**Query Expansion for Email Search**

Three state-of-the-art expansion methods are examined:

1) a global translation-based expansion model;

2) a personalized-based word embedding model;

3) the classical pseudo-relevance-feedback model.

Used two large datasets of real user queries and corresponding messages

the relevance of the message to the query is estimated by a complicated scoring function that considers many signals of relevance, including the message freshness, the textual similarity to the query, the user interaction with the message, and many more signals [2]

While the average query length on the Web is about three terms per query, the average length in the email domain is only 1.5 terms per query [2]

(problems with extremely short queries) -> Query expansion can help

**Translation model**

Given a query log of a commercial Web mail service, we extract a large dataset of email queries, each associated with clicked (presumably relevant) messages. e data pairs are used to train a translation model that maps query terms to relevant message terms. queries are then expanded by the most related terms to the query terms.

ignoring user personal preferences and biases

------------

Expands the query using a global translation model, learned from the system’s query log.

Query logs are composed from the user queries, each one is associated with a list of up to 100 messages, where messages that were clicked by the user are marked relevant. If there are more than one clicked, only the last one is relevant.

Query pairs < Q,s > where Q is original user query and “s” is the texto of the subject of the clicked message for that query.

IBM model 1 used for query expansion. For each term **t** the probability distribution over the Vocabulary V is Pr(w|t) = prior probability of translating term **t** to term **w**.

We score the vocabulary terms for a given query:

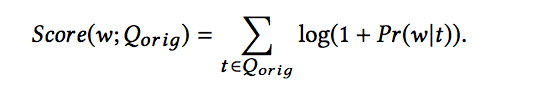


Ilustração 1

We then select the top-k scored words for query expansion. The selected terms are added to the original query, weighted according to their translation score, where term weight is normalized with respect to all k expanded terms.

Finnaly, the expanded query is linearly combined with the original query.

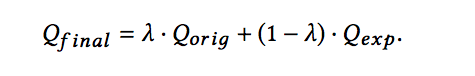


Ilustração 2

**Personalized model**

Our second expansion model expands the query in a personal manner.

The content of the personal mailbox of each user is used to train a word embedding model [11] which projects all terms of the messages into a dense lower dimensional space. e nearest neighbors of the query terms in the embedding domain are then used for expansion [8].

------------------------

As users search over their own data, using their own personal vocabulary, expanded terms should reflect their own preferences.

In the personal query expansion model, the query is expanded with terms that are “semantically similar” to the query terms, where similarity is measured in the context of the personal user content.

Specifically, we use the Word2Vec Continuous Bag- of-Words (CBOW) approach [11] which embeds terms in a vector space based on their co-occurrence in windows of text. e cosine similarity between the term vectors was shown to correlate with semantic similarity. Accordingly, we select terms similar to the query in this vector space for query expansion.

The “Subject” field terms, and the “From” and “To” field terms of all user mailbox messages are represented by the Word2Vec model.



Ilustração 3

The final term score for the given query is determined by aggregation over the query terms using Equation 1, and the top-k scored terms are then added to the query using Equation 2.

**Feedback model**

In contrast, relevance feedback based methods do not depend on any auxiliary data resource. Pseudo relevance feedback (PRF) methods construct an expansion model from the search results and then expand the query using the inferred model [9].

The expanded query can be used for re-ranking the search results, or to be submited as a new search task

PRF methods improve the search and efectiveness on average, however, they are very sensitive to the quality of the original search results.

-----------------

In this model, the top scored messages retrieved for the query are used for constructing an expansion model.

Given the list of retrieved messages, the relevance model is used to construct the expanded query.

model is defined by Pr(t|RM1) = m∈M PrMLE(t|m)Pr(m|q)

where PrMLE(t|m) is the maximum likelihood estimated of term t in message m

t f (t ∈ m) is the number of occur- rences of t in m.

A message is de ned here as a concatenation of its “Subject”, “From”, and “To” eld terms.

Given the relevance model, we select the top-k scored terms according to the model and expand the query with them using Equation 2. e expanded query is then used to re-rank the search results.

**Ranking model**

Explore the potential merit of the query expansion approaches over an effective representation of the original query. There were used two retrieval models.

