. We set m equals to 1 and δ equals to 0.2 in the experiment.

Alternating Least Squares (ALS) is used to solve our optimization problem by fixing $\mathbf{P}$ and $\mathbf{Q}$ alternatively while optimizing the unfixed parameter.

When fixing $\mathbf{Q}$ and solving $\frac{\partial \mathcal{L}(\mathbf{P},\mathbf{Q})}{\partial \mathbf{P}_{i\cdot}}$

$$\mathbf{P}_{i\cdot} = \mathbf{I'_i}\widetilde{\mathbf{W}_{i\cdot}}\mathbf{Q}(\mathbf{Q}^T\widetilde{\mathbf{W}_{i\cdot}}\mathbf{Q} + \lambda(\sum_j W_{ij}\mathbf{ID}))^{-1} \quad (8)$$

where $\widetilde{\mathbf{W}_{i\cdot}} \in \mathcal{R}^{(n+1)\times(n+1)}$ is a diagonal matrix with the elements of $\mathbf{W}_{i\cdot}$ on the diagonal and $\mathbf{ID} \in \mathcal{R}^{d\times d}$ is an identity matrix.

Similarly, when fixing $\mathbf{P}$ and solving $\frac{\partial \mathcal{L}(\mathbf{P},\mathbf{Q})}{\partial \mathbf{Q}_{j\cdot}}$

$$\mathbf{Q}_{j\cdot} = \mathbf{I'_j}\widetilde{\mathbf{W}_{\cdot j}}\mathbf{P}(\mathbf{P}^T\widetilde{\mathbf{W}_{\cdot j}}\mathbf{P} + \lambda(\sum_i W_{ij}\mathbf{ID}))^{-1} \quad (9)$$

Table 2: Weighting Schemes

Feedback Type	Weighting Scheme
Positive Sampled Feedback	$1 + (m - P_{rp}(u_i, e_{new}, t_j))$
Negative Sampled Feedback	$P_{rp}(u_i, e_{new}, t_j)$
Other Positive Feedback	1
Other Negative Feedback	δ

where $\widetilde{\mathbf{W}_{\cdot j}} \in \mathcal{R}^{(m)\times(m)}$ is a diagonal matrix with the elements of $\mathbf{W_j}$ on the diagonal. Details of using ALS to solve matrix factorization problems is not the concern of this paper and can be viewed in [28].

For each remaining user $u_i \in (\mathbf{U} - \mathbf{S})$, we can predict his preference to e_{new} as

$$y_{i,e_{new}} = \mathbf{P}_i\mathbf{Q}^T_{e_{new}} \quad (10)$$

4.2.2 Feedback-sensitive Classification

Once $y_{i,e_{new}}$ is estimated for all the remaining users, we can combine it with any additional features proposed by previous methods (e.g. content feature and label feature) and put them in a classification model (e.g. a logistic regression model as proposed in [1]) to predict the email priority labels for the remaining users.

In this work, to make it simple, we use $y_{i,e_{new}}$ as the only feature considered in priority classification. We devise a classification method that can make a feedback sensitive classification. The intuition is that for each email a certain percentage of users will consider it as important, but the percentage varies among different emails since they are with different topics, written quality etc. We can infer the percentage of important emails by the percentage of positive feedback in the sampling phase. We define the threshold for email e_{new} as

$$H(e_{new}) = \frac{\theta pos(e_{new}) + pos(\mathbf{E})}{\theta k + mn} \quad (11)$$

$pos(e_{new})$ is the total number of positive feedback from the k queried users. $pos(\mathbf{E})$ is the total number of positive responses from all the m users for all the n previous emails. θ is a constant to balance the global percentage of positive responses with e_{new}-specific percentage of positive responses estimated from the sampled feedback and can be inferred by cross validation. For the top $H(e_{new})$ percent of users according to $y_{i,e_{new}}$, we predict e_{new} as important while for others as unimportant.

