RC25059 (W1010-065) October 18, 2010
Computer Science

IBM Research Report

Information at Your Fingertips: Contextual IR in Enterprise Email

Jie Lu, Shimei Pan, Jennifer Lai, Zhen Wen
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

Research Division
Almaden - Austin - Beijing - Cambridge - Haifa - India - T. J. Watson - Tokyo - Zurich

Information at your Fingertips: Contextual IR in Enterprise Email

Jie Lu Shimei Pan Jennifer Lai Zhen Wen
IBM T. J. Watson Research Center
19 Skyline Drive, Hawthorne, NY 10532 USA
{jielu, shimei, jlai, zhenwen}@us.ibm.com

ABSTRACT
We present ICARUS, a contextual information retrieval system, which uses the current email message and a multi-tiered user model to retrieve relevant content and make it available in a sidebar widget embedded in the email client. The system employs a dynamic retrieval strategy to conduct automated contextual search across multiple information sources including the user's hard drive, online documents (wikis, blogs and files) and other email messages. It also presents the user with information about the sender of the current message, which varies in detail and degree based on how often the user interacts with this sender. We conducted a formative evaluation which compared three retrieval methods that used different context information: current message plus a multi-tiered user model; current message plus a single-tiered, aggregate user model; and lastly, current message only. Results indicate that the multi-tiered user modeling approach yields better retrieval performance than the other two. In addition, the study suggests that dynamically determining which sources to search, what query parameters to use, and how to filter/re-rank results can further improve the effectiveness of contextual IR.

Author Keywords
Contextual information retrieval, context-sensitive search, user modeling, email, inbox.

ACM Classification Keywords
H.5.2 [Information Interfaces and Presentation]: User Interfaces—User interaction styles, User-centered design; H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval—Query formulation, Search process.

1. INTRODUCTION

In the 14 years since Whittaker and Sidner coined the term "email overload" [30] we have seen neither a decrease in the amount of time required by knowledge workers to manage their inbox [6], nor a significant redefinition of the functionality of email clients. There have been visions and proposals published for reinventing email [e.g. 1, 14], but market-dominant products such as Lotus Notes and Microsoft Exchange remain essentially unchanged: 'overloaded' communication tools [8]. This may be because email users are comfortable and invested in their existing tool [2].

As if in testimony to the problem of overload, in 2007 we were introduced to the concept of "email bankruptcy" when prominent business people and authors declared that they would forego replying to any of the existing thousands of unread messages in their inbox [19]. The recent announcement of Google Priority Inbox was described as a response to email overload which has continued to worsen [21].

Further compounding the problem is the fact that knowledge workers, whose work requires synthesis and re-use of information that was previously created [9], must leave the context of the email application in order to find the information necessary to reply to messages. When the user leaves the current context to gather information (from an online resource or local hard drive) it can cause a delay of hours, or even days [1]. This interruption places the response at risk of falling through the cracks and it increases the cognitive burden – and therefore the feelings of email overload – on the user since this task is now extending over time and must be kept track of.

To help email users easily access relevant information, we developed ICARUS (**I**nformation in **C**ontext: **A**utomated **R**etrieval **U**ser **S**ervice), which automatically surfaces relevant content without requiring users to leave the inbox. The core technology of ICARUS is contextual information retrieval (IR) informed by user models and email analytics. The system's user modeling component builds a model for each user based on the email and calendar content associated with that user. The contextual IR component uses the user model, and the information from the current email message, to determine what content is relevant (e.g. documents from hard drive or online resources, information about sender). The content provided varies based on who the sender is, what the previous interaction has been between the sender and the user, and the topic of the current message. The retrieved information is provided "at the user's fingertips" within the email application so that the user doesn't need to leave his/her current context to gather necessary information.

ICARUS differs from previous work by providing three distinctive features. First, it obtains a comprehensive understanding of the current context by augmenting the content of the current message with a cumulative user model dynamically built from the user's email and calendar content. Second, ICARUS' user model includes representation of the user's interactions with different persons/groups on different topics, which enables the context-sensitive information needs to be identified with a finer granularity, leading to higher relevancy of the retrieved content. Third, ICARUS dynamically tailors its retrieval strategy (e.g. what sources to search, what types of documents to retrieve, how to sort multiple retrieval results), which improves the usefulness of the retrieved results given the limited screen real estate at the interface.

In the balance of the paper, we first present related work in Section 2. Then we use a set of scenarios in Section 3 to illustrate how ICARUS provides relevant information at the fingertips of email users. Section 4 provides an overview of ICARUS, followed by detailed descriptions of the system's key components in Sections 5 and 6. Finally we describe the formative evaluation in Section 7 and conclude the paper in Section 8.

2. RELATED WORK

ICARUS builds on prior work that focused on creating user models from email content [7, 11, 13, 18, 24]. For example the work described in [18] incorporates sender and recipient information when inferring topic distributions from email content. Most of the prior work with user models built from email has had as a goal either social network analysis [18] or email triage with assistance for related tasks (e.g. organizing email messages into folders, determining recipients during email composition, identifying tasks to create a to-do list). Furthermore, most of these models are either not topic-based [11, 13, 24] or encode a single topic distribution without incorporating person-specific information [7].

Our work is related to a number of efforts in context-sensitive information retrieval. There are systems that exploit the user's interest profile [12], query history [26, 27, 28], or interaction with the search results [15, 25, 28, 29] to better understand the user's search context. These systems are driven by explicit user queries, and rely on a static user interest profile or the history of the user's search activities to perform query expansion and/or result re-ranking or filtering. In contrast, ICARUS automatically infers the user's information needs and formulates queries while the user is reading or writing an email. In addition, the user model that ICARUS uses for understanding the current context is built from the user's activities through emails and calendar.

