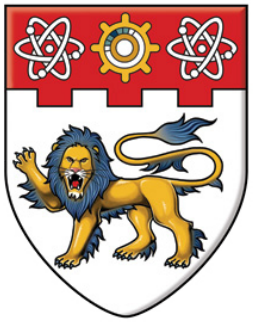


NANYANG TECHNOLOGICAL UNIVERSITY

Remembrance Agent: from Theory to Applications

Xiaogang Han

School of Computer Engineering

2014

Remembrance Agent: from Theory to Applications Xiaogang Han 2014

Remembrance Agent: from Theory to Applications

Xiaogang Han

Xiaogang Han

School of Computer Engineering

A thesis submitted to the Nanyang Technological University

in partial fulfillment of the requirement for the degree of

Doctor of Philosophy

2014

Acknowledgments

First, I would like to thank my PhD supervisor Dr. Miao Chun Yan. She gives consistently good advice and guidance, and allows an ideal level of freedom in pursuing ideas and research topics. Moreover, the discussions with her pushed me to think from many different angles of the research.

A special thanks to Dr. Shen Zhiqi, Dr. Mei Jianping, Dr. Liu Junfa, and Dr. Luo Xudong for their influential discussion and in depth feedback. I also would like to thank Dr. Zhang Huiliang, Chen Chaohai, and Dr. Cai Yundong at the Centre for Advanced Information Systems for their invaluable feedback on the drafts of the thesis.

Finally, thanks to my wife, Zhao Wenjing, for her constant, unwavering, and encouraging support.

Contents

List of Figures

List of Tables

Abstract

In the information age, people have difficulty to process and organize large amount of incoming information from online social network sites due to our limited memory capacity. The problem arise when such information is needed as reference to make decisions. People may not be able to figure out proper clues to retrieve them or may even forget their existence at all. Remembrance agents (RAs) are a type of agents which proactively retrieve and present the user's historical documents based on the his/her current context. Memory modeling is one of the fundamental components for RAs. Most of the existing RAs adopt memories with restricted semantic interpretation, which limit content relevance as the primary criterion for retrieval. Building user interest models based on the user documents stored in RA's memory is very critical to inferring user preferences. Through user interest modeling, potentially interesting documents can be recommended to the user whenever needed. To date, little work has been done on integrating user interest model into RAs. A user's documents are naturally distributed in multiple information sources. Each of the sources is of different characteristics and significance. This contradicts the assumptions made by most of the existing work that the user documents are collected from a single information source.

To address these challenges, this thesis proposes a novel RA model, *Episodic and Semantic memory based Remembrance Agent (ESRA)*, inspired by cognitive memory theories. The ESRA incorporates the functionalities of episodic memory (EM) and semantic memory (SM) of the human memory into the agent's memory structure. EM stores documents pertaining to the events experienced by the user in chronological order. SM stores the concepts which are extracted from the documents in EM and their corresponding aggregated activation scores. By explicitly modeling EM and SM and their functional interactions, we can capture both the episodic information and categorical semantic information to improve the memory retrieval performance. Moreover, with an ontological user interest profile constructed from the aggregated categorical concepts in SM, the ESRA is able to provide accurate personalized recommendations for the user. In addition, we propose an approach for unifying a user's historical documents from multiple information sources, and develop a source-aware retrieval algorithm subsequently to complement the user's cognition ability with the proposed ESRA in various scenarios.

To study how the proposed ESRA can help the user in real world applications, we

have incorporated the ESRA into three different application domains. The first application is the modeling of user interests on folksonomy services and its application for personalized search. The second area of application is tweets recommendation based on ontological user interest profiling on microblogging services. And the third area is to provide assistance to students in remembering things in virtual learning environments. Extensive experiments have shown that the proposed ESRA can enhance the user's cognitive ability by acting as an external memory and significantly improve the historical documents retrieval performance for the user.

Keywords: Remembrance Agent, memory consolidation algorithms, classification, user interest modeling, personalization

Chapter 1

Introduction

In this information age, we are overwhelmed with the large amount of information in our dynamic daily activities such as learning new knowledge, processing incoming messages (e.g., E-mail), and accessing to information on the Web. We try to absorb these information and store them into our long-term memory. The information, together with the context in which it was obtained, are constantly reorganized in our memory to become our knowledge about the world, and form the basis for us to learn new knowledge, make decisions, and solve problems.

Human memory is limited in capacity and ability to process and organize large scale information. Although we have browsed all the incoming information sequentially, only selected ones draw our attention and enter into our long-term memory. How well these information are stored and organized in our memory is based on the following factors:

- its relevance to our current context;
- the existing knowledge in the semantic memory;
- our topic of interests.

We often meet difficulties when we try to harness knowledge stored in our memory to solve problems in our task-oriented working environment. For example, people use a variety of social networking services to collect and organize Web information for possible future reference in a different context. We use social bookmarking services to collect the resources we are interested in and rely on search engines to find out the possible solutions to a problem. Such information is valuable in the user's current context but could also potentially be reused in future contexts. However, when such contents are actually needed to reply to a social network post, the user may not be able to retrieve them without proper clues or may even completely forget that they existed in the first place.

As an example, suppose a user A is interested in the topic "open science". He has collected and shared articles/videos on this topic on different social network websites. At a later time, one of his friends, user B, raised a conversation on a social network site (see Fig 1.1) with the topic about the recent trends of "open science". The conversation consists of the initial post by the user B and a list of existing replies by other participators. User A is quite interested in the conversation and would like to express his opinions. In order to compose the reply message, he may need to refer to the resources that he had collected on "open science" in previous contexts. However, due to the limited memory capacity, he may not be able to recall exactly where to find the resources. Even worse, he may not remember he had collected the previously-seen resources.

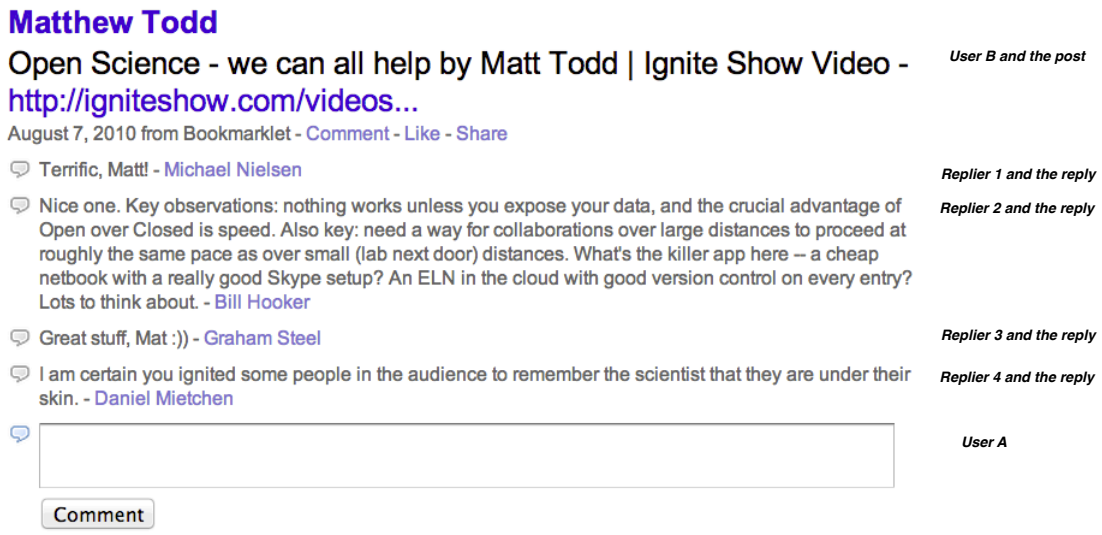


Figure 1.1: A conversation on a social network website

The problem illustrated in the above example is quite common for information workers in the social age. A conversation and its associated social participatory information represents a wide range of social context of a social network user. Such social context includes the questions on the question-answering websites, the conversations in the Email systems, and the social updates on the Microblogging systems. It is desired to build intelligent systems which acts as the external memory to store the information as collected by individual user and continuously provides just-in-time help when such information are needed.

In order to achieve these goals, it is very important for such systems to understand the user's social environment, current task, and the user's interests. Many technologies have been investigated to help us make connections between the user's

resources in different contexts. In particular, intelligent agents have been used as personal assistants [1] to recommend or filter resources for us. *Remembrance agents*, which pro-actively retrieve and present relevant information based on a user's current context in an accessible yet non-intrusive way [2] [3], were introduced to help us retrieve historical information by acting as our external memory. However, there are still many shortcomings and challenging issues awaiting to be addressed.

1.1 Research Problems

The first problem with the existing remembrance agent is that the role of different types of memories and the interaction between different memories were not well addressed. Most of the existing remembrance agents adopt restricted memory structure for information storage and ignore the rich information embedded in the user information interaction behaviors. Such approaches limit the features with which the information can be retrieved. However, based on cognitive memory theories, human have different types of memories and each has its role in information storage. To act as human external memory, the agent architecture should emulate the different storage mechanisms of human memory.

The human memory has an ability to consolidate event-based information into semantic knowledge based on prior world knowledge, which is the basis of advanced inference. As most of the existing remembrance agents don't model the world knowledge (they consider the incoming resources from the user's collection as the only information source), these agents have very limited classification or understanding abilities to connect the user's knowledge with the external world knowledge. Therefore, the information retrieval in those remembrance agent models cannot utilize the semantic information deduced from various world knowledge taxonomies.

It is expected that remembrance agent not only provides the functionality of automatic contextual retrieval, but also support exploration of the information stored in memory explicitly. For example, the user may occasionally like to browse the agent's memory to find specific events or topics, based on certain temporal-spacial or topical clues. However, as the existing remembrance agents store information as a list of resources indexed by their content, the temporal-spatial or semantic information of the individual resources are always not available, which means that the functionality is not well supported.

The purpose of introducing remembrance agent is to enhance human user's memory. It is supposed to continuously keep track of the user's information access history and infer the user's preferences to support autonomously retrieving information based on the user's local context. However, most of the existing remembrance agents

fail to model the user's knowledge level or preferences as a type of memory, which, turns out to be an important source of evidences for inference.

The problems described above are not trivial. The task of sorting and storing user historical information into different types of memory (knowledge base), detection and interpretation of local contexts for autonomous retrieval, and modeling user interests based on diverse information sources are all significant challenges. Resolving those challenges is the main theme of this thesis.

1.2 Research Objectives

In this thesis, we try to address these problems by building a remembrance agent model based on cognitive memory theories [4]. As the agent's goal is to equip the user with an external memory, it is required that the agent model emulate as closely as possible he structure and cognitive capacities of human memory. The agent needs to be able to work in the user's application environment to encode, store and retrieve information for the user.

The objective of this thesis is to develop a general remembrance agent model for personal information assistance. It incorporates the research in cognitive memory theories, knowledge extraction, intelligent agent, and information retrieval. The remembrance agent model is very general and is applicable to various interactive learning environments, personal information management systems, and lifelong user information management systems.