5. EXPERIMENTS

5.1 Dataset

We collected emails, view logs of emails and user information from a large business mailing list for employees within Samsung Electronics. Employees from all around the world receive emails with various topics, like win notices of deals, meeting agendas of customers, business objectives, news and technical issues. The dataset contains 6291 broadcasting emails sent to 2805 Samsung employees, generating 398,343 view records. For each email, we collected both text data like titles, contents and attributes like receiver, timestamp, timezone etc. All the emails have the same sender attribute

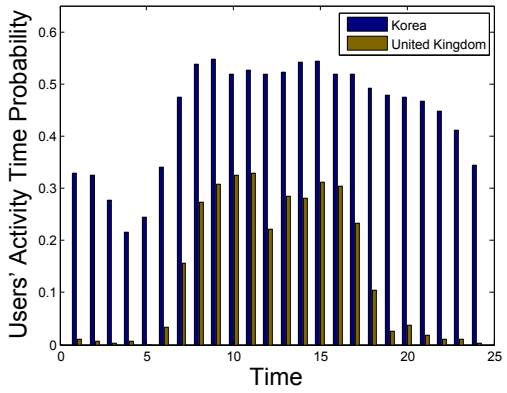

Figure 2: Activity Time Probability of Users from Korea and United Kingdom

(email-admin). We split the data set into training set (containing 5632 emails and their view logs) and testing set (659 emails and their view logs) based on a certain time point. Since we only have users' implicit email viewing action data, we assume an email is important to a user if the user has viewed it. It is worth noting that the data set is relatively small with only one sender and contains only view-email behavior. Accuracy could be better if we can incorporate information like deletions of emails, flagging emails as important and skipping an email. However, due to the privacy concerns, there is no public dataset containing importance judgments by real users for broadcast emails [39] and this is the best dataset we can get.

All user data was analyzed and stored in accordance with Samsung's privacy policy. Only the view logs of the authorized broadcast emails were extracted and all the users are Samsung Employees. The dataset was completely anonymized by mapping user ids and email ids to integer indices before any analysis. All the features extracted from messages were deleted after training.

5.2 Data Pre-processing and Analysis

The users in our dataset are located in 67 countries with widely varied timezones. To handle the timezone variance, the timestamps of users' view log were first converted to their local time. Then as mentioned in section 4, we calculated the active probabilities in all time intervals both for each user and for all the users from the same country, with the assumption that users from the same country share similar work routines. Based on our dataset, we noticed that users from different countries are with different activity time distributions due to working culture differences. For example, as shown in Figure 2, most users from United Kingdom tend to view emails at work time from 8:00 to 18:00. While in Korea, users tend to work in much longer time with many users checking their emails early in the morning or late at night.

5.3 Baselines

Many previous active learning recommendation methods [12, 17, 31] require at least a small amount of initial ratings, which is not applicable for our problem since emails requiring prioritization are completely cold items. Due to the restrictions of broadcast emails, i.e. the same sender and limited types of interaction, most email prioritization methods cannot be directly applied to our task [18, 13, 27] or lack key features [1, 39, 36] if applied.

In the experiment section, we refer to our own method as Positive-feedback-oriented Active Learning (*PAL*). We try our best to adapt the following methods from previous literature for comparison. The first baseline is an adaption of the email prioritization algorithm used for Gmail inbox. The second baseline is adapted from a hybrid recommendation algorithm. The next 3 baselines are 3 active learning based recommendation algorithms from different literatures. We use them to replace the user sampling part of our own algorithm and for all the active learning baselines, the classic weight regularized matrix factorization method designed for implicit feedback from [28][14] [11] is used for the preference score prediction and a logistic regression model is used for label prediction. The last baseline is a variation of our own method by eliminating the weighting scheme in the weighted low-rank approximation process. For all active learning methods including our own, the percentage of sampled users is set to 10% and the length of time period waiting for users' response is set to 1 hour unless otherwise specified.

5.3.1 Importance Ranking (IR)

We adapt the importance ranking algorithm used for Gmail Priority Inbox[1]. Four categories of features are considered in the model, including social features, content features, thread features and label features. However, due to the characteristics of broadcast emails, social features, thread features and label features are inapplicable. We generate its content based feature as follows. Each email waiting for prioritization is represented as a term vector where each dimension is the tf-idf value of the corresponding term extracted from the text data of emails. For each user, we generate a user interest profile by aggregating the term vectors of the emails he has read. For an email requiring priority prediction, the cosine similarity between the user interest profile and the email term vector is calculated as the content feature and a logistic regression model is trained for label prediction.

5.3.2 Hybrid Based Prediction (HBP)

HBP is the method described in section 4.1.2, which we used to predict a user's preference towards a new email. It is a combination of content based recommendation and item based collaborative filtering. Once we get users' preference scores for the email, a logistic regression model is used for label prediction. Please refer to section 4.1.2 for more details.