ICARUS is also related to implicit query systems or content recommendation systems that automatically retrieve and recommend relevant content to users while they are reading or composing information (e.g., web browsing [3, 5, 23], emailing [10, 23], and paper writing [5]). Such systems usually use limited context information (e.g. keywords and other content features extracted from the current web page, email, or document) to generate queries. Furthermore, these systems employ a static retrieval strategy that doesn't vary with context. In comparison, ICARUS exploits a more comprehensive context with finer-grained user modeling based on the user's email history to determine user information needs, and uses a dynamic retrieval strategy to tailor to different information needs in various situations.

3. USE CASE

In this section, we provide use case scenarios for an email user named Alice to illustrate how ICARUS aids the user by automatically providing relevant information at her fingertips, and how the retrieved content is tailored in different situations.

Scenario 1

Alice receives an email from Bob who works on a different team and in a different field. Alice doesn't know Bob and has had no previous interaction with him. He asks about Alice's work on System T and wants to know if it can help his own work. Based on the information extracted from the email (e.g. sender, subject, body) and Alice's user model built from her email and calendar content, ICARUS infers that: 1) Alice is unlikely to be familiar with Bob so she may need some background information on Bob (e.g. job description, location of work), 2) Bob is likely to only be interested in high-level information about the topic without much technical detail, 3) Alice is familiar with the topic and may need access to her local and online documents on this topic when generating a response to Bob. ICARUS then automatically formulates queries based on the sender name and the topic keywords of the current message to retrieve relevant content from multiple sources. The retrieved information includes a detailed profile of Bob, related files from Alice's hard drive (e.g. presentations, screen shots), and links to related online documents and resources such as the demo videos, and wiki entries for System T. The information is displayed within the email application, so Alice doesn't have to leave her inbox.

Scenario 2

Alice receives an email from Carol. Carol and Alice both work in the area of text analytics but on two different teams. Alice has had infrequent interaction with Carol. Carol wonders if they can find synergy between their projects and collaborate on a prototype that showcases the technologies developed by each team. For this case, ICARUS automatically retrieves: 1) brief profile description (as a reminder) of Carol, 2) documents on Carol's work, and 3) documents and links for Alice's work which could be shared with Carol, as well as related previous emails between Alice and Carol. Greater details in documents (e.g. .doc, .pdf) are included as well as presentations and videos.

Scenario 3

Alice receives an email from Dan, who works on the same team as Alice. The email is addressed to the whole team and contains an update of the text processing problem the

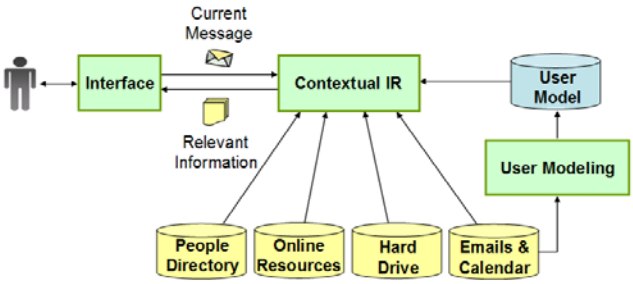

Figure 1. ICARUS system architecture

team has been working on lately. In this case, ICARUS determines that 1) Alice doesn't need photos and background information on the team members, 2) the information Alice may need is on a very specific topic and most likely contains technical details, and 3) recency of information becomes more important because of frequency of interaction. Thus ICARUS first retrieves from Alice's user model topics specific to Alice's previous interaction with the team, and uses these to infer the topic of the current email. The keywords associated with the inferred topic are then used to generate queries for retrieving recently updated relevant documents from online resources (e.g. new entries/documents added to the team wiki) as well as Alice's hard drive, and recent related emails exchanged between some or all of the team members. Messages that belong to the same email thread are grouped and presented together. For Alice's convenience ICARUS also provides basic contact information of the team members (e.g. phone number).

4. SYSTEM OVERVIEW

Here we provide an overview of ICARUS, starting with its system architecture, followed by its user interface.

System Architecture

Figure 1 illustrates the system architecture, which consists of three main components. The *user modeling* component, described in more detail in Section 5, includes data structures and algorithms for computing and representing user information such as how often the user interacts with other individuals or groups through emails and meetings and what topics they discuss.

The *contextual IR* component (detailed in Section 6) takes the user model and the current email message as input and outputs information relevant to the current context. Depending on the user information needs inferred from the user model and the current message, the relevant information can come from multiple sources including the user's email database, online communities, wikis, blogs, file-sharing tools (or information repositories), the user's hard drive, and enterprise directories with people's profiles.

Finally, the *interface* component communicates with the contextual IR component to provide the context information extracted from the email application and surface the retrieved information to the user.