1.3 Thesis Contributions

This thesis makes four main contributions. First, we proposed a general remembrance agent to cope with personal information overload problem. Second, we developed three memory consolidation algorithms to organize and consolidate information in the agent's memory. The next contribution is a context-aware retrieval algorithm to automatically extract relevant information from the agent's memory to support information reuse. Finally, we applied the agent model to real world problems with three case studies. We now give more details of each of these contributions below.

1.3.1 A General Remembrance Agent Model

The thesis developed a general remembrance agent model based on cognitive memory theories. Its knowledge base is built based on the user's diverse information

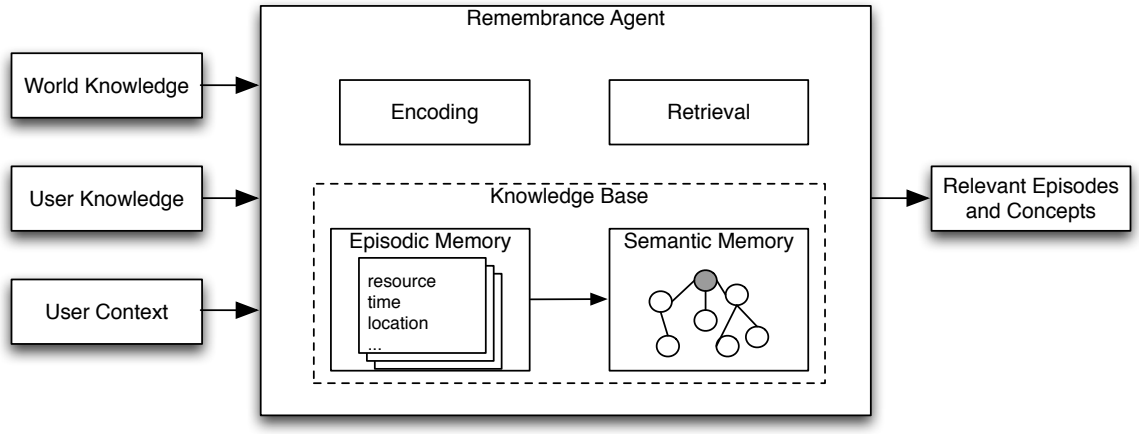


Figure 1.2: The proposed Remembrance Agent in which the episodic memory and semantic memory are explicitly modeled

consumption, which means that information sources are personalized to the user. On the other hand, the *resource-centric information organization* in the agent's episodic memory is consolidated into a *concept-centric knowledge organization* in the agent's semantic memory, which forms the user profile. The user profile and local context are used together for the retrieval. Secondly, the distinction of episodic memory and semantic memory mimics simplified human level cognition and information storage structure, which enables both episodic and semantic exploration of the historical information. Furthermore, multiple retrieval metrics are introduced to support different retrieval methods based on the nature of the information need at hand. Finally, the user profile in the user semantic memory is maintained and adapted to the absorption of new information.

1.3.2 Memory Consolidation Algorithms

We study the text based information organization in the agent's memory and developed three memory consolidation algorithms to extract information from the agent's episodic memory and update them into semantic memory. These algorithms are designed to classify normal text, social bookmarks, and short text, respectively. The information updated into semantic memory helps building the user knowledge/interests profile as a concept ontology. Based on the ontological user profile, various semantic inference can be made with regards to the user's preference and interests.

1.3.3 Memory Retrieval from Multiple Information Sources

Based on the retrieval framework specified in the remembrance agent model, we developed a memory retrieval algorithm in which the user's diverse information sources are modeled. As the resources in each of the user's multiple information sources have their own specific characteristics and significances, our ranking algorithm takes advantages of these features and unifies the resources into a weighted ranking scheme. The ranking algorithm is evaluated on a real world social network dataset. The results show that the proposed algorithm can help user in retrieving historical document distributed in diverse social network sites with high accuracy.

1.3.4 Case Studies

We applied the remembrance agent model to three real-world problems, in which the agent proactively pointing out the connections between what the user is doing now and things the user has done in the past. In the folksonomy based user interests modeling application, the user's bookmarks are regarded as information sources and the concepts consolidated in SM are used to model the user's semantic interests, which were evaluated via a personalized search system. In the tweets recommendation on microblogging platforms application, consolidation algorithms for short texts were developed and the knowledge in both the episodic memory and semantic memory are utilized to recommend interesting tweets to individual user. In the virtual learning environments application, the remembrance agent continuously monitors the player's gaming behaviors and organizes the information in its knowledge base. When the player is detected being stuck with certain problems, the remembrance agent will proactively make connections between the problems they are stuck at and the information objects that they've interacted with in the past.

1.4 Thesis Organization

The rest of the thesis is organized as follows. Chapter 2 reviews related work and points out the main problems with existing work. Chapter 3 defines and formalizes the Remembrance Agent model based on human cognition theories. Chapter 4 studies the memory consolidation algorithms. The contextual retrieval framework is depicted in Chapter 5. The application case studies are presented in Chapter 6. Finally, Chapter 7 summarizes our finding and outlines future work.

Chapter 2

Literature Review

This thesis proposes an agent based model to solve the information overload problem. In this chapter, we first describe the kind of information overload problems in Section 2.1. The existing work on remembrance agents are reviewed in Section 2.2. A family of agents that are relevant to remembrance agent is personal assistive agents, a brief survey is given in Section 2.3. As the proposed remembrance agent model is a form of cognitive architecture, we describe the existing cognitive architecture and cognitive memory theories in Section 2.4.

2.1 Information Overload on Social Media

The amount of resources on the social media platforms nowadays is so large that the retrieving of relevant information based on the user's context is becoming more and more difficult. On the other hand, the rapid growth of new social network services and virtual worlds has produced new information channels. For an individual user, it requires a lot of attention to find relevant information from the unmanageable information sources. To address these challenges, lots of personalization, recommendation, and filtering techniques have been proposed. The main objective of these techniques is to adapt relevant information to users based on their short-term and long-term perferences [5]. Remembrance agent is one of the solutions to the information overload problems.

2.2 Existing Remembrance Agents

Just-in-time information retrieval agents have been studied extensively in the past. The first one was proposed by Bush, called Memex [6][7], which is a theoretical proto-hypertext computer system in which an individual stores all his books, records, and

communications. Additionally, an individual can create annotations and associative trails on the documents in the library. Ideally, the device would act as human memory extender and recall those trails when needed. The original conceptual Memex machine is depicted in Figure 2.2. However, Memex was never built because, according to Bush [8], the idea was before its time.

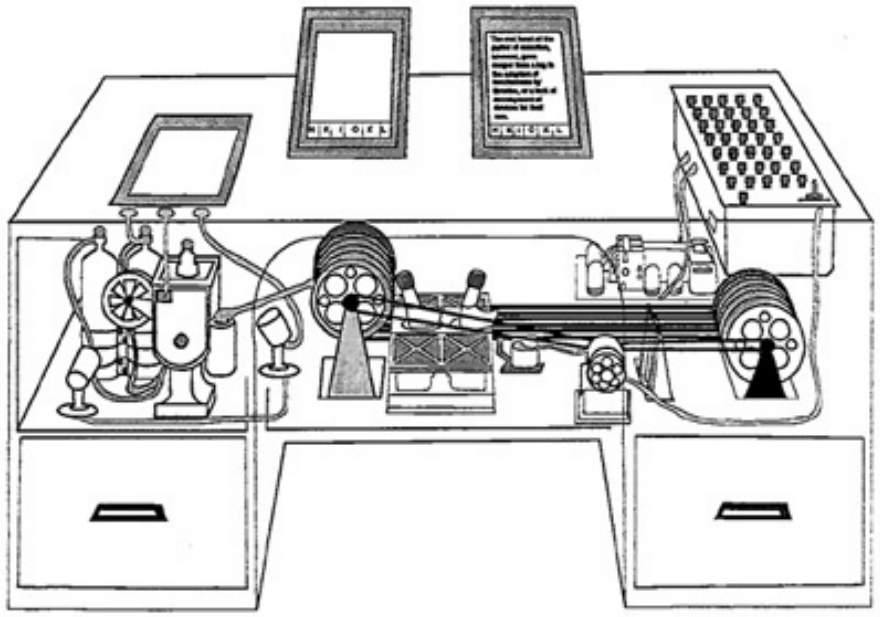


Figure 2.1: Memex

Since *Forget-me-not* [9] exposed the possibility of *intimate computing* (supporting human memory using user's context) back in the early nineties, the notion of augmenting human memory (or memory prosthesis) has become an important application domain in the research field of context-aware computing. In essence, the approach adopted by Forget-me-not is to utilize the user's context to index personal episodic information. However, the approach in *Forget-me-not* does not explicitly support the user's interests or preferences using any form of user model [10]. The approach in *Forget-me-not* does not model the user's current context to automatically recall relevant historical information, but requires the user to explicitly search within incomplete contexts related to certain historical activities.

Rhodes defined the term just-in-time information retrieval (JITIR) agent and implemented three JITIR agents. The first one is Remembrance Agent (RA) [11], which is a program that augments human memory by displaying a list of documents possibly relevant to the current user context. Unlike most information retrieval systems, the RA run continuously and proactively to detect the current user context and index historical information without user intervention. Its unobtrusive interface

enables a user to pursue or ignore the RA's suggestions as desired. The second one is a wearable RA [12][2][13], which took physical context such as location and people in the immediate area into consideration, but he concluded that the location and people in the area are poor distinguishing features for note-taking RA. The third one [3] is an automatic just-in-time information system for the Web. The RA here does not have knowledge of the user's interest or whether the user has previously seen a particular suggestion.