5.3.3 Popular Sampling Active Learning(PSAL)

Inspired by [10, 33], for a new email e_{new}, we sample k users who have viewed the highest number of emails. This method is also equivalent to sampling the users with largest Variance or Entropy of Ratings [33], because in our task, we only have one-class implicit feedback and if we treat all the items without feedback as negative feedback (rating 0), to sample a user with the largest entropy/variance, we need to find some one with equivalent number of positive and negative ratings. However, since most users have far more negative rating than positive ratings, sampling user with large entropy/variance is thus equivalent to sampling popular users.

5.3.4 Coverage Sampling Active Learning(CSAL)

Inspired by [9, 33], for a new email e_{new}, we sample k users who have highly co-viewed emails with other users. Here, $Coverage(i) = \sum_j n_{ij}$, where n_{ij} is the number of emails that are viewed by both users i and j. The users with high $Coverage$ values are then sampled. The heuristic used by this strategy is that users co-view the same emails with many other users can better reflect other users' interests, and thus their viewing behaviour is more helpful for predicting the viewing behaviour of other users.

5.3.5 Exploration Sampling Active Learning(ESAL)

Exploration is important for the completely cold emails in our task [37]. Inspired by [32, 3], we firstly construct the user-email viewing matrix (entries are equal to 0 or 1). Then the matrix is normalized and each row of the matrix is regarded as a user vector. Finally, we gradually sample users ensuring that the sampled users are similar to the unsampled users and are less similar with the already sampled users. The intuition is the sampled users should be both representative of the unsampled users while diverse among the sampled users.

5.3.6 PAL Without Weight(PAL-SVD)

In order to test how the weighted low-rank approximation impacts the result, we propose another baseline called *PAL-SVD*, which is the same as our algorithm except that we eliminate the whole weighting scheme mentioned in section 5.2.1 and just use a basic SVD model to do the prediction.

5.4 Evaluation Metrics

Since our task is a classification task, we use precision, recall and f-score as the main evaluation metrics. Based on the predicted label from the algorithm and ground truth label from the data set, a prediction is either true positive (tp), true negative (tn), false positive (fp), or false negative (fn). The metrics are defined as

$$Precision = \frac{tp}{tp + fp}$$

$$Recall = \frac{tp}{tp + fn}$$

$$F - score = \frac{2 * precision * recall}{precision + recall}$$

5.5 Results and Analysis

5.5.1 Algorithm Comparison

In this section, we compare PAL with all baselines. Since different active learning algorithms query different sets of users for feedback, to make fair comparisons, the performance is evaluated on the same set of users, which is the intersection of the unsampled user sets of all the compared algorithms.

As shown in Figure 3, our method significantly outperforms all the baselines on all the evaluation metrics. Traditional email prioritization methods like IR [1] does not perform well in broadcast email prioritization, because many types of key features cannot be applied here due to the "one sender" and "limited type of interaction" challenges. HBP can work on our task, but the noises hidden in the content

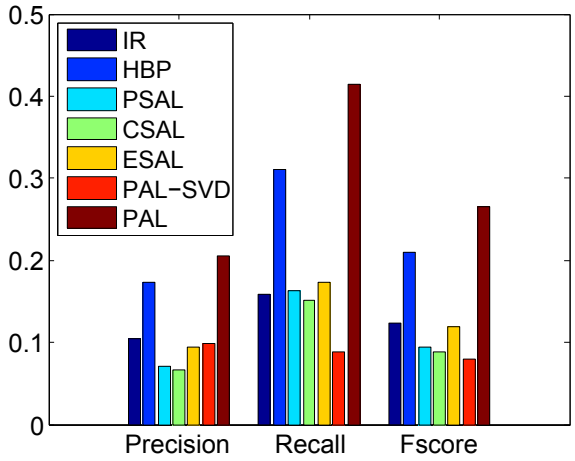

Figure 3: Performance Comparison of Various Algorithms

feature inevitably prevent the algorithm from generating relatively more accurate results.

Active learning recommendation methods may improve the prediction accuracy because the collected feedback allow us to predict the prioritization based on other user's responses by collaborative filtering. However, sampling strategies matter here, since in our task we can only collect the implicit feedback from a small portion of users in a very short time period. Our method outperforms all the other active learning baselines by considering users' preferences, active time distributions and email-viewing frequencies when sampling users. The proposed weighted low-rank approximation method uses the predicted positive feedback probabilities from the sampling phase as penalty weights to eliminate the bias from the sampling phase and it also works well in practice (PAL-SVD vs. PAL). It is worth noting that sampling methods like PSAL, CSAL and ESAL are non-personalized, which means they cannot generate different user samples for different emails and results in a negative impact on the prediction accuracy. Methods like PSAL and CSAL tend to sample popular users who almost like (read in our case) every item they see and these users actually cannot provide enough information for the matrix factorization based prediction. Moreover, they do not comply with our proposed fairness criterion, which means the same set of users may be sampled frequently. This issue will be discussed in section 5.5.4 in details.