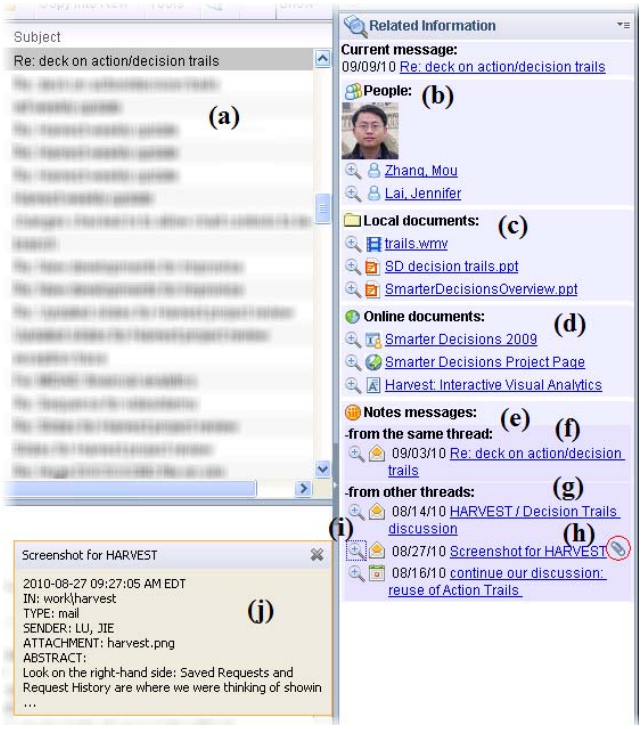

Figure 2. ICARUS user interface

User Interface

Figure 2 shows the interface of ICARUS developed as a Lotus Notes 8 sidebar widget plug-in. When the user clicks on an email message in his/her inbox or any of the user-created folders (Figure 2a) to read the message or generate a response to it, ICARUS automatically conducts contextual IR based on the information extracted from the message and the user model. The retrieved information is displayed in the widget. Entries are organized into sections by type/source. The section of "*People*" (Figure 2b) includes the profiles of people related to the current message. "*Local documents*" (Figure 2c) are retrieved from the user's hard drive, while "*Online documents*" (Figure 2d) are retrieved from intranet applications and repositories (e.g., communities, wikis, blogs, file-sharing tools, paper/patent databases, etc.) using APIs from an enterprise search system ([22]). Finally, "*Notes messages*" (Figure 2e) includes the messages retrieved from the user's email database, with subsections to distinguish between the messages from the same email thread as the current message (Figure 2f) and the messages from other related threads (Figure 2g). An attachment or link indicator occurs next to a message (Figure 2h) when the content of this message contains one or more file attachments or embedded web links.

Each entry displayed in the widget is hyper-linked. Thus a single mouse click opens the corresponding profile, document, or message with its associated application (e.g. a Notes view, or a browser page). The user can also click on the zoom-in icon next to an entry (Figure 2i), to display a slider (Figure 2j) with key attributes for this entry (e.g., subject, abstract, attachments, and embedded links of an email).

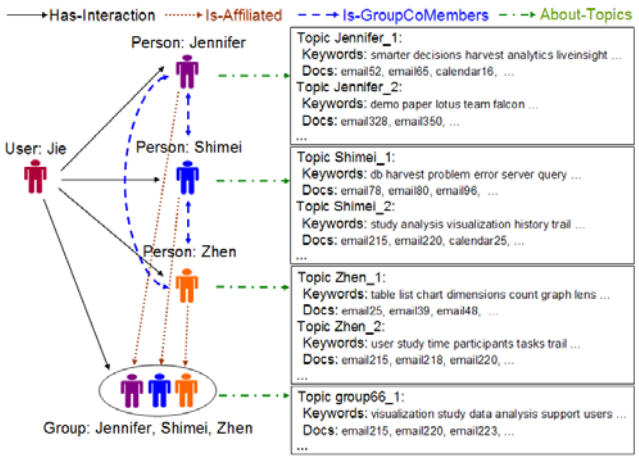

Figure 3. Illustration of ICARUS user model

5. USER MODELING

The user's model is created from his/her email and calendar content, encoding information that can help ICARUS determine the user's dynamic, context-sensitive information needs. In this section, we explain our approach to user modeling in two steps. First, we describe how a user's information is represented in his/her model. Next, we present the process of dynamically creating a user model.

User Model Representation

A user model encodes multiple tiers of information to represent the user's information at different granularities. There is *basic* information extracted from email and calendar messages, including textual content such as subject and body, as well as metadata about the attached files, the embedded web links, and the persons as email senders/receivers and meeting participants. *Aggregate* information is created by grouping basic information. Email and calendar messages are grouped into threads by subject. Persons are grouped based on their associations with email and calendar messages. *Derived* information such as interactions and affiliations link each person or group that has had interaction with the user to the corresponding set of basic and aggregate information.

Based on the basic, aggregate, and derived information encoded in a user model, *multiple* topic models are created and stored in the user model as well. Each topic model is created based on the aggregate content of the user's interaction within a specific *interaction scope*. An interaction scope can be an email thread with multiple messages, the interaction with a single person/group, or the user's overall interaction with other people as a whole. A topic model associated with a thread represents the topics discussed in this thread. A topic model associated with a person or group reflects the user's topics of interest specific to this person or group. A general topic model derived from the aggregation of the user's interaction with all others represents the user's overall areas of work. The use of multiple topic models enables a user's topics of interest to be represented at a finer granularity, which yields more accurate inference of the user's context-sensitive information needs, thus resulting in higher relevancy of the retrieved content.

Each topic model contains a set of topics. Each topic is associated with two types of information: the probability of a word given this topic for all the words, and the probability of this topic given a message for all the messages in the associated interaction scope. The former probability provides a list of representative keywords that describe the topic, while the latter provides a list of messages that are strongly associated with the topic. Topics are derived from content based on statistical language models (see next section for more details).

Figure 3 illustrates the information encoded in a user model. The user is linked to all the persons she has had interaction with through emails and calendar, and the groups of persons derived from the lists of email recipients and meeting participants ("*Has-Interaction*"). Each person is linked to the group s/he is affiliated with ("*Is-Affiliated*"). There are also group co-member relations among persons in the same group ("*Is-GroupCoMembers*"). Each person or group is linked to the topic model associated with this person or group ("*About-Topics*"). Particularly, Figure 3 shows three topic models specific to Jie's interaction with Jennifer, Shimei, and Zhen respectively, and a topic model specific to Jie's interaction with them as a group.

User Model Creation

To build a user model, the system first processes the user's email and calendar content to extract basic information and creates a *basic index* with Lucene [16], which includes email and calendar messages, file attachments, web links, and persons. Then the system further processes basic information to compute aggregate and derived information, and derives topics to create a *model index* that contains threads, groups, interactions, affiliations, and topic models. In addition, the system computes a set of statistics such as each term's frequency of occurrences in all the messages, and gathers the organizational and social relationship (using an enterprise search system [22]) between the user and each person in the basic index. Such information is stored in the model index as well. Both indices are updated periodically to incorporate information from new messages.