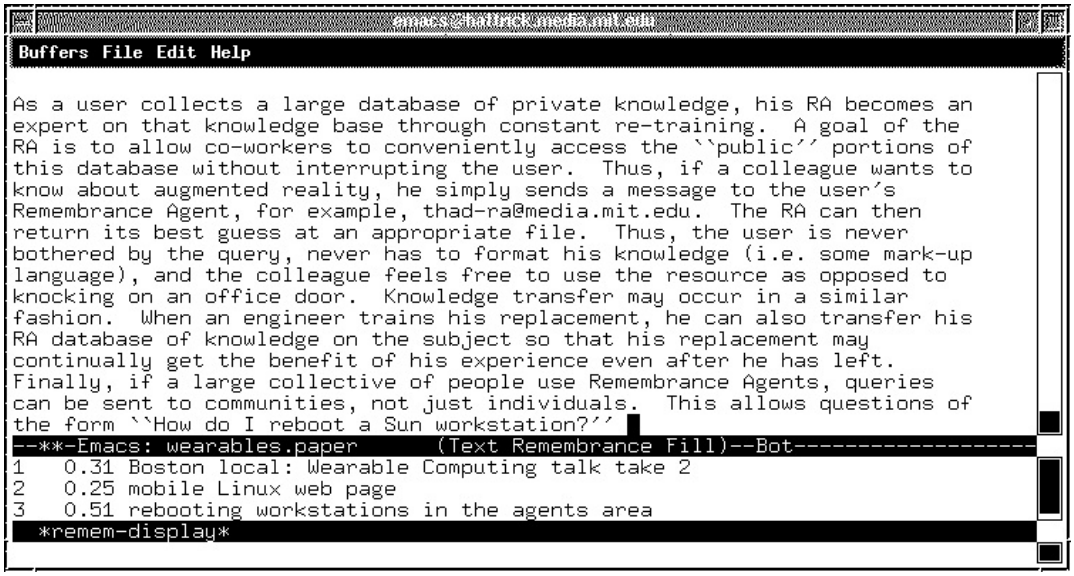


Figure 2.2: Remembrance Agent as a contextual information retriever — retrieving relevant e-mails from the e-mail archives of an individual based on current editing texts

JITIRs are characterized as a type of *contextual information retrieval* agent that provides information retrieval and context based information filtering. Watson [14] [15] uses a simple and explicit task model to interpret user actions (interaction with everyday applications) in order to anticipate a user's information need by querying Internet information sources for information. Letizia [16] assists users in browsing the *Web* by suggesting and displaying relevant web pages based on user interests. The difference between Letizia and Rhodes's RAs is that the RAs *remember the past* - show the user relevant material that they have already seen, whereas Letizia *remembers the future* - shows relevant material not yet seen.

In recent years, more and more work on RAs are being researched. For general RA design, Xia et al. [17] extends Dong Engelbart's human intellect augmentation framework for the design of RA in the domains of memory, motivation, decision making, and mood. For the memory domain, they uncovered the potential of developing artifacts that assist in the encoding and recall of closely related episodic and

semantic memory, while the details are not outlined. Eyal et al. [18] investigated the amount of information users can consume and the information delivery modes and studied how to reduce disruption and enhance conversation quality in face-to-face conversations.

RAs have also been applied to support email composition [19] and email reply prediction [20]. They offer information relevant to the current email by extracting information from historical emails. ACLD [21] extends RA with speech-based search interface using automatic speech recognition to retrieve relevant content. One of its main application scenarios is to unobtrusively display meeting related documents for demonstration purpose. Moreover, an interactive tabletop system was developed as a tool for assisting memory recall to augment the everyday work patterns of small collaborative groups [22]. SidePoint [23] is a peripheral panel which shows information relevant to the slide content to support presentation authoring. RAs have also been applied to implicitly providing search results to online social network status message questions [24], integrating Web search into the software development environment [25], and automatically provide context-sensitive learning resources to support the user's learning of complex software [26]. Finally, the growth of mobile devices has inspired researchers to design RAs for mobile and wearable devices to help the user obtaining relevant information when needed [27][28].

The primary problem with the RAs is that most of them are designed from the information retrieval perspective, in which ranking is the only relevance measure to retrieve information. However in the real world, it is necessary to organize the documents [29][30] to help user browsing through the documents. The episodic memory and semantic memory in our proposed agent model support meta-data relevance and topic relevance, respectively, to address the problem. Another problem is that user interests are not explicitly modeled in JITIRs. Previous research focuses on presenting information based on common knowledge corpus rather than personal knowledge repository of the particular user [2]. We addressed the problem with concept based user interests modeling in semantic memory.

2.3 Personal Assistive Agents

The group of personal assistive agents are analogical to RAs in terms of personal information reuse. CALO [31] is a proactive personal assistive BDI agent that can aid a human in managing and performing tasks. The agent can perform four categories of activities: Act directly, Act indirectly, Collect information, and Remind, notify, ask. But CALO define neither a context nor user model, and the domain is the daily events of a person. Autominder [32] is a cognitive orthotic system intended

to help older adults adapt to cognitive decline and continue the satisfactory performance of routine activities. Autominder uses a range of AI techniques to model an individual's daily plans, observe and reason about the execution of those plans, and make decisions about whether and when it is most appropriate to issue reminders. Autominder utilizes individual user interaction histories to make predication, thus the model can not be easily applied to other scenarios. PDS [10] is a personal digital secretary with a framework integrating user models and context-awareness. PDS defines context models and user models in terms of the issues concerned in context-aware systems: data acquisition, coupling to applications, representation, and period required for data acquisition.

Lifelogger such as MyLifeBits [33] was designed to be a personal database for everything, including scanned and encoded archival material, e.g., articles, books, music, photos, and videos as well as everything digital, e.g., office documents, email, digital photos. It later evolved to include web pages, phone calls, meetings, room conversations, keystrokes and mouse clicks for every active screen or document, and all the 1–2 thousand photos that SenseCam captures every day. Lifelogger systems focus on recording aspects of personal life. However in reality recording is not a serious problem. By contrast, the ability to retrieve specific bits of materials based on local context is more important.

2.4 Cognitive Memory Theory

2.4.1 Long-Term Memory

In psychology, Long Term Memory is typically divided into declarative memory and procedural memory [34]. The declarative memory in long-term memory consists of semantic memory, which stores facts and rules, representing context independent general knowledge, and episodic memory [35] that stores temporally ordered snapshots of working memory, which can be used for case based reasoning. As defined by Tulving et al. [36], episodic memory refers to the personal experiences and their temporal relations, while semantic memory is a system for receiving, retaining, and transmitting information about meaning of words, concepts, and classification of concepts. The two memory systems differ on the nature of the stored information, and the encoding, storage, and retrieval of information.

Based on the theory of *Experiential learning* [37], personally experienced events are stored in episodic memory and, over time, used to construct generalized knowledge structures in semantic memory. Semantic memory deals with concepts and facts by which you can maintain a mental representation of the world, such as factual knowledge. Episodic memory is the context wherein semantic knowledge

acquisition takes place. As you are engaged in events yourself, you will store the context in which each event happened.

Both semantic and episodic memory provides clues for recall, thereby spreading the activation of interrelated knowledge. Based on semantic network modeling of semantic memory [38], knowledge are stored as network of connections between individual knowledge nodes. The recall of such semantic memory is through the activation of concepts based on semantic relevance [39]. For episodic memory, researchers in psychology have developed episodic or autobiographical memory [40] [41], which concludes that we naturally organize our memories of past events into episodes [40], and that the location of the episode, who were there, what was going on, and what happened before or after, are all strong clues for recall [42] [35]. Although semantic memory is independent of the context in which it was acquired, distinguishing between these two memory units is difficult [36].

Psychology studies [43][44] have found that our memory will fade away after a long period of time and a quantitative retention function can be used to describe how the memory decay over time. The implication is that the memory model design should take memory retention into consideration.

2.4.2 Cognitive Architectures

The research work on cognitive architecture, which develop general computational systems that mimic human cognitive abilities, have done some preliminary work to integrate episodic memory and semantic memory into agent architecture [45][46]. SOAR (State, Operator And Result) is a notable rule based general cognitive architecture [47][48][49][50]. The knowledge base of the original SOAR was a set of production rules stored in a single long-term memory, which was extended with episodic memory [51][52] and semantic memory [53] gradually. The episodic memory stores the experiences extracted from working memory including the context and temporal relationships with other experiences, whereas the semantic memory stores the structures that occur in working memory.

Despite the advances of cognitive architecture research, including SOAR, there remains additional work with regards to the most critical problems with cognitive architecture — memory and learning [54][46]. We address some of these issues within our research scope as follows.

(i) Using loose taxonomy as commonsense knowledge base. The knowledge base, especially world knowledge, in cognitive architecture are usually quite difficult to collect and represent. Although the efforts of the Cyc project [55] have led to the construction of a large scale commonsense knowledge ontology, it

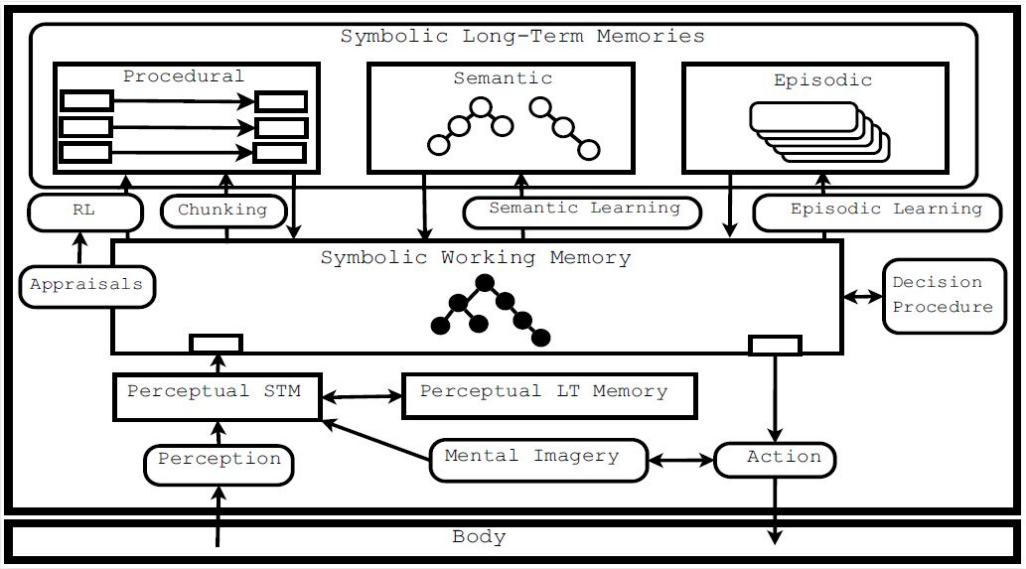


Figure 2.3: Structure of Soar

is criticized for its complexity and limitations. On the other hand, the efforts to build ontology from the existing loose taxonomy, like DBpedia [56] and Yoga [57], are becoming more and more recognized as alternate solutions to build large scale knowledge base. In our work, we also use loose taxonomy like Wikipedia[1] and DMOZ[2] as commonsense knowledge ontology to build the agent's internal knowledge.

(ii) Developing categorization algorithms for memory consolidation. Although the cognitive architecture like SOAR distinguish between different types of memories, the role played by each memory and how different memories interact with each other has not been stressed. The semantic memory and episodic memory in SOAR were assumed to be independent of each other and both of them perceive information from working memory. Based on experiential learning theories [37], we assume that the agent's semantic knowledge about the user is built gradually by categorization and understanding of the information objects encoded in the episodes in episodic memory. To support categorization, the agent model must specify the features of the episodes and learn how the information objects in the episodes can be categorized into the concepts in the world knowledge.

Although we investigate the cognitive architecture in this subsection, the objective of this thesis is **not to build a general cognitive architecture**. In contrast,

[1]http://www.wikipedia.org/
[2]http://www.dmoz.org/

what we developed is a cognitive architecture for remembrance agent. To this extent, we mainly reviewed the *memory* of the cognitive architecture, and left the *learning* part out.