First one measures only textual similarity of the query to the message based on the sequence dependence model. Each message is then scored with respect to the query using a linear interpolation of the SDM scores of its fields.

The second retrieval model was used a learning-to-rank scoring function for email search, which considers the message freshness, the textual similarity to the query, and the user actions on the messages. The similarity of the expanded query to the message is added as an additional feature to the scoring function.

**Experiments**

Each entry in the query log consists of a query text and the corresponding result list

we consider only search results that were received no more than six months before the query was issued, and only the latest clicked message in the result list is considered relevant.

we constructed a personal search index for each of the users in our training set

Messages were collected differently in two query sets.

* **LogData**, we collected the union of all messages of the user in the query log, i.e., all messages retrieved as a result of at least one of the user’s queries
* **MailBox**, we used the mail data of a small group of users for which their mailboxes are fully available for us; all of the messages in the specific user’s mail box constitute her personal collection.

Queries and messages were processed in the same manner. Stopwords were removed, and text was stemmed using Lucene’s minimal English stemmer1

For training the translation model we sampled queries of 11,200 users, over a period of 9 months, resulting in approximately 2.5M pairs of queries and subjects. these users are different from those whose data was collected for our query sets.

the performance of the different models is evaluated using two measures: Mean Reciprocal Rank(MRR) and success@n.

The models that we study incorporate several free parameters. Free parameter values were set in the following manner. We split the query set at random to training (1/3) and test (2/3) sets. Then, we choose parameter values that maximize MRR on the training set, and report performance on the test set.

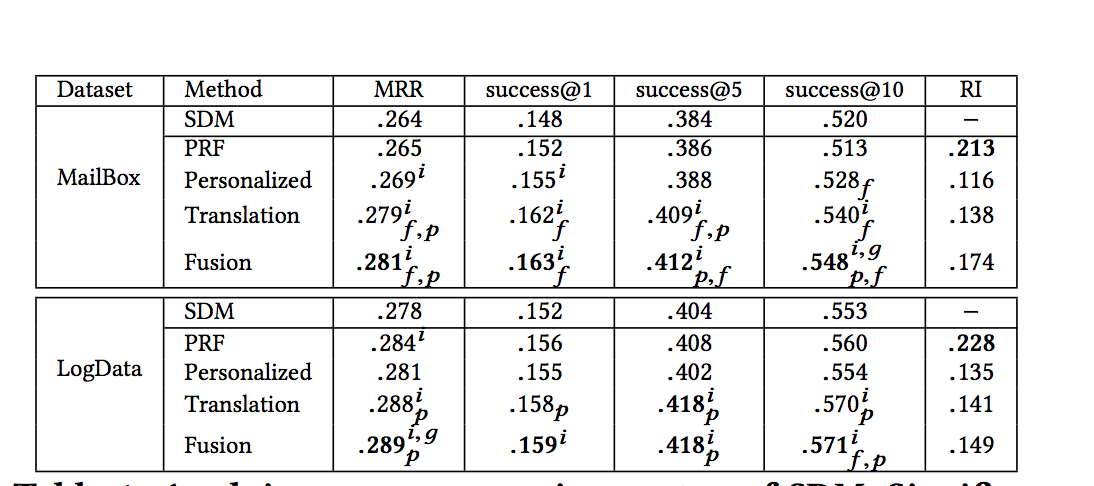
**Experimental Results**

According to the results, all query expansion approaches outperform SDM in a vast majority of reference comparisons; most of the MRR improvements are statistically significant.

We also study the Fusion approach which utilizes both the Personalized and the Translation models

Among the three suggested approaches, Translation is the most effective according to Table 1.

the performance of Translation (MRR and success@n) often dominates those of PRF and Personalized



The Personalized expansion approach is effective in the case of MailBox. It outperforms SDM for all evaluation measures .

In LogData, on the other hand, the Personalized approach never outperforms SDM in signicant manner. Tis difference in performance between the two datasets can be ascribed to the number of available messages per user which is larger in the case of MailBox. As the Word2Vec approach requires large amounts of data in order to learn an efective model, the resulted personalized models in the case of LogData may be less efective.

The PRF approach signi cantly improves SDM in terms of MRR only in LogData.