Creating Multiple Topic Models

As mentioned earlier, a user model contains multiple topic models each associated with a specific interaction scope. For each interaction scope, the system first creates a text document for each message within this scope by concatenating the subject and body of the message. "Non-content" information (e.g. signature, disclaimer, and text formatting markup) is removed using heuristic rules. Then a topic model is created by deriving topics from the collection of these text documents using the Latent Dirichlet Allocation (LDA) method [4]. More specifically, through statistical inference, LDA estimates two topic-related probability distributions: $\Phi(w, t)$: topic-specific word distribution which describes the probability of a word w given a topic t, and

Label	Calculation
interaction_freq_person *interaction_freq_group*	$\min(f, T) / T$ where: f is the total number of email and calendar messages included in the user's interaction with the person/group associated with the current message, T is the interaction frequency threshold (empirically set to 50 for a person and 10 for a group).
interaction_recency_person *interaction_recency_group*	$1 / (log(d_{current} - d_{most\_recent} + 1) + 1)$ where: $d_{current}$ is the current date, and d_{most_recent} is date of the user's most recent interaction with the person/group.
relationship_strength_person	r_p whose value is: 1.0 if the user and the person p has a managing/managed relationship or if they have the same direct manager; 0.8 if they have co-authored any paper/patent; 0.6 if they are co-members of the same online community/wiki, friends of the online social network, or have shared files with each other; 0.2 for other online relationships such as tagging/tagged.
best_match_group_score	$\max_{g'}(match(g, g'))$ where $match(g, g')$ is calculated using Formulas 1−3.
msg_is_threaded	1 if the message belongs to an email thread of multiple messages, and 0 otherwise.
topic_familiarity_user	The user's familiarity with the topic inferred from the current message, which is equal to the probability of the inferred topic.
topic_familiarity_person	$\min(n_o, T_o) / T_o$ where: n_o is the number of online documents authored by the person associated with the current message (e.g. sender), retrieved using the top 5 keywords associated with the inferred topic, and T_o is the topic familiarity document threshold (set to 10 empirically).
best_match_topic_model	The topic model with the best degree of match between its interaction scope and the interaction scope of the current message.

Table 1. Context factors

$\Theta(d, t)$: document-specific topic distribution which describes the probability of a topic t given a document d.

We use the LDA implementation in MALLET [17] to estimate the main parameters of LDA, i.e. $\Phi(w, t)$ and $\Theta(d, t)$. Other LDA parameters are determined empirically (e.g. the number of topics in each topic model is proportional to the number of messages included in the associated interaction scope). To ensure the quality of the derived topics, a person/group-specific topic model is created only when there are at least 10 messages, and a thread-specific topic model requires at least 5 messages. Topic models can be periodically updated to incorporate new messages overtime.

6. CONTEXTUAL INFORMATION RETRIEVAL

The contextual IR component receives information extracted from the current email message such as sender, subject, and body, and goes through a series of steps to retrieve information relevant to the current context. First, the system selects one or more topic models to best infer the topic of the current message. Second, the system computes the values of a set of *context factors* (Table 1), such as whether the current message is threaded, the frequency and recency of the user's interaction with the sender, the user's organizational/social relationship with the sender, and the user's degree of familiarity with the topic of the current message. Based on such context factors, the system next determines its retrieval strategy in the form of a set of *retrieval factors* (Table 2). Guided by the retrieval factors, the system then formulates queries to search one or more sources, processes the results returned from them, and creates a presentation of relevant information to display to the user. Below we describe the key steps of this retrieval process in detail.

Selecting Relevant Topic Models

With multiple topic models representing the user's granular topics of interest within different interaction scopes, the problem of which topic model(s) to use for inferring the topic of the current message arises. Our solution is to calculate the degree of match between the interaction scope of each topic model and that of the current message, and select the model(s) whose matching score(s) is/are above a threshold.

For a thread-specific interaction scope, the degree of match between the associated topic model and the message is 1 if the model represents the same thread that this message belongs to and 0 otherwise. For a person-specific interaction scope, if the message is an incoming email, its degree of match against the topic model associated with the sender is 1. If the message is an outgoing email, its degree of match against the topic model associated with any of the direct (i.e., not copied on) recipients is 1. All other person-specific topic models receive a matching score of 0. For a group-specific interaction scope, the degree of match between the associated topic model and the message is computed based on the degree of overlap o among members of the model's group g' and those of the message's group g (which includes all the people associated with the message, i.e. sender, direct and indirect recipients), as well as the average normalized co-membership strength s between members of these two groups (Formulas 1–3):

$$match(g, g') = 0.5 \times o(g, g') + 0.5 \times s(g, g') \quad (1)$$

$$o(g, g') = \frac{\#common\_members(g, g')}{(|g| + |g'|)/2} \quad (2)$$

$$s(g, g') = \frac{\sum_{m \in g} \sum_{m' \in g'} \frac{\#common\_groups(m, m')}{\max_{m'' \in g'}(\#common\_groups(m, m''))}}{|g| \times |g'|} \quad (3)$$

where # denotes "*the number of*" and $|\bullet|$ denotes the size of the group.

The topic models whose matching scores are above a threshold (empirically set to be 0.6) are considered relevant topic models. If none of the above topic models have a

Label	Value Set
profile_type	{*min*, *short*, *long*}
doc_author	{*user*, *sender*, *both*, *any*}
doc_type	{*basic*, *detailed*, *all*}
msg_filter	{*thread*, *group*, *person*, *none*}
sort_by	{*relevance*, *date*}

Table 2. Retrieval factors

score greater than the threshold, the general topic model (derived from all of the user's email and calendar content) is selected and returned.