2.4.3 The Role of Prior Knowledge in Memory

Prior knowledge stored in semantic memory plays an important role in storage and retrieval of episodic memory information [36], such as for learning new knowledge [58] [59] and comprehension of scientific text [60]. When processing new information, people always try to fit it into their prior knowledge stored in the memory and find connections between them, which can be regarded as the process of refining existing knowledge. The implication of the Prior Knowledge theory for the development of our remembrance agent are summarized as follows:

- By retrieving prior knowledge from the memory, the remembrance agent autonomously relates new episodic information to the user's existing knowledge, which can improve information processing efficiency,

- Suppose that the knowledge in the user's memory is that user's topic of interests, then by comparing the prior knowledge with the stream of new episodic information, the agent can filter non-relevant or non-interesting information for the user.

- By maintaining a copy of the user's prior knowledge (through long-term monitoring of user information consumption behaviors), the remembrance agent can recommend influential and significant information to the user.

2.5 Summary

In this chapter, we reviewed the related work on existing remembrance agents and cognitive memory theories. We observed the following problems with existing remembrance agents:

- The agents are not modeled based on cognitive architecture. Most of the existing remembrance agents regard the problem as a contextual information retrieval problem. The information sources studied in each of the remembrance agent are task specific, which makes it hard to generalize their model to other application domains.

- The information is usually organized using indexing, rather than mimicking human long-term memory. This approach has some disadvantages. First, *content relevance* is the only criteria for retrieval, the temporal-spatial relations, which expose rich contextual clues, are ignored. Second, there is no support for the exploration of memory from episodic and semantic views, which is always helpful for metadata based retrieving.

- The user's knowledge level and preferences are not explicitly modeled. The resources collected by the user expose rich information about the user and can be utilized to provide advanced reasoning ability to help the user.

We attempt to solve these problems by proposing a remembrance agent model based on cognitive memory theory. By distinguishing episodic memory and semantic memory and developing memory consolidation algorithms to transfer knowledge between these two memories, we can address some challenging issues with existing remembrance agents by taking the following advantages: (1) the knowledge base is built based on the user's cognitive information consuming activities, therefore the temporal and spatial attributes are encoded into the knowledge base, specifically the episodic memory. Such information can serve as trails for retrieval. On the other hand, the *resource-centric episodes* in the agent's episodic memory are consolidated into *concept-centric categorical mapping* in the agent's semantic memory, which forms the user profile. The user profile and local context are used together for the retrieval; (2) the distinction between episodic memory and semantic memory mimics simplified human level cognition and storage of external information, which enables episodic perspective and semantic perspective exploration of the historical information; (3) Multiple retrieval metrics are introduced to support different retrieval methods based on the nature of the information need observed in the user context; (4) the user profile in the user's semantic memory is maintained and adapted to the absorption of new information and user's relevance feedback.

In the next chapter, we will describe the model in details.

Chapter 3

The Remembrance Agent Model

In this chapter, we describe the proposed remembrance agent model in detail. We follow the definition as described in [2]: a *Remembrance Agent* is an intelligent agent that continuously encodes and manages incoming information for individual user, and provides just-in-time information retrieval based on the user's real-time context. Suppose the user collected a large amount of resources, the remembrance agent works both at the back-end, to record what the user has done and automatically organize the resources in a way that emulate human long-term memory, and at the front-end, to monitor the user's task environment to provide just-in-time information help by retrieve relevant resources in its memory.

To formalize the remembrance agent model, we borrowed some ideas from the cognitive mechanism of human long-term memory in psychology and cognitive science [35][61]. A diagram of long-term memory structure is shown in Figure 3.1. In this figure, long-term memory is divided into declarative memory and procedural memory. Procedural memory is responsible for knowing how to do things, i.e., how to ride a bicycle, whereas declarative memory is responsible for knowing facts and experiences. Declarative memory is further divided into episodic memory and semantic memory. Episodic memory stores the episodic events that one experienced in the past, whereas semantic memory stores the facts and knowledge about the world. How these two types of memories function and interact has been studied extensively in physiology science [4][62]. Our remembrance agent is based on declarative long-term memory which implements a computational model of the episodic memory and semantic memory as its knowledge base. Therefore, we denote the proposed remembrance agent as *Episodic and Semantic memory based Remembrance Agent (ESRA)*.

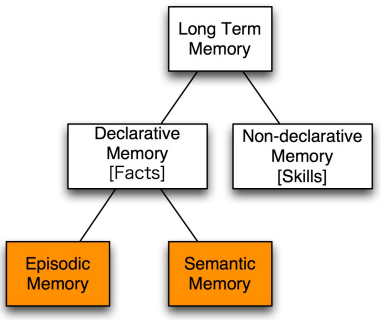


Figure 3.1: Human Long-Term Memory

3.1 Overview

The proposed computational ESRA model is largely based on the physiological studies of human memory [35][61][4][62]. These studies have found out that the context-specific information entering the memory is first encoded into EM. The repeated associations in EM are consolidated into SM to form the general knowledge of the subject over time. The information stored in the memory can be retrieved by either of two types of memory retrieval process: semantic retrieval of relevant knowledge node in the semantic network, and episodic retrieval of relevant episodes from chronologically ordered episode collection.

The ESRA model is shown in Figure 3.2. The agent acts as external memory for individual user, records the user's information consumption behaviors, and is always ready to proactively make connections between what the user has done in the past and what the user is doing now. The input to the agent includes the predefined world knowledge and the behaviors of individual user. The output from the agent is the actively retrieved knowledge which might be helpful to the user's current context.

To handle such information, the agent is designed with four core elements: the encoding module, the knowledge base module, the retrieval module, and the adaption module. The encoding module is responsible of extracting user behaviors and converting those behaviors into internal event representation which can be stored in the knowledge base. The storage module maintains the knowledge about the world and the real-time information about the user. The agent also has two interface modules: the sensor module and the actuator module, based on which it can interact with the environment.

The goal of the agent is to provide proactive information retrieval based on user context and user preferences, constrained by its world knowledge. In order to implement the goal, it will schedule the encoding module to monitor the user behaviors

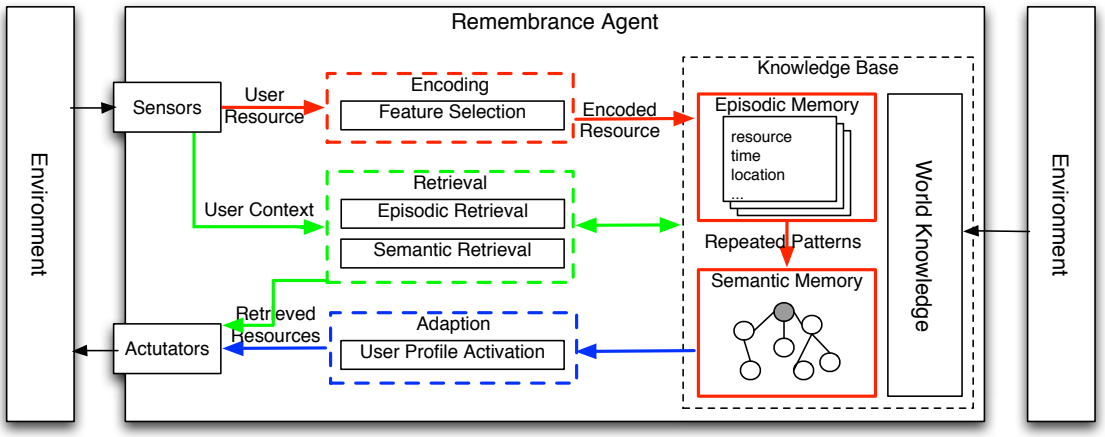


Figure 3.2: ESRA Architecture

and identify those behaviors in which the user interacted with some information objects. The behaviors are encoded as events and stored chronologically in the episodic memory. The user preferences (e.g., user interests, user knowledge level) are inferred by incorporating user events in episodic memory and world knowledge in offline manners subsequently. Such preferences are maintained in the semantic memory. On the other hand, the agent monitors the user's real-time context and retrieves relevant information from the knowledge base to assist the user.

3.2 Internal Structure of the Agent Model

In this section, we describe how the information objects are represented and formalize the essential elements of ESRA.

Definition 3.1 *An **information object**, denoted as d, is an atomic information item from the environment. The set of n detected information objects is represented as $D = (d_1, ..., d_n)$. Each d_i in D is the feature vector for instance i, which describes the attributes of the information object.*

We assume that the user behavior in our research scope is limited to the user's information behavior, in which the user will interact with certain information objects. The input to the agent (experienced by the user as well) is a stream of information behaviors. We represent such information behavior using episode, denoted as e.

Definition 3.2 *An episode, denoted as e, represents an event via which the user u interacts with an information object, d. The **time** at which the user experienced e*

is denoted as t. The physical or virtual ***location*** *at which the user experienced e is denoted as l. Optionally, an episode may also include the set of friends of u who are involved in the event, denoted as $\mathcal{F}_u$. We have:*

$$e = (u, t, l, d, \mathcal{F}_u).$$

The episodes are stored in the agent's knowledge base, specifically, the episode memory.

Definition 3.3 ***Episodic Memory*** *of the agent, denoted as EM, is part of the agent's knowledge base in which episodes are stored chronologically. We have:*

$$EM = \{e_1, e_2, ..., e_{|EM|}\}, t_1 < t_2 < ... < t_{|EM|}.$$

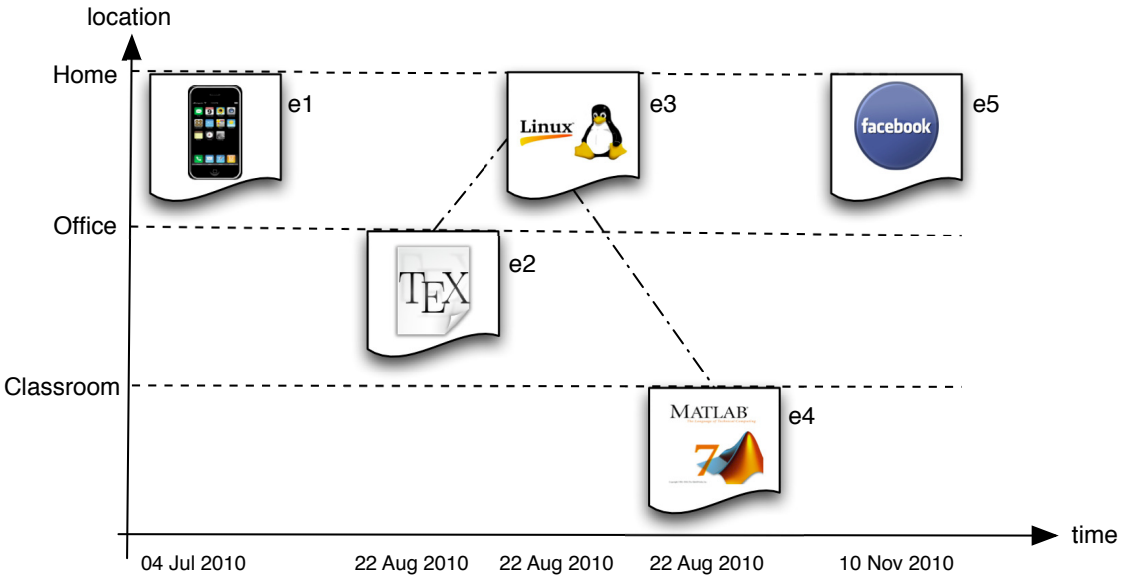


Figure 3.3: The sequential episodes in EM

An example of chronologically ordered episodes in EM and their the temporal-spatial relationship is shown in Figure 3.3. Suppose the user read five articles over some time period at different locations. Each of these articles are of different topics. We regard each of the five reading behaviors as an episode. In this figure, the x-axis labels denote the timestamps and the y-axis labels denote the locations. The episodes, e_1, e_3, e_5, are of the same location, *home*, whereas the timestamps of e_2, e_3, and e_4 are quite near to each other compared with other episodes. Such latent geographical and temporal relations among the episodes exposes rich information about the semantic relevance between the information objects embedded in them.