In addition, PRF is the most robust expansion approach in terms of RI. A possible explanation for this is that PRF models are induced from an initial result list. Such models, therefore, tend to be closer to the original query as compared to other approaches that utilize some external source (which is constructed in a non-query dependent manner). Consequently, PRF models may be less risky.

In Fusion, the term lists of both approaches are linearly interpolated using a free parameter learned from {0.0, 0.1, 0.5, 0.9, 1.0}. we fuse expansion term lists of 50 terms of Translation and Personalized, and then extract k terms with the highest combined score for expansion (using Equation 2). the performance of Fusion dominates those of the two approaches.

In terms of RI, we can see that Fusion is much more robust than both approaches. Given the high performance of the oracle, the question of how to optimally combine the two approaches remains open.

Queries divided into three intents: Person (looking for messages of a speci c contact), Company (messages of a speci c company), and Content (all the rest). Person, Company, and Content queries are (approximately) 40%, 10% and 50%, of all queries in both data sets, respectively.

In both datasets the best performance of Translation is a ained for company queries; the improvements over SDM are signi cant and by more than 17%. A possible explanation is that Translation is learned using query logs of many users. Company queries and the corresponding relevant messages are o en common among di er- ent users since these messages are usually generated by machine. erefore, the resultant translation model is be er in generalizing over users.

Integration with REX. e performance of the di erent approaches when applied on REX is presented in Table 3. We also present the performance of Time, an approach in which messages are ordered according to their received date. Such approach is very common in commercial email services.

As in the case of SDM, Translation is the most e ective expansion approach over REX in both datasets. In MailBox we can see that all methods outperform REX

In LogData, on the other hand, while all methods, except PRF, improve over REX in terms of MRR, the improvements are signi cant only in two cases. A possible reason for this modest improvement is that in LogData the original ranking of REX is presumably of higher quality. In such a case, the added value of expansion terms to the high quality ranking function might be limited.

CONCLUSIONS

All the expansion methods outperform a standard SDM based ranking model in signi cant manner, and the translation model outperforms the other expansion methods.

However, the amount of improvement over REX, a mature and well trained LTR ranking model, was found to be only modest. e questions how to optimally integrate the expanded query into learning-to-rank model and how to optimally fuse the expansion models are far from being solved and are le for future work.

**Understanding and Modeling Success in Email Search**

**Introduction**

In contrast to other search searching, such as web search, there have been few studies of user behavior and models of email search success. Third party judges can not look at email search queries or email message content requiring new modeling techniques.

opt-in client application which monitors a user’s email search activity and then pops up an in-situ survey when a search session is finished. We then merged the survey data with server-side behavioral logs. is approach allows us to study the relationship between session-level outcome and user behavior, and then build a model to predict success for email search based on behavioral interaction patterns.

Our results show that generative models (MarkovChain) of success can predict the session-level success of email search better than baseline heuristics and discriminative models (RandomForest).

Although it is important to develop evaluation metrics for email search, there are also challenges in doing so. In this work, we turn to server-side email search logs for building evaluation metrics. Modern email systems may operate in the web browser or in a client-server setup that allows the centralized collection of activity logs. For evaluating email search, these logs may be comparable to logs studied in web search.

However, there are some important differences between web and email search logs. Email search logs should be managed in a way that can preserve the privacy of users. The logs studied in this paper do not have any text from the user’s query or messages, to maintain the privacy of users, unlike in web search logs where the content of queries and web pages are often studied. Tis adds extra challenges in finding signals that will be useful for evaluating email search logs.

in email search the user remains in the system and their activities can be logged. For instance, after querying and selecting (clicking) a message, the timing and type of their next actions may indicate the relevance of the selected message. Given this availability of rich signals, the next challenge is finding a good interpretation of them, which is a nontrivial task due to the variety of signals.

Building a model of success has been extensively studied in Web search settings [13], mostly relying on annotation of search sessions by human judges. To build such model for email search we need to switch from third party annotation of sessions to in-situ labeling by the searcher [12].

First, we characterize the overall differences between successful and unsuccessful email search, at the user level, session level and the message level. Second, we characterize, for the variety of email actions (such as reply, forward, ag, delete), which actions are associated with successful rather than unsuccessful search sessions. Third, we develop predictive models to distinguish between successful and unsuccessful email search sessions. These insights and success models for email search are new, and should offer a useful comparison point in future email lab studies, test collections and real-world controlled experiments.