Inferring the Topic of the Current Message

ICARUS infers the main topic discussed in the current email message by identifying the best topic that can explain the current message from the selected relevant topic models. For each relevant topic model, the system computes the topic distribution of the current email message based on the probability distributions encoded in the model [4, 17]. The topic with the highest probability is considered the best topic within this topic model. Then the system chooses the topic with the highest overall probability among the best topics from all the relevant topic models as the inferred topic of the current message.

The inferred topic t has two types of information associated with it: a list of keywords, and a set of messages. The keywords are the top-ranked words that have the highest topic-specific word probabilities in $\Phi(w, t)$. They are used in determining the query terms for retrieval. The messages are the top-ranked documents that have the highest document-specific topic probabilities in $\Theta(d, t)$. They are useful for filtering the retrieved messages during result processing.

Determining Retrieval Strategy

The goal of contextual information retrieval is to provide relevant information to satisfy the user's context-sensitive information needs. Because the user's information needs vary depending on the message and context, the system dynamically determines its retrieval strategy for selecting sources to search, formulating queries, and processing search results in order to optimize the relevance and usefulness of the information it provides to the user within the limited space of the interface.

To help the system infer the user's context-sensitive information needs, we define a set of *context factors* that represent different aspects of the current context. Each context factor has a normalized value between 0 and 1 so that the values of different context factors have the same scale. A retrieval strategy is represented with a set of *retrieval factors* that imply different user information needs. Tables 1 and 2 list the context factors and retrieval factors used by the system. The process of determining a retrieval strategy is thus the process of mapping from the values of context factors to those of retrieval factors.

We employ an instance-based mapping algorithm [20] due to its advantages in flexibility and extensibility over the rule-based algorithm with hard-coded rules. An *instance* is consisted of two parts: situation and decision. Using a vector representation where each dimension of the vector corresponds to a particular context factor, a *situation* represents a particular value combination of context factors. A *decision* uses a similar vector representation to encode a particular value combination of retrieval factors. The instance-based algorithm matches the situation s derived from the current context against each situation s' of all example instances, and returns the decision associated with the instance that has the best matched situation (i.e., with the smallest distance d calculated using Formula 4):

$$d(s, s') = \sum_c |v(c, s) - v(c, s')| \quad (4)$$

where c denotes a context factor listed in Table 1, $v(c, s)$ and $v(c, s')$ denote the values of the context factor c in the situations s and s' respectively, and $|\bullet|$ denotes the absolute value. We choose a summation-based metric over the standard cosine similarity metric due to its simplicity, interpretability and low computational cost.

Currently the example instances pre-loaded in the system are determined based on empirical observations. We plan to explore methods that enable the system to dynamically create and modify example instances based on the user's interaction behavior at the interface.

Formulating Queries

The queries used for retrieving people's profile information (from an enterprise employee directory) are created based on the name and email address of each person associated with the current message as a sender or direct recipient.

The queries used for document retrieval (from the user's email database, hard drive, online resources) are automatically generated based on the prominent words contained in the current message as well as the keywords associated with the inferred topic of the current message. Specifically, to determine query terms, the system first extracts all the words from the subject and body of the current message (excluding stopwords and "non-content" information such as signature, disclaimer, and text formatting markup) to create a *context vector* of terms. The weight of each term is its frequency of occurrences in the content of the current message multiplied by its inverse document frequency (the number of existing messages from the user model that contain the term).

Next the system obtains the list of the 20 top-ranked keywords associated with the inferred topic of the current message. Because the weights of these keywords generated by the topic inference algorithm are not compatible with the term weights in the context vector, they cannot be directly combined. Instead, the system simply increases the weight of a context vector term by a fixed percentage if it occurs in this topic keyword list to boost terms that are representative of the inferred topic. An empirically determined value of 50% is used.

Finally, the system selects the 5 top-ranked context vector terms as the primary query terms. The 5 top-ranked topic

keywords that are not stopwords, person names, or primary query terms are added as expansion query terms.

For selected sources that support other search parameters in addition to query terms, the system generates the values for these parameters based on the retrieval strategy determined. For example, if the value of the "*doc_author*" retrieval factor indicates that the system should restrict the retrieved online documents to be those authored by the user/sender, the user/sender's identity is included in the query for searching online resources that support filtering results by person. If a source allows the query to specify how the retrieval results should be sorted (e.g. by date or by relevance), the system includes the specification based on the value of the "*sort_by*" retrieval factor.

Processing Search Results

The top-ranked search results returned from the selected sources (based on source-specific cutoff thresholds) are further processed prior to being displayed in the interface. The means by which the results are processed depends on the values of the relevant retrieval factors, including "*profile_type*", "*doc_type*", and "*msg_filter*".

To process a result returned from a directory of people's profiles, the system checks the value of "*profile_type*" and extracts the required information. A "*min*" profile requires basic information about a person such as name, email, and phone number. It serves the purpose of providing the user easy access to the contact information of people whom the user interacts with frequently. A "*short*" profile requires a person's photo in addition to the standard contact information, which can help refresh the user's memory about someone s/he has interacted with infrequently. A "*long*" profile requires the most information about a person, including contact information, photo, geography, and job description. It is the most comprehensive type of profile but also requires the most space real estate at the interface. Therefore, it is only necessary for people with whom the user has little previous interaction.

For each document in the search results returned from the user's hard drive or online resources, the system determines its type and discards the document if the type is not included in the value of the "*doc_type*" retrieval factor. Currently we distinguish between two types of documents: "*basic*" for documents targeted at a general audience, and "*detailed*" for documents geared towards an audience with relevant background. The type of a document is determined with a simple algorithm that classifies a document based on the extension of its name and its source. For example, a file with a "*.ppt*", "*.avi*", "*.wmv*", etc. name extension or a starting web page of an online community/wiki is considered "*basic*", while a file with a "*.doc*", "*.xls*", "*.ps*", etc. name extension or a web page from an online paper/patent database is considered "*detailed*". If the need for more granular document types and a more sophisticated classification algorithm arises in applications, the system can easily be extended to incorporate them.