The semantic memory, which we will talk about later, is the place where those semantic relations are stored.

Before defining semantic memory, we need to formalize the agent's world knowledge memory. Because the generation of information in the semantic memory relies on the knowledge from the world knowledge memory.

Definition 3.4 ***World Knowledge Memory****, denoted as WK, is part of the agent's knowledge base which stores the agent's knowledge about the world (the knowledge about the external environment of the agent). The agent is assumed to be initialized with WK. WK is represented as a graph $G = (C, E)$ comprising a set of concepts C and a set of edges E. Each concept c_i, denotes as $c_i = (topic, D_{c_i})$, represents an entity of the world, in which topic denotes the description of c_i and D_{c_i} denotes the set of information objects belonging to c_i. Each edge in G represents the relation between two concepts.*

Definition 3.5 ***Semantic Memory****, denoted as SM, is part of the agent's knowledge base in which information objects are extracted from the episodes which they are attached to and mapped onto the concepts in WK with consolidation function f. We aggregate the scores of each of the mapping and get:*

$$SM = f(WK, EM) = \{(c_1, score_1), (c_2, score_2), ..., (c_{|SM|}, score_{|SM|})\},$$

where f is the consolidation function, which will be described in next subsection.

An example of information storage in SM is shown in Figure 3.4. In this figure, there are five mapped concepts in the SM, $(c_1, score_1), (c_2, score_2), (c_3, score_3), (c_4, score_4)$, and $(c_5, score_5)$, where the concepts themselves are organized hierarchically, e.g., both concepts c_4 and c_5 belongs to concept c_2. Under each concept, a list of information objects which are considered semantically related to the corresponding concept.

Definition 3.6 *The* ***Knowledge Base****, denoted as KB, is all the knowledge of the agent, which is composed of EM, SM, and WK. We represent KB as $KB = \{EM, SM, WK\}$.*

3.3 Work Flow of the Agent Model

ESRA interacts with the user and the environment by performing a series of three main operations on the information objects: encoding, storing, and retrieval. In this section, we describe these operations in details.

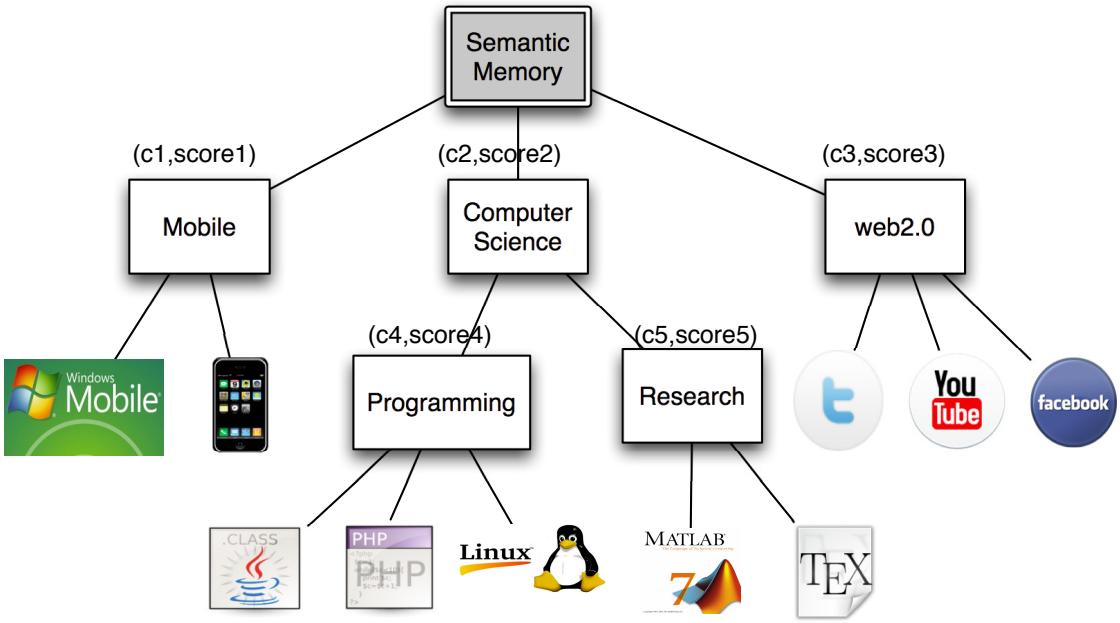


Figure 3.4: The illustration of SM structure

3.3.1 Encoding Information Objects

Definition 3.7 ***Encoding*** *is the process through which an information object, d, are converted into an episode, e, which can be stored in EM:*

$$encoding : d \rightarrow e.$$

The definition of an episode (refer to **Definition 3.2**), $e = (u, t, l, d, \mathcal{F}_u)$, provides the template based on which the *encoding* algorithm can identify the available features accompanied with d. Those features can serve as clues in the retrieval process. Moreover, the decision on when to initialize an encoding need to be considered. By continuously monitoring the user behaviors, the agent identifies new information objects depending on how the user behaviors are characterized.

3.3.2 Storing Information in EM and SM

It is assumed that the agent has two different ways to organize an user's information objects, D. An information object, d, is first stored into EM as an episode, e, the intensity of which is gradually degraded with a *decay* function. Over time, the information object, d, encoded in e, are conceptualized into SM via a *consolidating* function. In this subsection, we describe these two functions in detail.

3.3.2.1 Memory Degradation in EM

Memory decay theory shows that decline happen as new information are encoded into the memory. In the long-term, more recent information are usually more relevant to the user's latest context. Therefore, each e in EM is weighted based on the time when it was encoded.

Definition 3.8 ***Decay*** *is the function to model the retention of episodes in the episodic memory:*

$$decay = decay_0 \cdot exp^{-\lambda\sqrt{t}}. \quad \text{(Eq. 3.1)}$$

We introduce the retention function defined by Rubin et al. [43] as the decay function. The decay function is the exponential in the square root of time with parameters $decay_0$ and λ, as shown in Eq. 3.1. A retention function with $decay_0 = 0.9$ and $\lambda = 0.5$ is shown in Figure 3.5. In the figure, the x-axis denotes the number of days e has been in EM, whereas y-axis denotes the *decay* of the agent's memory on e.

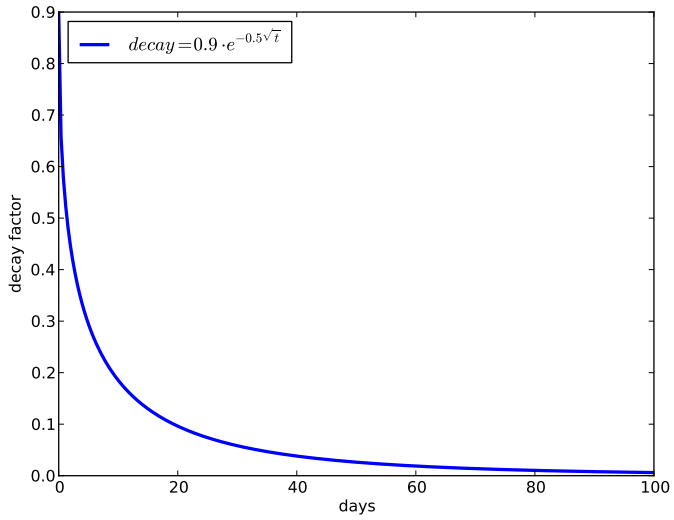


Figure 3.5: Decay strength of the episode in EM

3.3.2.2 Memory Reinforcement in EM

The episodes in EM are related with each other via their semantic or episodic associations. The encoding of a new episode may reinforce some existing episodes with different activation weights. In order to model the interconnection between

the episodes, we introduce episodic reinforcement, which involves adjusting the activation weight of each episode in EM based on the associations tied among them. We use a graph to represent the associations, in which each node is an episode and each edge denotes the association strength between two episodes. The association strengths are calculated using episode reinforcement.

Definition 3.9 ***Reinforcement*** *is a function to calculate the association strength p_{ij} between each pair of episodes, e_i and e_j, in a graph.*

The calculation of p_{ij} depends on the type of episodes involved. In section 5.2.2.4, we will develop a concrete reinforcement algorithm to calculate the importance scores of each post in a collection of social posts.

3.3.2.3 Consolidating From EM to SM

Generalization from specific events is one of the main strength of human memory, which enables us to understand the semantic of the concepts in concrete information objects. We model the process via which the agent learns to map episodes onto WK as *consolidate*.

Definition 3.10 ***Consolidate*** *is a function by which information from EM is extracted and updated into SM. The function takes the episodes in EM as inputs, identifies the information objects in the episodes, and maps them onto the concepts in WK via various semantic matching and classification methods:*

$$consolidate : e \overset{WK}{\rightarrow} (c, score).$$

The consolidating occurs when repeated patterns were found in EM. Such patterns implies the semantic relevance among the information objects in the episodes, which are independent of the specific episodes themselves. The three **Consolidate** algorithms we have developed for different type of information objects will be described in Chapter 4.

As the information objects encoded into EM are specific to individual user, the $(c, score)$ mappings aggregated in SM can be used to represent the user's preferences on those concepts, which forms the *User Profile*.

User profile is derived from SM, which shares the same structure as SM. Therefore, we will use the two terms interchangeably in the rest of the text. In EM, the information objects are embedded in the episodes, which means both the features and temporal-spatial information are recorded. By contrast, in SM, the concepts inferred from individual information object are consolidated into user's knowledge

as a weighted concept network. The temporal-spatial information associated with the information objects are discarded and the information objects are reorganized and categorized into the corresponding concepts. User profile represents the user's knowledge or preference on the world knowledge, including concepts and the relations between them.

Maintaining the consistency of user's topics of interest is an important problem in user modeling research, which facilitates the tasks such as exploration and search of historical episodes, as well as recommending and filtering of new information objects based on the user profile. The use of user profile generated in SM for practical reasoning problems will be discussed in Chapter 5.