**RELATED WORK**

Information retrieval evaluation:

The relevance judgments are usually made by third party judges rather than in-situ users as they interact with the search engine. For this reason it is more desirable to build a behavior-based evaluation metric. Then Markov Chain models are built to characterize the relationship between logged activity and annotations. Our only source of query and message text (such as message subject) in this study is from the in-situ survey, not the logs.

Because of these privacy concerns, there is a hard tradeoff between having real users with real email and having third party relevance judging. We now consider evaluation approaches that do not rely on third party judging. Table 1 summarizes characteristics of the users, types of labels collected and evaluation metrics for five studies.

Email management and retrieval:

Email management strategies have been identified [21] according to the number of folders used and the frequency of their use. No filers do not use folders except to archive or delete messages every few months. Frequent filers use folders, making daily passes of move and delete actions. Spring cleaners do the same but every 1-3 months rather than daily.

**3 DATA COLLECTION**

in-situ survey we ran for client-side data collection, followed by the description of server-side activity logs and the merging of client-side and server-side data.

**3.1 In-situ Survey Methodology**

collect the out- come of each email search session. the app monitors search activity in the user’s email client. When the end of a search session is detected, the app pops up an interface for a short survey about the success and ease of the search session.

• Explicit ending: the user clears the query input control and results by navigating away from the search interface

• Implicit ending: the user abandons the email search inter- face, which we defined as 3 minutes of inactivity

we found these conditions work well in most cases.

Figure 1 shows the user interface for the survey app. the user first answers a question about the success of their search session which included the query (or queries) shown. If the user responded ’yes’ to the first question, then they are asked to answer additional questions about effort and relevance of clicked messages. the first additional question is the ease of finding what they were looking for. the second question asks about the relevance of each email message that was clicked. Finally, there is an optional box for additional comments about the search.

Once a survey is submitted, all the information visible in the survey pop-up window was sent to a remote database. the participants were mostly IT professionals working in various locations in US and abroad. After 2 months, we had 1875 email search sessions collected from 65 users.

**3.2 Activity Log Collection**

In order to obtain a full view of users’ search activities, we also collected the server-side log data from our organization’s email server. For privacy reasons, the log data does not contain any textual information, yet it includes the activity user engaged in along with timestamp. e list of activities are shown in Table 2.

**3.3 Data Processing**

Our goal is to get a full picture of the users’ email search including users’ subjective impression along with activity records, therefore we merged the survey data with server-side activity logs. we first aligned server-side (activity) and client-side (survey) data for each user by timestamp. BUT server and client-side timestamp does not always align for various reasons, which resulted in some data loss. then split the data by session. We used the rst search activity as the beginning of the session, and the user’s submission of survey as the end of the session.

**4 UNDERSTANDING SUCCESS IN EMAIL SEARCH**

we analyze the in-situ survey data to better understand success and effort in email search.

**4.1 Understanding User-level Success**

We first compute six measures of search activity for each user and the look at the distribution across users. the first two plots show that the median number of queries per session is slightly less than 2 and the median number of read messages is about 3. the third plot shows that proportion of viewed messages that were relevant was slightly less than 50%. the fourth plot shows that the proportion of queries with advanced operators (e.g., from:) was approximately 20%, but the variance across users is large. the fifth plot shows that search success is high (median 90%) and there is li le variance across users. the final plot shows the number of judgments submitted per user.

search operators as shown in Table 4. Most query operators correspond to different email fields and are therefore self-explanatory, except for the ‘Conversation:’ operator which provides a way to find the messages in the same thread.

We now consider the extent to which these per-user session measures are correlated.

**4.2 Understanding Session-level Success**

As for query counts, we find that Success/Easy sessions have the lowest number of queries. Interestingly, Success/Hard sessions had higher message count than Failure sessions. It is possible that the type of search tasks in Success/Hard sessions are very important, so users were willing to spend extra e ort to complete the search compared to Failure sessions where they gave up.

We can see that failure sessions have longer queries on average compared to successful sessions.