For messages retrieved from the user's email database, the system filters them based on the value of the "*msg_filter*" retrieval factor. If the value is "*thread*", the system discards the messages that do not belong to the same thread as the current email message. If the value is "*person*" or "*group*", the system discards the messages that are not associated with the person or group of the current message. To further improve the relevance of the messages to be presented to the user, the system also discards the messages that do not have a strong association with the inferred topic of the current message.

Finally, the system performs duplicate removal, and removes redundancy from threaded messages by discarding any message whose content is already fully contained in the later messages of the same thread.

7. EVALUATION

We designed and conducted a user study to evaluate ICARUS. In this section, we first describe our evaluation methodology, followed by an analysis of the study results.

Methodology

We focused on evaluating two key aspects of ICARUS: multi-tiered user modeling and dynamic retrieval strategy.

Baseline Methods

To evaluate the effectiveness of our user modeling approach, we compared three retrieval methods that used different context information: current message plus ICARUS' multi-tiered user model ("*msg+multi-model*"); current message plus a single-tiered user model, which was created by aggregating all of the user's email and calendar content ("*msg+single-model*"); and lastly, current message only without any user model ("*msg+no-model*"). All three methods dynamically determined their retrieval strategies using the approach described in Section 6.

We also compared retrieval methods using three different strategies which varied the retrieval factors: "*dynamic*": ICARUS' dynamic retrieval strategy; "*related*": a static retrieval strategy that always sorted the results by relevance and returned top-ranked documents of all types as well as top-ranked messages; and "*individual*": a static retrieval strategy similar to "*related*", except that additional query parameters and filters were applied to return related documents or messages only if they were from the user or the individuals associated with the current message. We chose these two static strategies as baseline because they are most widely used by today's retrieval systems. For this comparison, the same multi-tiered user model was used in determining context information.

Study Design

We recruited five people within our organization for the study. All of them use Lotus Notes everyday for their work-related email exchange and meeting schedule. We installed the ICARUS sidebar widget on a Lotus Notes client. All participants used this Notes client to access their own password-protected email databases. When a participant logged into Notes with his/her credentials, the widget created a

user model based on the messages from this participant's inbox, calendar entries, and any email folders selected by the participant using the widget's menu. Each participant was asked to select 10 messages from his/her inbox (or email folders). We advised the participants to select messages about a variety of topics from people with whom they had various degrees of interaction frequency. The participants then provided relevance judgment for the retrieved results for each message. The result set consisted of the profiles for all the people associated with the message, and all of the top-ranked documents and messages retrieved by all the methods described in the previous section. A rank threshold of 10 was used, so each method contributed a maximum of 10 entries to each result type (i.e. document, message) in the aggregate result set. Each person's profile had three labels for the participants to choose from: "*min*", "*short*", and "*long*". The participants were asked to select the label that corresponded to the profile type they expected to see for this person. Each document or message in the result set also had three labels: "*relevant*", "*somewhat relevant*", and "*not relevant*".

We collected additional feedback from the participants through a questionnaire at the end of the study. The questionnaire included a set of statements which the participants were asked to rate using a Likert scale that ranged from 1 for "*strongly disagree*" to 7 for "*strongly agree*". For example, one of the statements was "*I find it helpful that the profile displayed for a person varies based on whether I have had previous interaction with this person and how frequent the interaction has been.*" The questionnaire also included open-ended questions such as "*what features of the tool do you like the best*", "*which features do you find most difficult to understand/use*", and "*what additional features would you like to see in the system*" to further collect user comments and suggestions.

Evaluation Metrics

To measure the degree of relevance among the retrieved documents and messages, we treated the participants' relevance judgment as the ground truth in calculating the standard *precision* (percentage of retrieval results that were relevant) and *recall* (percentage of relevant documents and messages retrieved) at rank 10 for each <selected message, method> pair. "*Somewhat relevant*" was counted as 0.5 in calculation. Note that because the relevance judgment was only performed on the result set that aggregated up to 10 top-ranked documents or messages from each method, the recall here measured a method's ability in providing relevant content within its top-ranked results, not its overall ability in retrieving relevant content. Based on the calculated values of precision and recall, we also computed *F-measure* for the results. F-measure is the harmonic mean of precision and recall (Formula 5), which is a commonly used measure to evaluate the overall effectiveness of retrieval.

$$F\text{-}measure = 2 \times precision \times recall \,/\, (precision + recall) \quad (5)$$

To measure the effectiveness of a retrieval strategy in providing the proper level of detail for the retrieved profiles, we calculated the average distance between the system-determined profile types and those indicated by the participants. The distance was 1 (and –1 in the reverse direction) from "*long*" to "*short*" and from "*short*" to "*min*". The distance was 2 (and –2 in the reverse direction) from "*long*" to "*min*".

Results

In this section, we summarize the results of the study.

Evaluation of Multi-Tiered User Modeling

Figure 4 shows the average values of precision, recall, and F-measure at rank 10 for three retrieval methods with different context information, averaged over the results for the 50 selected messages in total. The figure indicates that the method of "*msg+multi-model*" outperformed the methods of both "*msg+single-model*" and "*msg+no-model*" in all three metrics. The relative performance increase of "*msg+multi-model*" over the other two methods was 11%-18%, which was statistically significant ($p<.05$), demonstrating the promise of our user modeling approach and still leaving room for further improvement.