3.3.3 Retrieving Information from EM and SM

One of the primary objectives of ESRA is to retrieve relevant concepts and episodes based on the clues in the user's task environment. Here we describe how the agent extracts the local context in the task environment as clues to formalize the queries and uses the queries to perform episodic retrieval and semantic retrieval.

Local context are specified by the user's information behaviors which implies the user's instant information need. As such information behaviors are similar to the information behaviors via which the episodes are defined, we regard local context as a special kind of episode.

Definition 3.11 *Suppose the task environment is characterized by the current user u, time t, location l, information object d, and optionally the set of friends of the current user $\mathcal{F}_u$, then* ***context episode****, denoted as e_c, can be expressed as:*

$$e_c = \{u, t, l, d, \mathcal{F}_u\}.$$

When the user is observed in need of historical information to solve certain problems, all the elements specified in e_c can serve as clues for the retrieval. However, because each of the clues are of different data types, we need to analyze each of them to derive the corresponding relevance measures and integrate these measures into a uniform retrieval model.

3.3.3.1 Episodic Retrieval

The objective of the episodic retrieval algorithm is to identify the most relevant episodes in the EM. The ranking of episode e, given user context e_c, can be measured by the similarity between e and e_c:

$$ranking(e|e_c) = sim(e, e_c).$$

We will next describe the four aspects of relevance we considered to calculate the similarity between e and e_c, namely *document relevance*, *social relevance*, *geographical relevance*, and *temporal relevance*.

Definition 3.12 *The **document relevance** between two episodes, which is denoted as $document\_sim(e, e_c)$, measures the similarity between the information objects, d_e and d_{e_c}.*

We adopt the vector space model [63], which represents text documents as term vectors, to calculate the similarity between two documents. Let $d_e = (w_{1,d_e}, w_{2,d_e}, ..., w_{t,d_e})$ and $d_{e_c} = (w_{1,d_{ec}}, w_{2,d_{ec}}, ..., w_{t,d_{ec}})$, where w_{i,d_e} and $w_{i,d_{ec}}$ are the weights of the ith term in d_e and d_{e_c}, respectively[1]. We have:

$$document\_sim(e, e_c) = sim(d_e, d_{e_c}) = \sum_{i \in t} w_{i,d_e} \cdot w_{i,d_{ec}}. \qquad \text{(Eq. 3.2)}$$

Definition 3.13 *The **social relevance** between two episodes, which is denoted as $social\_sim(e, e_c)$, measures the similarity between the friends (of the user) appeared in e and e_c.*

The set of the user's friends involved in an episode reveals the social aspects of the context, which can serve as social clues for retrieval. We calculate the similarity between the two sets of friends using Jaccard similarity coefficient [64] and obtain:

$$social\_sim(e, e_c) = J(\mathcal{F}_e, \mathcal{F}_{e_c}) = \frac{|\mathcal{F}_e \cap \mathcal{F}_{e_c}|}{|\mathcal{F}_e \cup \mathcal{F}_{e_c}|}. \qquad \text{(Eq. 3.3)}$$

The geolocation of an episode may exposes the type of information behaviors. Therefore, we need to model the geographical similarities between the task environment and each of the episodes.

Definition 3.14 *The **geographical relevance** between two episodes, denoted as $geo\_sim(e, e_c)$, measures the geographical similarity between the location of e and e_c. We have,*

$$geo\_sim(e, e_c) = \begin{cases} 1 & \text{if } l_e = l_{e_c}, \\ 0 & \text{if } l_e \neq l_{e_c}. \end{cases} \qquad \text{(Eq. 3.4)}$$

[1]How the weights of the terms being calculated is quite corpus-specific. Without loss of generality, we do not introduce specific weighting scheme in this chapter. However, it will be addressed in later chapters.

Definition 3.15 *The **temporal relevance** between two episodes, which is denoted as $decay'(e, e_c)$, measures the relative decay rate of e and e_c. Suppose $t_{e_c} > t_e$, we have:*

$$decay'(e, e_c) = decay_{e_c} \cdot e^{-\lambda\sqrt{t_{e_c} - t_e}}. \quad \text{(Eq. 3.5)}$$

Based on the relevance measures defined above, the final contextual retrieval can be formalized as:

$$\begin{aligned} ranking(e|e_c) &= sim(e, e_c) \\ &= decay'(e, e_c) \cdot (\beta_1 \cdot document\_sim(e, e_c) \\ &\quad + \beta_2 \cdot social\_sim(e, e_c) \\ &\quad + (1 - \beta_1 - \beta_2) \cdot geo\_sim(e, e_c)). \end{aligned}$$

The top-k episodes with the highest similarity scores are selected and displayed to the user.

3.3.3.2 Semantic Retrieval

The *consolidate* process maps an episode, e, onto a set of concepts, $\{C_e\}$, which reveals the categorical information of the information object, d, in e. Such categorical information can be used to measure the semantic relevance between different episodes.

Definition 3.16 *The **categorical relevance** between two episodes, denoted as $cat\_sim(e, e_c)$, measures the conceptual similarity between e and e_c.*

We calculate the similarity between two sets of concepts using Jaccard similarity coefficient and obtain:

$$cat\_sim(e, e_c) = J(C_e, C_{e_c}) = \frac{|C_e \cap C_{e_c}|}{|C_e \cup C_{e_c}|}. \quad \text{(Eq. 3.6)}$$

By evaluating the categorical similarities between the context and each episode, the semantic retrieval algorithm can determine the episode's relevancy at the semantic level. By incorporating with episodic retrieval, ESRA will be able to retrieve both episodic-relevant and semantic-relevant information objects to the user.

3.4 Cognitive Capacities of the Agent Model

ESRA possesses several dimensions of cognitive capacities, which make the model suitable for various real world personalized information retrieval scenarios.

3.4.1 Episodic and Semantic Based Exploration

By introducing episodic memory and semantic memory into the agent model, the user's historical information can be viewed from multiple perspectives. On the one hand, in the episodic memory, the temporal-spatial similarity can be used as clues to find resources with certain attributes, such as their incoming source similarities and time pattern similarities. On the other hand, in semantic memory, the concept network makes it easy to find target concepts or information objects.

With ESRA, it opens the opportunities for the user to explore his memory from multiple views. The memory system provides interfaces to access EM and SM, respectively. The EM interface supports the exploration of episodic resources by time and location. The SM interface supports the exploration of the concept network based on topics.

3.4.2 Reasoning

Resource recommendation is the technique to recommend information objects that are potentially of interest to the user. Given a stream of unseen information objects, the **user profile** in SM can be used to rank the objects based on the similarities between the user profile and each of the candidate information objects.

Resource filtering problems are similar to resource recommendation. Given a stream of incoming resources, the filtering algorithm will try to filter out items that are not of interest to the user. To this extent, the SM can be used to filter information for the user.

The reasoning capacity powered by user profile shows that SM plays an important role in the processing of episodes in EM. More detailed discussions on the agent's reasoning capability is presented in Chapter 5.

3.5 Evaluation Methodology

The agent model described in this chapter is a high-level architecture which might be applicable to various personal information systems. To show its efficacy, it is necessary to build ESRA in specific task environments in which both the environment and the agent's task are explicitly defined. While it is hard to evaluate all

the functionalities of ESRA at the same time, we can demonstrate its primary functional components individually. To sum up, ESRA will be evaluated in terms to its functional interactions and cognitive capacities in the remainder of this dissertation, as briefly depicted as follows:

- Information encoding and storage, especially memory consolidation methods are demonstrated in Chapter 4.
- Context formulation and automatic personal information retrieval from the memory is evaluated in Chapter 5.
- The cognitive capacities of ESRA, such as exploration and reasoning are evaluated in the applications as described in Chapter 6.

3.6 Summary

In this chapter, we described the proposed episodic and semantic memory based remembrance agent model, ESRA, which is based on cognitive memory theories. EM and SM are two parallel and partially overlapping information processing systems [36], each of which has different cognitive structures. The encoding, store and information retrieval mechanisms in EM and SM are different. By implementing a computational model for EM and SM, ESRA is equipped with multi-level retrieving, exploring, and reasoning capacities.

Given the detailed description of ESRA, together with its core elements, actions, and reasoning abilities, it can be observed that two of biggest challenges of the model are consolidation and contextual retrieval. The user's episodic events cannot be converted into user profile without solid consolidating algorithms. On the other hand, without proper context representation and extraction algorithm, the user's context cannot be converted to meaningful queries for the retrieval. In Chapter 4, memory consolidation will be discussed in more details. This is important because it is the function by which the temporal-spatial based information are updated into semantic knowledge. In Chapter 5, an ontology based contextual retrieval algorithm will be developed and evaluated.

Chapter 4

Memory Consolidation Methods

In this chapter, we study consolidation methods which convert user experiences in EM to user knowledge in SM. The purpose of memory consolidation is to build user interests profile, which is similar to the knowledge acquisition process in human memory. Based on cognitive memory theories [4], the repeated associations in episodic memory will be consolidated into semantic memory, which will be reactivated during remembering. We will study the processes by which the agent's abstract knowledge is formulated through inference from the concrete information in individual episodes. Specifically, we take the information objects in EM as input and automatically classify those information objects into predefined categories. The information objects in our research scope are the user's text documents from diverse information sources. The target categories are the concepts in large taxonomies that can represent general human knowledge and be interpreted by both the computer and the human to make inference. As the documents from different information sources are of different characters, we identify the three most representative types of documents and propose three consolidation algorithms to deal with each type of documents. Several experiments are presented to show that the proposed algorithms can significantly improve mapping accuracy when compared with existing approaches.

4.1 Introduction

Consolidating information objects from EM to SM is one of the main objective of ESRA. It services with the purpose of extracting semantic information from individual documents and making links between documents and concepts. Such form of generalization is the foundation with which we humans understand the world. We study consolidation as text classification and entity linking problems. Given a piece of text document, the consolidation algorithms will determine which categories the

document belongs to by analyzing the term features or the entities (on a taxonomy) mentioned in the document.

In this chapter, we study three typical types of user documents in the real world and design specific algorithms for each of them.

- Regular text documents.

  Along with more and more documents collected by the users, comes the need to automatically classify the documents into well structured Web directories. Wikipedia is one of the most popular Web directories which organize concepts hierarchically. While there are already some work done on text classification, the scale of Wikipedia poses great challenges to both classification accuracy and efficiency. As normal text documents are common information objects experienced by the users, we propose a hierarchical classification algorithm based on k-NN to classify documents onto Wikipedia. The detail algorithm design and experiment analysis will be presented in Section 4.2.

- Social tagging in folksonomies.

  Social tagging services such as Delicious[1] enables users to bookmark links for future retrieval, which are primary information sources for the users. The tags given to the links by the user provides extra information about the user's opinion on the corresponding resources, which are not available in normal text documents. In Section 4.3, we study how to derive algorithms to map links onto an existing ontology to enable semantic level understanding of the links.

- Short texts on Microblogging services.