We also looked at the ratio of messaged marked relevant by participants, and the ratio was 60% in Success/Easy sessions, which is more than twice as high as Success/Hard sessions. Since we did not ask people to judge the relevance of messages when they failed to nd what they were looking for, this value is not available for Failure sessions.

Message age is de ned by the interval between when the email was sent or received and the time of search. e overall age of messages read during a search session could be a ected by various factors.

**4.3 Understanding Message-level Success**

In addition to session-level success labels, we also collected relevance labels for each message read. Here we delve into this, focusing on dwell time at each message, which is an important signal in understanding the users’ engagement with individual messages [15].

if a message is the last one read during a session, it is an important signal for relevance in and of itself. Table 7 confirms this intuition, where the ratio of relevant messages among last-read messages was 54%, compared to 24% for non-last read messages.

If being the last message read indicates relevance, what would the dwell time for non-last messages tell us about its relevance? Table 8 shows that the average dwell time for relevant messages is about 37 seconds, whereas the value is 22 seconds for non-relevant messages.

**5 MODELING SUCCESS IN EMAIL SEARCH**

build a model that could predict the success of an email search. could be used to evaluate a new search method against baseline.

we decided to model session-level success. We focused on interpretability in this work because the goal was to build a transparent model which provides insights about its outcome upon inspection.

**5.1 Experimental Design**

the data contained the list of activities that occurred during each search session, along with various session-level properties and labels from the survey. we used the data from the first month for training models and the data from the second month for evaluation.

the evaluation setup is as follows: each model takes in various session-level features and calculates a score for session-level success. the score for each model is then compared against the ground truth session-level labels we obtained from the survey. this is a binary classification task which predicts the success vs. failure of a given session. Since reading time has previously been shown to be an important feature in predicting search success [13], we built every model with or without reading time.

metric we used AUC (area under curve) of the ROC (receiver operating characteristic) curve [4].

**5.2 Models Compared**

Our first model is a heuristic baseline, which associates a subset of activities with session-level success, and outputs a positive label when any of these activities are found during a search session. activities we considered as success for this baseline model includes ‘Move’,‘Reply’, ‘Forward’, ‘Flag’, ‘ReadingLong’

A more data-driven approach is to use the survey data we collected to determine the weights. Since we have a sequence of activities along with success label for each search session, a natural choice is a discrete Markov Chain of user behavior.

where the zero-order Markov Chain assumes the independence among activities, and the first-order Markov Chain assumes that the present activity is dependent on the previous activity. the Markov Chain models we experimented with can be further distinguished based on whether reading time is considered or not. we decided to incorporate time by splitting the reading activities into short reading vs. long reading, where the short reading is defined by the reading activity with duration less than 30 seconds.

Now we focus on how we derive success score from models. we can estimate two models (Ms and Mf ) corresponding to successful sessions and failure sessions, respectively.

EXPLICAR COMO FUNCIONA, BLA BLA BLA

We use RandomForest [5] for this reason, which is a discriminative model known to perform well out-of-the-box. We experiment with two models with Markov Chain features – activity counts from zero-order Markov Chain (RandomForest(0)), activity counts from first-order Markov Chain (RandomForest(1)). these are models based on the same set of features as Markov Chain models, except they are fed into RandomForest model.

the third RandomForest model we tested employed session-level features along with activity counts from first-order Markov Chain (RandomForest(1+S)). Session-level features we used include the count of queries, count of messages, ratio of operators in session queries and average message age. Note that the Markov Chain model does not accommodate these features since they do not have a probabilistic interpretation.

Markov Chain models have high interpretability in that the scores from individual activity contribute linearly to the final score. In contrast, the RandomForest models employ a large collection of decisions trees to produce a prediction, which makes the results harder to interpret. it has the advantage of being able to incorporate arbitrary features

**5.3 Experimental Results**

Here we present the outcome of prediction for session-level satisfaction. Table 10 summarizes the Area under ROC values for the models we evaluated. Note that the results are grouped into 1) baseline 2) variants of Markov Chain models 3) variants of RandomForest, with the columns showing the results with or without taking into account the reading time.

both Markov Chain and RandomForest models are much better than the baseline and that the Markov Chain model is slightly better than the RandomForest model with the same feature set. YET, Since RandomForest is a more powerful model, it may outperform Markov Model given enough training data.

first-order models that use activity pairs are always better; adding reading time provides a further boost in AUC in all cases; adding session-level features to RandomForest(1+S) model improves the performance as well.