The performance of "*msg+single-model*" relative to "*msg+no-model*" indicates that for contextual IR in the typical enterprise email environment we studied, using a single-tiered user model to help determine context degraded the retrieval performance rather than improving it. This was because when using a single-tiered user model to infer the topic of the current message, the inferred topic was often broad and covered the user's interaction with multiple people for different purposes. In consequence, some keywords associated with the topic introduced noise when selected as query terms and ended up doing more harm than good. By

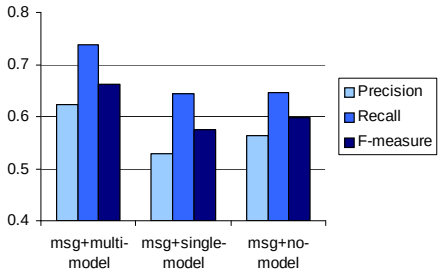

Figure 4. Evaluation results of methods using different context information.

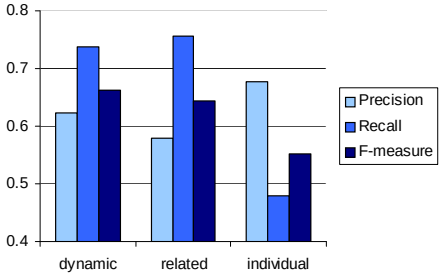

Figure 5. Evaluation results of methods using different retrieval strategies.

contrast, the multi-tiered user model contains multiple topic models representing the user's interactions with different individuals or groups separately. Furthermore, only topic models associated with the people involved in the current message are examined for topic inference. As a result, the inferred topic is more specific and accurate, and the keywords from this topic are more helpful to retrieval.

Evaluation of Dynamic Retrieval Strategy

The average distance between the profile types dynamically determined by the system and those indicated by the users was 0.36, which shows that the system was largely accurate (with slight "overestimation") in the level of detail needed for a profile. In comparison, the average distance values between the fixed profile types of "*long*", "*short*", or "*min*" and what the users indicated were 1.45, 0.45, and –0.55 respectively. These results indicate that our dynamic strategy, which determined the profile type of a person based on the user's interaction frequency and organizational/social relationship with this person, yielded better performance than either fixed strategy in providing an appropriate level of detail for the retrieved profiles.

Figure 5 compares the system's performance when using three different retrieval strategies. The "*related*" retrieval strategy, which retrieved content from all available sources solely based on the estimated relevance, had the highest recall. When compared with the lowest recall from the "*individual*" retrieval strategy, the result suggests that documents or messages relevant to an email message could be from people *not* associated with the email. However, querying and filtering results based on the individuals associated with the email is still needed in some situations, as indicated by the highest precision from the "*individual*" retrieval strategy and the lowest precision from the "*related*" strategy. By contrast, the "*dynamic*" retrieval strategy used information such as the user's interaction frequency and recency with the sender as well as their familiarity with the topic of the current message to dynamically determine which sources to search, what query parameters to use, and how to filter/re-rank results. As a result, it balanced between precision and recall and yielded the highest F-measure value, thus showing an improvement in overall retrieval effectiveness when compared with static retrieval strategies.

Questionnaire Results and Comments

We analyzed the subjective data collected through questionnaire and computed the average answer ratings for the Likert-scale questions. All of the users agreed (with an average rating of 6, which corresponded to "*agree*") that they found it helpful to have relevant content automatically provided to them in the sidebar of the email client. The users all agreed (with an average rating of 6) that the retrieval results presented to them provided a good coverage of the types of content (e.g. people's profiles, files from hard drive, web pages from online resources, email and calendar messages) they found helpful when reading or writing an email. However, feedback from the users indicates that not all content types were considered equally important/useful. All five users commented that relevant email and calendar messages were most important. The majority of the users appreciated the display of the profiles about related people, but one user didn't find such profile information particularly valuable. One user emphasized the importance of relevant local files, while the other users didn't single out this content type. Three users thought that although content from online resources could be relevant and sometimes helpful, most of the time they would not need this type of content when processing an email. They suggested moving the section of online documents to the bottom below other sections, or allowing a user to "turn off" the retrieval from online resources when not needed. Two users would like to see sections listing the embedded links and file attachments extracted from related messages, but one user worried that a lot of irrelevant links (e.g. links to personal web pages in signature blocks) could be included, and the rest of the users remained neutral on this subject. Lastly, two users considered it helpful to include relevant content from chat transcripts of instant messaging tools.

All users gave positive responses (with an average rating of 6) when asked if they found it helpful that the content provided varied based on context information such as who the sender was, what the interaction had been between the user and the sender, and the topic of the current message.

In terms of presentation of information, the users somewhat agreed (with an average rating of 5.2) that they could easily determine the relevance of a result solely based on its label created from URL/title/subject. They felt that it was easy sometimes (when the label was informative) but quite difficult at other times. Regarding the "zoom-in" feature that displayed key attributes in a slider (Figure 2j), three users found it very helpful for determining the relevance of a result and/or gathering the needed information without having to read the full content, but the other two users rarely used it during the study since they felt that accessing the full content of a result was equally easy and more straightforward. The users unanimously agreed (with an average rating of 6.6, between "*agree*" and "*strongly agree*") that it was very helpful for them to be able to access the full content of a result with a single mouse click and read the content within the email client. However, most of the users didn't like opening a new view or page of the email client every time the full content of a result was displayed. Instead, they preferred a single dedicated view/page or content preview pane for displaying content.

For the open-ended questions, when asked about what features of ICARUS they liked the best, the answers included automatic retrieval of relevant people's profiles and emails, ease of accessing the full content of retrieval results within the email client, and dynamic tailoring of retrieval results based on the user's email interaction history in addition to the content of the current email message. Regarding what feature was the most difficult to understand/use, the common complaint was that it was difficult to judge the rele-

vance of an online document whose content was not familiar to the user, or a long email message of which only a part contained relevant content. The users desired system support (e.g. keyword highlight) to help them quickly understand why a document was retrieved and/or easily locate the relevant portion of the content.