  The rise of Microblogging platforms makes short texts another main information sources for the users. However, understanding short texts is very challenging because the context is rare. To classify the short texts onto its corresponding categories, it is critical to analyze the entities mentioned in the short texts as well as other social features which are not available in previous studies. In Section 4.4, we will study entity linking problem for short texts and propose a efficient mapping algorithm to identify and match entities appeared in short texts.

[1] www.delicious.com

4.2 k-NN based hierarchical text classification

In this section, we describe a large scale hierarchical text classification algorithm based on k nearest neighbor algorithm. Our method integrates N-grams into the computation of k nearest neighbors, and the category hierarchy into the candidate category ranking. Firstly, we enhance the Bag-of-Words model with N-grams to represent documents. Secondly, two k-NN algorithms, using $TF \cdot IDF$-weighted cosine similarity and $BM25$ respectively, are performed to select a subset of candidate categories. Furthermore, several features of the candidate categories are extracted. A log-linear model aggregates these feature values with different weights, which are tuned through cross-validation. The model outputs a ranking score for each candidate category, and finally the top few candidate categories are chosen as the predicted categories. The experiments performed on both tracks show that our method can achieve high accuracy.

Hierarchies are becoming more and more important for the management of web documents. Automated classification of new documents into the categories in a given hierarchy is required to help search, recommend, and filter information for the user. Wikipedia[2] and ODP[3] provide hierarchical categorization information for a wide range of topics. Efficient and accurate classification of new documents into the categories in such web hierarchies is a challenging task to the text mining community.

In this section, we study the problem of large scale hierarchical text classification based on Wikipedia and ODP data. Our approach is a multiple stage classification model which is motivated by the intuition that similar documents often belong in the same categories, and the most likely categories a query document belongs in are usually among the categories of the documents most similar to the query document. Based on this intuition, a k nearest neighbor (k-NN) classification algorithm along with various enhancement is proposed.

First, we use the N-gram extension of the Bag-of-Words (BoW) model to represent a document. Second, each of two k-NN algorithms, using $TF \cdot IDF$-weighted cosine similarity and $BM25$ respectively, is performed to select the k documents most similar to the query document. We collect the categories of each set of k documents and get two sets of categories. These categories are referred to as *candidate categories* of the query document. The two sets of candidate categories are then ranked respectively, and combined afterwards. Since only a small number of categories become candidates, the prediction among them becomes very efficient. In order to find the most possible categories of the query document, we assign a ranking score to each candidate category using a log-linear model of four predefined

[2] `http://www.wikipedia.org`
[3] `http://www.dmoz.org/`

features. The weights of these features are estimated through cross-validation on a training set. Finally, the candidate categories which have the largest ranking scores are chosen as the predicted categories. The competition results show that our approach is one of the best performers. Moreover, our approach is general and can potentially be applied to various hierarchical text categorization tasks beyond these two web hierarchies.

The rest of the section is structured as follows. Section 4.2.1 recaps the basic concepts and notations in our method. Section 4.2.2 presents the k-NN based candidate category retrieval, the features of the candidate categories, and the candidate category ranking model. Section 4.2.3 presents the evaluation results. Section 4.2.4 summarizes the algorithm.

4.2.1 Hierarchies and Classification

In this subsection, we define the terms and notions that will be used in the description of the classification algorithm.

A hierarchy H of categories is a collection of superior categories (*superiors* or *parents*), each of which subsumes a collection of subordinate categories (*subordinates* or *children*). Each subordinate could have its own subordinates, until the most specific categories (*leaf categories*) are reached. Legal categories for the classification task are only those leaf categories in the hierarchy which do not have any subordinates. In the classification task, each leaf category has a collection of documents in the training set, which are compiled by human experts. The structure of the hierarchy is shown in Figure 4.1.

We denote the set of m leaf categories as $C = \{c_1, c_2, ..., c_m\}$. C is the category space for the classification task. The set of n training documents is denoted as $D = \{d_1, d_2, ..., d_n\}$, and the set of u terms (i.e. N-grams with $N \leq 3$) appearing in D as $T = \{t_1, t_2, ..., t_u\}$. In our current approach, only the immediate superiors of the leaf categories are used in the category prediction. We refer to these categories as the parent category space, denoted by $P = \{p_1, p_2, ..., p_s\}$. We denote the set of parent nodes of category c by $Parents(c)$, as one category can be subsumed by more than one parent category. $Children(p)$ denotes the set of children categories of category p. For instance, in Figure 4.1, $Parents(c_1) = \{p_1\}$, and $Children(p_1) = \{c_1, c_2, c_3\}$. The set of documents in categories c is denoted by $D(c)$, the set of categories of document d is denoted by $Cat(d)$, and the set of terms in document d is denoted by $T(d)$. One natural property of the category hierarchy is, if a document d belongs in a category c, it also belongs in each of c's parent categories, i.e., $\forall p \in Parents(c), d \in D(p)$.

The hierarchical classification problem can be stated as follows: given a new query document d, find the leaf categories in C that d belongs in.

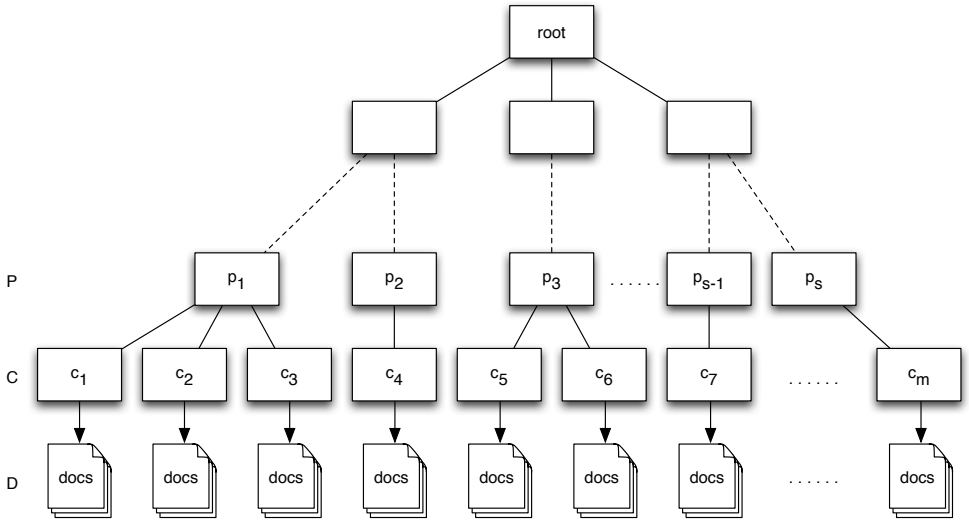


Figure 4.1: Hierarchical Categorization Structure.

4.2.2 A k-NN Based Hierarchical Classification Algorithm

4.2.2.1 Overview

In text classification, k-NN method classifies documents based on the votes of its most similar training documents (k nearest neighbors). We use k-NN to retrieve the most similar k documents for each query document. Rather than ranking the candidate categories by votes (weighted or not) and predict the categories of the document as the most voted categories, we use a log-linear model to rank the categories more reasonably based on four critical features. The weights of these features are tuned through cross-validation. The top ranked categories are chosen as the predicted categories. The overall algorithm is shown in Fig. 4.2.

4.2.2.2 Document Representation

The first step is to prepare the training data and get the tuple set $DS = \{(d_i, \vec{d_i}, c_j)|1 \leq i \leq n, c_j \in Cat(d_i)\}$ in which $\vec{d_i}$ is the term vector representation of document d_i and c_j is one of d_i's category. Note if d_i belongs in multiple categories, i.e. $Cat(d_i) = \{c_{j_1}, c_{j_2}, \cdots\}$, then each category $c_{j_k} \in Cat(d_i)$ corresponds to a tuple $(d_i, \vec{d_i}, c_{j_k})$.

In the BoW representation, in addition to the unigram features, we also extract N-grams (N is up to 3) to expand the feature space.

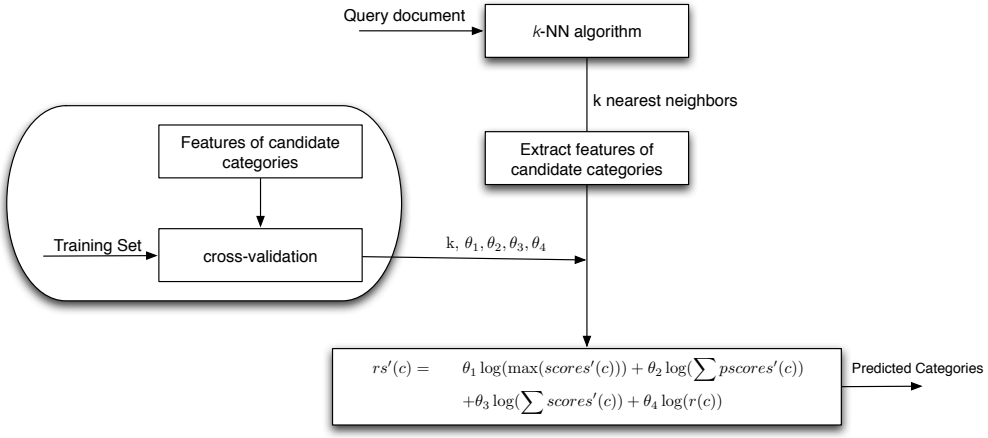


Figure 4.2: Algorithm Workflow

4.2.2.3 Similarity Measures

We introduce both $TF \cdot IDF$-based cosine similarity (referred to as *tfidf*) and $BM25$ as measures to calculate the similarity between a query document and each document in the training dataset.

tfidf We use a variant of $TF \cdot IDF$ model ([65]) to represent the weights of each term in a document, as this variant can improve the accuracy significantly, compared to standard $TF \cdot IDF$ model. The $TF \cdot IDF$ variant is defined as:

$$\omega_{t,d} = log(tf_{t,d} + 1) \cdot idf_t \qquad \text{(Eq. 4.1)}$$

where $tf_{t,d}$ is the frequency count of term t in document d, n is the number of documents in the training corpus, and

$$idf_t = log(\frac{n}{n_t}) \qquad \text{(Eq. 4.2)}$$

is the *inverse document frequency* of term t, in which n_t is the number of documents containing t.