5.5.2 Factors for Positive Feedback Prediction

As mentioned in section 4.1, we consider three different factors to help sample more positive feedback, namely, users' preferences to the email, their activity time distributions and their email checking frequencies. In this section, we remove these three factors one at a time and evaluate how each factor affects the percentage of positive feedback we get during sampling. Moreover, the length of the time-window during which we wait for responses from sampled users also impacts the percentage of positive feedback we can get during the sampling phase. So we also plot the percentage of positive feedback against different lengths of waiting time-windows. For users' activity time probability feature, if the time-window length is larger than one hour, we consider the

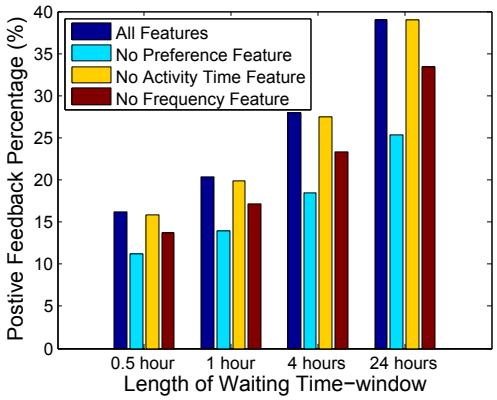

Figure 4: Factor Comparison for Positive Feedback

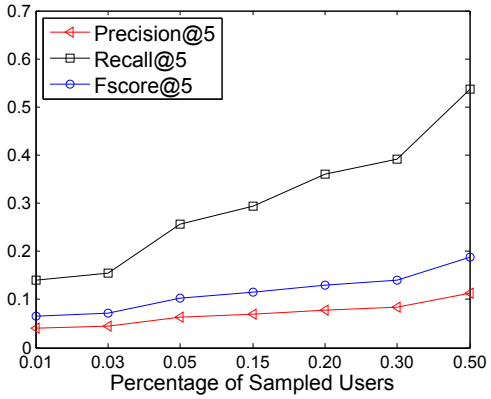

Figure 5: PAL Performance with Different Sampling Percentage

average probability of all the time intervals that are in the time window.

From the results displayed in Figure 4, we can see all these three factors are helpful in boosting the positive feedback rate in the sampling phase and thus further increase the prioritization precision. User preference and email checking frequency contribute more than activity time distribution. This makes sense because a user's active time is not always fixed. For instance, one may be on leave or travelling to another country on business. The noises and uncertainty in the activity time distribution factor affect its performance.

The percentage of positive feedback is positively correlated with the length of the waiting time-window, which is in accordance with our expectation. Our sampling strategy can work well even if we only give users a very short time period for response, because we consider users' responsiveness related factors (i.e. activity time distribution and email checking frequency) when sampling users. It is also worth noting that with the increase of the length of waiting time-window, responsiveness related factors become less important and the user preference factor becomes more important in raising the positive feedback rate.

5.5.3 *Active Learning Cost*

Active learning has cost. That's to say the sampled users to whom we send emails for feedback cannot benefit from the email prioritization service. Figure 5 shows that how the percentage of sampled users affect the final performance of the prioritization task. Note that even though increasing the percentage of sampled users helps to improve the prediction performance of the remaining users, sampling more users also increases the cost of active learning. Our method works well even if only a small portion of users(e.g. 5-10%) are sampled. This is because our sampling strategy tends to sample positive feedback and even with only a small percentage of users sampled, we can still get enough positive feedback for the model training and preference prediction.

5.5.4 *Sampling Fairness*

In our broadcast email prioritization framework, it is real users that we are sampling, so fairness is a very important criterion. If we keep sampling the same set of users again and again, they will quickly get annoyed and it is also unfair since they cannot benefit from the prioritization service. We compare our method with the active learning baselines with regard to the fairness criterion. For better comparison, We also add a random sampling baseline, which, due to its natural randomness, undoubtedly is the best strategy when we only consider fairness in user sampling. We use user coverage (the total number of sampled users during test) and the average sampled times as metrics. The results are stored in Table 3. From the results we can see, our method covers 2 times more users compared with PSAL, CSAL and ESAL. The results make sense since PSAL, CSAL and ESAL are unpersonalized active learning sampling strategies, which can easily cause serious fairness issues.