8. CONCLUSION AND FUTURE WORK

In this paper, we present a contextual information retrieval system, called ICARUS, which automatically provides relevant content at the fingertips of an email user based on the current email message the user is working with and a user model created from his/her email and calendar content. We have focused our work on two core technologies. First, creating a multi-tiered user model, which encodes basic, aggregate, and derived information, as well as topic models for the user's interaction with others at the level of persons, groups, and email threads. Such a fine-grained user model helps the system identify more accurately the topic of the current message and the user's information needs. Second, ICARUS employs a dynamic retrieval strategy that uses context information (e.g. how often the user interacts with the sender, the user's familiarity with the topic of the current message, etc.) to determine which sources to search, how to formulate queries, and how to process retrieval results to optimize the value of the information displayed within the limited space of the interface. Our study results demonstrate that contextual retrieval using ICARUS' multi-tiered user model yields better performance than retrieval that uses a single-tiered user model or the current message only without a user model. The evaluation also shows that ICARUS' dynamic retrieval strategy can further improve the overall effectiveness of contextual IR in enterprise email.

We plan to explore methods that can further improve the accuracy of topic inference for the current email message and the quality of the automatically formulated queries. We would also like to investigate how a user's interaction behavior during email processing (e.g. what information is used when the user is reading/writing his/her emails) can be utilized for automatic personalization of dynamic retrieval strategy and fine-grained customization of contextual retrieval to suit different users. Lastly, we would like to improve the interface design of ICARUS with the goal of further enhancing the users' email experience.

REFERENCES

1. V. Bellotti, N. Ducheneaut, M. Howard, and I. Smith. Taking email to task: The design and evaluation of a task management centered email tool. In *CHI'03*.
2. V. Bellotti, N. Ducheneaut, M. Howard, I. Smith, C. Neuwirth. Innovation in extremis: Evolving an application for the critical work of email and information management. In *DIS'02*.
3. D. Billsus, D. Hilbert, and D. Maynes-Aminzade. Improving proactive information systems. In *IUI'05*.
4. D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.
5. J. Budzik and K. Hammond. Watson: Anticipating and contextualizing information needs. In *ASIS'99*.
6. L. Dabbish and R. Kraut. Email overload at work: An analysis of factors associated with email strain. In *CSCW'06*.
7. M. Dredze, H. Wallach, D. Puller, and F. Pereira. Generating summary keywords for emails using topics. In *IUI'08*.
8. N. Ducheneaut and V. Bellotti. Email as habitat: An exploration of embedded personal information management. In *Interactions*, September-October; 8(5): 30–38, 2001.
9. S. Dumais, E. Cutrell, J. Cadiz, G. Jancke, R. Sarin, and D. Robbins. Stuff i've seen: A system for personal information retrieval and re-use. In *SIGIR'03*.
10. S. Dumais, E. Cutrell, R. Sarin, and E. Horvitz. Implicit queries (IQ) for contextualized search. In *SIGIR'04*.
11. A. Faulring, B. Myers, K. Mohnkern, B. Schmerl, A. Steinfeld, J. Zimmerman, A. Smailagic, J. Hansen, and D. Siewiorek. Agent-assisted task management that reduces email overload. In *IUI'10*.
12. S. Gauch, J. Chaffee, and A. Pretschner. Ontology-based personalized search and browsing. *Web Intelligence and Agent System*, 1(3-4):219–234, 2003.
13. J. Goodman and V. Carvalho. Implicit queries for email. In *CEAS'05*.
14. B. Kerr and E. Wilcox. Designing remail: Reinventing the email client through innovation and integration. In *CHI'04*.
15. R. Kraft, F. Maghoul, and C. Chang. Y!q: contextual search at the point of inspiration. In *CIKM'05*.
16. Lucene. http://lucene.apache.org.
17. A. McCallum. MALLET: A machine learning for language toolkit. http://mallet.cs.umass.edu, 2002.
18. A. McCallum, X. Wang, and A. Corrada-Emmanuel. Topic and role discovery in social networks with experiments on Enron and academic email. In *Journal of Artificial Intelligence Research*, 2007.
19. M. Musgrove. E-mail reply to all: 'Leave me alone'. http://www.washingtonpost.com/wp-dyn/content/article/2007/05/24/AR2007052402258.html
20. S. Pan. A multi-layer conversation management approach for information seeking applications. In *ICSLP'04*.
21. J. Perez. Google to roll out e-mail prioritizing feature in Gmail. http://www.reuters.com/article/idUS110492765220100831
22. I. Ronen, E. Shahar, S. Ur, E. Uziel, S. Yogev, N. Zwerdling, D. Carmel, I. Guy, N. Har'El, S. Ofek-Koifman. Social networks and discovery in the enterprise (SaND). In *SIGIR'09*.
23. B. Rhodes and P. Maes. Just-in-time information retrieval agents. *IBM Systems Journal*, 39(3–4):685–704, 2000.
24. R. Segal and J. Kephart. MailCat: An intelligent assistant for organizing e-mail. In *AGENTS'99*.
25. X. Shen, B. Tan, and C. Zhai. Context-sensitive information retrieval using implicit feedback. In *SIGIR'05*.
26. X. Shen and C. Zhai. Exploiting query history for document ranking in interactive information retrieval. In *SIGIR'03*.
27. M. Speretta. Personalizing search based on user search histories. In *CIKM'04*.
28. J. Teevan, S. Dumais, and E.Horvitz. Personalizing search via automated analysis of interests and activities. In *SIGIR'05*.
29. Z. Wen, M. Zhou, and V. Aggarwal. Context-aware adaptive information retrieval for investigative tasks. In *IUI'07*.
30. S. Whittaker and C. Sidner. Email overload: Exploring personal information management. In *CHI'96*.