An ROC curve characterizes the performance of a classifier by plotting Sensitivity (true positive rate) against Specificity (1 - false positive rate).

Figure 4 presents the ROC curve for two Markov Chain models, along with a baseline model. It shows that the use of activity pairs improves the sensitivity in all but the leftmost side of the plot (high specificity / low false positive rate region). this shows that the use of activity pairs provides additional evidence when the signal from individual activity is weak.

Figure 5 presents the ROC curve for three RandomForest models, along with the baseline model. It shows that the use of activity pairs improves the true positive rate in all but the leftmost side of the plot (high specificity / low false positive rate region), as we saw in Markov Chain model. Also, the inclusion of session-level features improves the performance across the board.

Figure 6 compares the ROC curve for MarkovChain(1) against RandomForest(1) model, along with the baseline. the Markov Chain model outperforms the RandomForest model in all but the leftmost side of the figure. this shows that the discriminative model is better at getting the first few instances right (i.e., high sensitivity at high specificity region) yet worse in overall.

**5.4 Further Analysis**

these analyses are based on Markov Chain models, which provides a simple interpretation of session-level results in terms of individual activities.

the odds ratio is calculated as the probability of observing an activity in successful sessions divided by the probability in failure sessions. top part of the table is dominated by typical message-level actions, which would be done for relevant message(s).

Table 12 presents the odds ratio for each activity pair.

Interestingly, we can see ReadingShort activity both at the top and bottom of the table, by which we can see that a short reading is positive when followed by reply or send activity, yet negative when it is found around search, delete, or another short reading activity. this result exemplifies the power of evaluating an activity in the context of other activities.

**6 CONCLUSIONS**

email search occurs over a private corpus, so it is useful to develop new methods of analysis and evaluation. We employed an in-situ survey that is tightly coupled with user activity logs, to get both explicit and implicit data on search success for the same individuals.

In the survey, 20% of queries had an operator, with the most common operator being “from”.

Easy successful sessions were also associated with finding more recent messages, with fewer shorter queries, and fewer opened messages.

High-e ort successful sessions, while rare, had more queries and opened messages, and the opened messages were older, suggesting the need for better information retrieval support for such cases.

Failed sessions were in between, perhaps indicating that the user gave up because the search was not all that important or they used other means for finding the relevant message.

At the message level, having a higher dwell time and being the last-selected message were both good success indicators

After joining with activity logs, a wide variety of activity types were found to be associated with session success. Specifically the following actions were all associated with search success: Reply, Move, Forward, Delete, Mark, ReadingLong and Flag.

Once the predictive models of search success have been learned, they can be applied to new log data without any new survey data. For example, MarkovChain(0) can be applied using only the data in Table 11 in equation (1). Future work could consider how to au- tomatically optimize an email search system to yield these positive events and avoid the negative events. One approach would be to treat our predictive model as a utility function, then use a factored design as in [20].

Mailbox-Based vs. Log-Based Query Completion for Mail Search

**Abstract:**

Recent research studies on mail search have shown that the longer the query, the better the quality of results. A known mechanism to assist users in this task is query auto-completion, which has been highly successful in Web search.

This is approach cannot be applied directly to mail search as personal query logs are small, mailboxes are not shared and other users’ queries are not necessarily generalizable to all.

We therefore propose here to leverage the mailbox content in order to generate suggestions. We then compare this approach to a recent study that augments an individual user’s mail search history with query logs from “similar users”, where the similarity is driven by demographics. Finally we show how combining both types of approaches allows for be er suggestions quality but also increases the chance that the desired message be retrieved.

**Introduction**

Users struggle more with formulating queries in mail than in Web search, with an average query length of 1.5 as opposed to 3 terms. The most dominant approach for query assistance today is query auto-completion that has been highly successful in Web search.

Two challenges: **First**, the personal query log (i.e., a single user’s search history), which is the primary source of auto-completion, is sparse; **second**, as argued in [1], messages are not shared across and queries, which are most often personal, might not generalize well to other users.