Table 3: Sampling Fairness Comparison

Sampling Methods	Coverage	Average Sample Times
Random	2805	65.78
PAL	898	205.48
PSAL, CSAL, ESAL	280	659

6. CONCLUSION

In this paper, we present the first framework for personalized broadcast email prioritization and propose a novel active learning framework to solve the problem. To exploit the collaborative filtering features of broadcast emails, we devise an active learning strategy that aims to sample enough positive feedback within the limited time window by exploiting factors including users' preferences, their activity time probability distributions and their email checking frequencies. A weighted low-rank approximation method is proposed to eliminate the possible bias from the sampling phase and generate accurate preference estimates for the remaining users. Finally we develop a feedback-sensitive classification method for personalized priority prediction. Comprehensive experiments are conducted on an industrial dataset from Samsung Electronics and the results show that our method outperforms all the baselines and the various factors considered are indeed useful to help to collect more positive feedback even under strict restrictions.

Since this is the first work using active learning for email prioritization, there is still much room for future research. First, in our work an approach that prefers positive feedback is used to sample users, which may not be the optimal sampling approach. For example, better performance may

be achieved by combining our positive feedback sampling approach with the coverage sampling baseline mentioned in section 5.3. Active learning models considering multiple sampling criteria at the same time will be proposed in our future work to increase the prediction precision. Secondly, even though we use a real industrial dataset in our experiment, the dataset is still small and with its limitations. We will try to look for a larger dataset with multiple mailing lists and more diverse user actions in the future.

7. ACKNOWLEDGMENTS

This work was supported by the National Basic Research Program of China(973 Program) under Grant 2013CB336500, NSERC Discovery Grant, National Science Foundation of China (Grant No. 61373118, 61522206, 61173186), National Key Technology R&D Program (2014BAK15B02 & 2012BAI34B01).

8. REFERENCES

[1] D. Aberdeen, O. Pacovsky, and A. Slater. The learning behind gmail priority inbox. In *LCCC: NIPS 2010 Workshop*, 2010.

[2] N. Chan. A-optimality for regression designs. *JMAA*, 87(1):45–50, 1982.

[3] R. Chattopadhyay, Z. Wang, W. Fan, I. Davidson, S. Panchanathan, and J. Ye. Batch mode active sampling based on marginal probability distribution matching. In *SIGKDD*, pages 741–749. ACM, 2012.

[4] M. Chui, J. Manyika, J. Bughin, R. Dobbs, C. Roxburgh, H. Sarrazin, G. Sands, and M. Westergren. The social economy: Unlocking value and productivity through social technologies. *McKinsey Global Institute*, (July):1–18, 2012.

[5] S. A. Danziger, J. Zeng, Y. Wang, R. K. Brachmann, and R. H. Lathrop. Choosing where to look next in a mutation sequence space: Active learning of informative p53 cancer rescue mutants. *Bioinformatics*, 23(13):i104–i114, 2007.

[6] M. Elahi, M. Braunhofer, F. Ricci, and M. Tkalcic. Personality-based active learning for collaborative filtering recommender systems. In *AI* IA*, pages 360–371. Springer, 2013.

[7] S. Feldman. Recommendations with Thompson Sampling. pages 1–5, 2015.

[8] Z. Gantner, L. Drumond, C. Freudenthaler, S. Rendle, and L. Schmidt-Thieme. Learning attribute-to-feature mappings for cold-start recommendations. *ICDM*, pages 176–185, 2010.

[9] N. Golbandi, Y. Koren, and R. Lempel. On bootstrapping recommender systems. In *CIKM*, pages 1805–1808. ACM, 2010.

[10] N. Golbandi, Y. Koren, and R. Lempel. Adaptive bootstrapping of recommender systems using decision trees. In *WSDM*, pages 595–604. ACM, 2011.

[11] G. Guo, J. Zhang, Z. Sun, and N. Yorke-Smith. Librec: A java library for recommender systems. In *UMAP*, 2015.

[12] T. Hofmann. Collaborative filtering via gaussian probabilistic latent semantic analysis. In *SIGIR*, pages 259–266. ACM, 2003.

[13] E. Horvitz, A. Jacobs, and D. Hovel. Attention-sensitive alerting. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pages 305–313. Morgan Kaufmann Publishers Inc., 1999.

[14] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 263–272. IEEE, 2008.

[15] Z. Huang. Selectively acquiring ratings for product recommendation. *ICEC*, pages 379–388, 2007.

[16] T. Jackson, R. Dawson, and D. Wilson. Case study: evaluating the effect of email interruptions within the workplace. *EASE, Keele, UK*, (April):3–7, 2002.

[17] R. Jin and L. Si. A bayesian approach toward active learning for collaborative filtering. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 278–285. AUAI Press, 2004.

[18] L. Johansen, M. Rowell, K. R. Butler, and P. D. McDaniel. Email communities of interest. In *CEAS*, 2007.

[19] R. S. John and N. R. Draper. D-optimality for regression designs: a review. *Technometrics*, 17(1):15–23, 1975.

[20] R. Karimi, C. Freudenthaler, A. Nanopoulos, and L. Schmidt-Thieme. Non-myopic active learning for recommender systems based on Matrix Factorization. *IEEE IRI*, pages 299–303, 2011.

[21] R. Karimi, C. Freudenthaler, A. Nanopoulos, and L. Schmidt-Thieme. Towards optimal active learning for matrix factorization in recommender systems. *ICTAI*, pages 1069–1076, 2011.

[22] R. Karimi, C. Freudenthaler, A. Nanopoulos, and L. Schmidt-Thieme. Exploiting the characteristics of matrix factorization for active learning in recommender systems. *RecSys*, page 317, 2012.

[23] A. Kohrs and B. Merialdo. Improving collaborative filtering for new users by smart object selection. In *ICMF*, 2001.

[24] Y. Koren, E. Liberty, Y. Maarek, and R. Sandler. Automatically tagging email by leveraging other users' folders. *SIGKDD*, pages 913–921, 2011.

[25] W. Liu and T. Wang. Online active multi-field learning for efficient email spam filtering. *Knowledge and Information Systems*, 33(1):117–136, 2012.

[26] C. E. Mello, M.-A. Aufaure, and G. Zimbrao. Active learning driven by rating impact analysis. In *RecSys*, pages 341–344. ACM, 2010.

[27] C. Neustaedter, A. B. Brush, M. A. Smith, and D. Fisher. The social network and relationship finder: Social sorting for email triage. In *CEAS*, 2005.

[28] R. Pan, Y. Zhou, B. Cao, N. N. Liu, R. Lukose, M. Scholz, and Q. Yang. One-class collaborative filtering. In *ICDM08*, pages 502–511. IEEE, 2008.

[29] S. Radicati, P. Analyst, and J. Levenstein. Email Statistics Report , 2013-2017. 44(0):2013–2017, 2013.

[30] L. Rokach, L. Naamani, and A. Shmilovici. Pessimistic cost-sensitive active learning of decision trees for profit maximizing targeting campaigns. *DMKD*, 17(2):283–316, 2008.

[31] N. Roy and A. McCallum. Toward optimal active learning through monte carlo estimation of error reduction. *ICML*, pages 441–448, 2001.

[32] N. Rubens and M. Sugiyama. Influence-based collaborative active learning. In *RecSys*, pages 145–148. ACM, 2007.

[33] N. Rubens, R. Tomioka, and M. Sugiyama. Output divergence criterion for active learning in collaborative settings. *IPSJ*, 2(3):87–96, 2009.

[34] B. Settles, M. Craven, and S. Ray. Multiple-instance active learning. In *NIPS*, pages 1289–1296, 2008.

[35] D. J. Sutherland, B. Poczos, and J. Schneider. Active Learning and Search on Low-Rank Matrices. *SIGKDD*, pages 212–220, 2013.

[36] G. Tang, J. Pei, and W. S. Luk. Email mining: tasks, common techniques, and tools. *Knowledge and Information Systems*, pages 1–31, 2013.

[37] N. Tintarev and J. Masthoff. *Active Learning in Recommender Systems*, volume 54. Springer US, 2011.

[38] B. Wang, C. Wang, J. Bu, C. Chen, W. V. Zhang, D. Cai, and X. He. Whom to mention: expand the diffusion of tweets by@ recommendation on micro-blogging systems. In *WWW*, pages 1331–1340, 2013.

[39] S. Yoo, Y. Yang, F. Lin, and I.-C. Moon. Mining Social Networks for Personalized Email Prioritization. *SIGKDD*, page 967, 2009.