One solution to address the first challenge is to increase the size of query logs.

Consider leveraging additional sources such as mailbox content. Leveraging content has been used in the past for improving logs suggestions. In the enterprise domain, took advantage of additional sources such as structured catalogs to increase coverage. Closer to our work Bhatia et al.[3], proposed to generate suggestions directly from the corpus in absence of query logs, yet, they did not specifically tailor their approach to mail.

More specifically, we propose to generate suggestion candidates by extracting N-grams directly from the user’s messages, and rank them using a model specifically-tailored to mail. We go beyond simple frequency counts.

We associate with the generated suggestion, attributes from the message it originates from, such the message freshness, the folder it belongs to, or the actions that were performed over it, such as read, reply, etc. We compare the quality of these message-based suggestions to those generated by query logs, using the method proposed by [6] as our baseline.

We present a combined approach, where suggestions originate from both query logs and mailbox content.

**MAIL QUERY AUTO COMPLETION**

In the most common settings, the query completion mechanism takes as input a short string of characters entered by the user and returns a ranked list of full queries, referred as completions or more generally suggestions. We detail below our method for generating suggestions from the mailbox content, as well as a combined approach where suggestions are generated from both mailbox content and query logs.

**Mailbox-Based Suggestions**

We first generate from the user’s mailbox a list of terms and bigrams. Note that as we extract bigrams, we use the same approach adopted by [3], and “jump over” stop words. So for the text confirmation of order, instead of generating two bigrams, confirmation of and of order, none of which is a complete phrase, we generate one bigram, i.e. confirmation of order.

We then rank candidates, using a set of features, listed below according to their level of granularity:

**Mailbox-Level Features:** considering its frequency of occurrence in a given mailbox as compared to the entire mail domain. We thus use tf-idf score, where the mailbox is unified into a single document and the corpus is the collection of all mailboxes in a given geography. The **tf-idf** score is computed separately for unigrams and bigrams, then all scores are added as features to our model.

**Message-Level Features:** suggestions originating from a less important message should be demoted. We use several attributes of a message in order to compute its importance: actions performed over a message (e.g. read, ag, forward, etc.), which have been shown to be beneficial in mail relevance; we consider also whether the message was received or sent, as well as its parent folder; we consider the message’s recency, and use a variant of Equation (1), where we penalize candidates that appear mostly in very old messages.

**Field-Level Features:** Message fields (subject, from/to, body, etc.) play an important role in mail ranking. We compute the frequency and idf scores for each field. Specifically, we consider five fields: from, to, cc, subject, and a attachment name.

**Ranking Algorithm:** We use AROW [8], an online variant of SVMRank, which learns a linear weight vector through pairwise comparisons between the relevant candidate and other top-ranked candidates for each query. We tune the learning parameters through standard training, validation and testing on separate sets, where each set includes queries corresponding to a different time-frame. this way, we guarantee that the sets do not overlap and that we remain close to real-world settings.

**Combining Mailbox and Query Log**

A major drawback of using content rather than queries as source of suggestions is the well-known query-document mismatch. One immediate benefit of using mailboxes in addition to query logs, is to validate suggestions originating from other users’ query logs so as not to lead to empty results.

We first consider an online validation method: verifies that each term that appears in a log-based query suggestion does appear in the user’s mailbox, using an offline generated set of all the terms appearing in the mailbox. Limitation: multi-term suggestions will be validated without necessarily leading to an existing message. We still propose to compare its benefits to the safer online validation method, where we validate each of the top candidates using the search index and remove candidates that lead to an empty result list. This online validation method, while clearly preventing empty results, presents an obvious overhead in terms of latency.

The actual combination model is done by computing for each candidate a score that combines features from both the query logs and mailbox sources. Two main factors account for the candidate score: **(1)** its prevalence in important messages with respect to the user’s mailbox **(2)** its likelihood as a query, with respect to the user’s past queries as well as the joint query log of the entire user population.

**EXPERIMENTS**

**Dataset**

Our dataset consist of a sample of 23.7M queries issued by 4.4M unique users in March-July 2016 in the US traffic of Yahoo Mail.

We use it to calculate the frequency distribution of each suggestion, in a joint query log across all users in our dataset, as well as across different demographic sectors. Overall, we collected 38M messages, with an average of 21K messages per user.

Followed and employed a method for automatically generating training labels.

First, we sampled from our dataset a set of queries followed by a clicked result and generated for each query a set of prefixes that consist of the 1, 2, 3, 4 first characters and the first term of the query. For each prefix, we obtained all matching query candidates using an auto-completion trie structure over the entire data. Next, for each prefix, we marked as relevant the suggestion that was identical to the query that was issued by the user. We considered only queries that were followed by a click on a result in order to avoid low-quality queries. We refer to this evaluation method as query-based evaluation.

Consider a suggestion candidate to be useful if it leads to a search result page, where the position of the clicked message is equal or higher than its position in the search result page of the original query. We refer to this evaluation method as result-based evaluation.

**Mailbox-Based Suggestions**

We measured the quality of our mailbox-based suggester with respect to different feature sets, using the same metrics as [6], namely Mean Reciprocal Rank (MRR) of the relevant suggestion and success@k, with k = 5 as mail users are typically presented with a small number of suggestions.

As expected, the performance of the suggester varies with prefix length. When adding message-level features to uni ed-mailbox features, we note an increase in MRR of 90% for 1 character, vs. 3.5% for 4 characters.

we observe an additional increase when we consider the message fields, which sums up to an increase of 111.8% for one character and of 4.3% for four characters.

In terms of Success@5, as before, we observe similar improvement. We retain the Field-based features as our most successful Mailbox- based suggestion method and refer to it as the “Mailbox” method in the following experiments.

**Combining Mailbox and Query Log**

Baseline relies on personal and global query logs. We **first** evaluate the effect of our offline validation method by comparing its MRR and Success@5 scores to our baseline as well as to the more costly online validation.

the validation techniques leads to an MRR improvement of 4.1% for 3 chars prefix as compared to 4.6% for the online evaluation. the offline techniques is quite close to the online technique for 1-4 chars, and underperforms only for 1-term prefixes with an improvement of 0.4% as compared to 5.5% for online.

Given that the latency of online validation will be in most cases prohibitive in the case of a large scale mail service like Yahoo Mail, these results are quite encouraging, allowing us to use a cheap offline validation method to reduce the risk of empty results sets.

Thus, for further experiments, we use the offline method in order to evaluate our combined model. We now incorporate mailbox-based suggestions and compute a ranking over all candidates using features from both sources.

**First**, we note that the mailbox-based suggester outperforms the log-based one with an increase of between 13.3% and 34.2% in MRR in all settings except for single character prefixes.

The combined approach performs best, with a significant increase in MRR and Success@5 for all prefix lengths. On one hand, the personal mailbox content ensures the relevance of suggestions to the user’s needs. On the other hand, the query logs help bridge the query-document language gap, and, even if in a more moderate manner than in Web search, offer popular suggestions.

**Manual Evaluation**

In order to get some qualitative feedback, we asked ten Yahoo assessors to issue 20 queries each, using their own mailboxes, and rate the presented suggestions for a prefix of 2,3 and 4 characters. They judged whether each suggestion was well-formed or not, and whether it was relevant to their query intent. In terms of usefulness, the assessors found our combined model to perform best, on average 88.4% better than Logs and 3.8% better than Mailbox. In addition, the assessors rated 93% of the suggestions from the combined model as well-formed.

**Result-Based Evaluation**

we conducted a result-based evaluation of the log-based, mailbox-based and combined approaches. can be more than one relevant suggestion, we used the MAP metric for different prefix lengths.

We note that the combined suggester significantly outperforms both the log- and mailbox-based suggesters.

**CONCLUSION**

We demonstrated that due to the private and personal nature of the mail domain, the mailbox content of the individual user brings clear value. We used a range of mail-specific features at different levels of granularity (specifically, mailbox-, message- and field-based features), and analyzed their efectiveness. We conducted a manual evaluation and large scale experiments on real data from Yahoo Mail, and compared our mailbox-based suggestions to a recent log-based method proposed in [6]. In addition, we presented a combined approach that incorporates the benefits of both logs-based and mailbox-based methods, and significantly outperforms both.