The vector space model for calculating the similarity between the query document d and a training document d_i can be expressed as:

$$cossim(d_i, d) = \frac{d_i \cdot d}{\|d_i\| \cdot \|d\|} = \frac{\sum_{k=1}^{u} \omega_{k,i}\omega_k}{\sqrt{\sum_{k=1}^{u} \omega_{k,i}^2}\sqrt{\sum_{k=1}^{u} \omega_k^2}}. \qquad \text{(Eq. 4.3)}$$

BM25 In BM25 [66] weighting scheme, suppose the query document d contains terms $\vec{t} = \{t_1, ..., t_m\}$. The BM25 score of a training document d_i is:

$$bm25sim(d, d_i) = \sum_{j=1}^{u} idf(t_j) \cdot \frac{f(t_j, d) \cdot (k_1 + 1)}{f(t_j, d) + k_1 \cdot (1 - b + b \cdot \frac{|d|}{avgdl_D})} \cdot \frac{f(t_j, q) \cdot (k_1 + 1)}{f(t_j, q) + k_1 \cdot (1 - b + b \cdot \frac{|q|}{avgdl_Q})} \quad \text{(Eq. 4.4)}$$

where $f(t_j, d)$ and $f(t_j, d_i)$ are t_j's term frequency in document d and d_i, respectively. $|d|$ and $|d_i|$ are the length (count of terms) of document d and d_i, and $avgdl_D$ and $avgdl_Q$ are the average document length in the training data set D and testing data set Q, respectively. k_1 and b are free parameters. In our experiment, we set $k_1 = 1.9$ and $b = 0.9$. $idf(t_j)$ is the inverse document frequency of the term t_j in d, which is computed as:

$$idf(t_j) = log(\frac{n - n_t + 0.5}{n_t + 0.5}) \quad \text{(Eq. 4.5)}$$

where n is the number of document in the training data set and n_t is the number of documents containing the term.

4.2.2.4 k-NN Algorithm

Suppose a query document d in the testing set is given. **Algorithm** 1 depicts our method to calculate its k nearest neighbors in D and the corresponding candidate category tuples. $sim(d_i, d)$ in **Algorithm** 1 could be $tfidf$ or $BM25$.

4.2.2.5 Combination of Two Similarity Measures

Since the k nearest neighbors selected using different similarity measures capture different aspects of the semantics and may complement with each other, we combine the two sets of nearest neighbors, along with their categories, to form the input of our prediction model. The details are trivial and omitted here.

4.2.2.6 Candidate Category Feature Extraction and Ranking

A known drawback of *voting* in k-NN classification method is that documents are not uniformly distributed in all categories, and so the categories with more documents

Require: query document d
for $d_i \in D$ **do**
  $score_i = sim(d_i, d)$
end for
Sort documents in D by $score_i$ in descending order.
Denote the top k documents as KNN(d).
Retrieve tuples from the tuple set DS for each document in KNN(d), get $CD = \{(d_i, \vec{d_i}, c_j) | d_i \in \text{KNN}(d)\}$.
$CS = \phi$.
for $(d_i, \vec{d_i}, c_j) \in CD$ **do**
  $CS = CS \cup (d_i, c_j, score_i)$
end for
return CS

Algorithm 1: Calculating k-Nearest Neighbors and Candidate Category Tuples

tend to come up more often in the k nearest neighbors, leading to a bias favoring frequent categories. Furthermore, for the hierarchical classification, the hierarchical relationships between different categories expose rich information helpful for the categorization of a document, but a simple voting scheme leaves the hierarchical information unutilized. Therefore we use a ranking scheme incorporating both the distributions of the candidate categories and the category hierarchy to improve the classification accuracy.

In this subsection, we will discuss a few features of the candidate categories, and explain how to assign reasonable weights to the features using cross-validation.

Note for a query document d, we will use both $tfidf$ and $BM25$ as similarity measures, get two different sets of candidate category tuples. The following discussion applies to the candidate category tuples under either similarity measure.

Let $CS = \{(d_i, c_j, score_i)\}$ denote the set of tuples representing the k nearest neighbors, their categories and similarity scores with the query document d. Let $CC = \{c_j | (d_i, c_j, score_i) \in CS\}$ denote the set of categories in CS. Based on the analysis on CS, we have found the following quantities are important in our classification task:

(i) For each category c, the set of similarity scores of the documents belonging in c in CS, denoted by $scores(c) = \{score_i | (d_i, c_j = c, score_i) \in CS\}$:

As each category may correspond to multiple documents in the k nearest neighbors (with different similarity scores), it is important to aggregate these scores for better classification. We mainly use two ways to aggregate these

scores: $\max(scores(c))$ and $\sum scores(c)$, which denote the maximum and the sum of the scores in $scores(c)$ respectively.

$|scores(c)|$ denotes the count of scores in $scores(c)$, which is also the number of documents belonging in c;

(ii) The count of training documents in each category c, denoted by $|D(c)|$:

It is observed that the total number of training documents in a category c positively affects the count of documents in c in CS.

(iii) The similarity scores of the nearest neighbors in the parents of a category c, denoted by $pscores(c)$:

In order to utilize the hierarchical structure of the categories, we incorporate into our model the documents belonging the parents of c. The intuition is, if more nearest neighbors of the query document d belong in $Parents(c)$, the chance that d belongs in c is also higher. The similarity scores of those document to d will positively affect category c's ranking. These scores are formally defined as the scores of a subset of CS:

$$pscores(c) = scores(Parents(c)) = \{score_i | (d_i, c_j, score_i) \in CS, c_j \in Parents(c)\}.$$

If a document is in c, it is also in every category in $Parents(c)$. Therefore $scores(c) \subseteq pscores(c)$.

Currently we only use the immediate parents of the leaf categories in the hierarchy, as the higher is the category in the hierarchy level, the less specific is it, and the less informative for the classification.

We derive four features from these quantities as the input of our prediction model:

(i) $\max(scores(c))$, the strongest evidence supporting category c;

(ii) $\sum pscores(c)$, aggregating all the evidence supporting the parents of category c, thus supporting c indirectly;

(iii) $\sum scores(c)$, aggregating all the evidence supporting category c;

(iv) The ratio $r(c) = \frac{|scores(c)|}{|D(c)|}$, intended to reduce the effect of $|D(c)|$ on $|scores(c)|$.

In addition, our model has a few parameters which need to be tuned to get the optimal classification accuracy:

(i) The choice of k, i.e., the size of the nearest neighbors;

(ii) The weights of the four features, denoted by $\theta_1, \theta_2, \theta_3$ and θ_4. These weights scale the contributions of different the features differently, matching with their different correlations with the real categories.

These parameters are tuned using cross-validation on the training set.

We use a simple log-linear model to calculate the scores of the candidate categories. The candidate categories are then ranked by the scores, and the top few categories are chosen as the predicted categories of the query document.

The ranking score of a category c is denoted by $rs(c)$, which is defined as follows:

$$\begin{aligned} rs(c) = \quad & \theta_1 \log(\max(scores(c))) + \theta_2 \log(\sum pscores(c)) \\ & + \theta_3 \log(\sum scores(c)) + \theta_4 \log(r(c)) \end{aligned} \qquad \text{(Eq. 4.6)}$$

4.2.2.7 Parameter Tuning by Cross-validation

To tune the parameters, cross-validation is performed on the medium-size Wikipedia training set. Other tasks use the same set of parameters.

It is observed that the testing data was sampled evenly from each category, we do the same sampling procedure on the training data. We randomly selected one document under each category in the training data, into the validation set. The rest documents form the sub-training set. Eventually, we get a sub-training set of 420,386 documents and a validating set of 36,500 documents.

The documents in the sub-training set along with the corresponding category labels are used to train the model. The validation algorithm then tries different combinations of the parameters on the validating set, and chooses the one that maximizes the prediction accuracy as the optimal parameter values.

Each individual parameters is tried with different values in a pre-specified range. The step size of the values is fixed to 0.1 for all parameters.

The ranges and the optimal parameter values are listed in Table 4.1.

Parameter	Range	Optimal
θ_1	1∼5	3.4
θ_2	0∼1	0.6
θ_3	0∼1	0.8
θ_4	0∼1	0.2

Table 4.1: Parameter tuning range and results

4.2.2.8 Optimized Ranking Score Scheme

We tried the following transformation on the quantities $scores(c)$, and a new function in place of $pscores(c)$ to incorporate the parent-children relationships between categories. The new schemes below are verified to slightly improve the classification performance:

$$\begin{aligned} scores'(c) &= \{\, log(1 + score) \mid score \in socres(c) \,\}, \\ pscores'(c) &= \{\, log(|Children(p)|) \,\big|\, p \in Parents(c) \,\}. \end{aligned}$$

$pscores'(c)$ uses the logarithm of the number of children of each parent of c to represent these parents. Parents with more children will have higher scores, and thus their child categories are more favored in the prediction. But if two categories are both the children of a parent, then the scores of the children of this parent will be the same, and other feature values will determine their precedence.

The new ranking score is as follows:

$$\begin{aligned} rs'(c) = \quad & \theta_1 \log(\max(scores'(c))) + \theta_2 \log(\sum pscores'(c)) \\ & + \theta_3 \log(\sum scores'(c)) + \theta_4 \log(r(c)) \end{aligned} \qquad \text{(Eq. 4.7)}$$

4.2.2.9 Multiple Category Classification

As each document can be assigned to multiple categories in the hierarchy, we select top-M categories as the predicted categories for a query document d. The problem is how to decide M for each document d. Note that M varies across documents.

Let $avgCats$ denote the average number of leaf categories per document within the hierarchy, which is pre-computed from the training set.

For the ranked list of categories $(rs'(c_{i_1}), rs'(c_{i_2}), \cdots, rs'(c_{i_k}), \cdots)$ computed by the candidate category ranking algorithm, we choose all categories whose ranking scores are large enough relative to the largest score $rs'(c_{i_1})$ as d's predicted categories, i.e. $rs'(c_{i_k})/rs'(c_{i_1}) > \alpha$, where $0 < \alpha < 1$ is a constant ratio threshold. In order to tune α, we calculate the predicted average number of categories per document, denoted as $avgPredCats(\alpha)$. By iteratively trying different values of α and calculating the $error = |avgPredCats(\alpha) - avgCats|$, the α value with the minimum $error$ is chosen as the ratio threshold.

4.2.3 Evaluation and Results

4.2.3.1 Evaluation Metrics

The evaluation of the hierarchical text classification algorithm is performed on the datasets provided by two competitions:

- Three categorization datasets provided by the 2nd edition of the Large Scale Hierarchical Text Classification Pascal Challenge (LSHTC2)[4].

  The first dataset, based on Dmoz, contains 27,875 categories, in which most documents belong to only one category and each category has only one parent category. The second and third datasets, both based on Wikipedia, contain 36,504 categories and 325,056 categories, in which a document may belong to multiple categories and each category can have multiple parent categories.

- Two Wikipedia datasets provided by ECML/PKDD 2012 Discovery Challenge[5]. The datasets are multi-class, multi-label and hierarchical. The number of categories range between 13,000 and 325,000 roughly and the number of the documents between 380,000 and 2,400,000.

The metrics used for evaluating the classification algorithms include accuracy, example-based F-measure, label-based macro F-measure, label-based micro F-measure and multi-label graph-induced error [67].

4.2.3.2 Experimental Results

Results for LSHTC2 datasets We compared the performance of the proposed algorithm with the algorithm of a top ranked LSHTC2 participator and the k-NN baseline algorithm. The top ranked method is a neural network based algorithm [68] denoted as *brouardc*. The comparison results in terms of various measures are shown in Table 4.2.

Our algorithm was ranked in the second place in all the three tasks. In contrast, *brouardc* only participated in Task 1 and Task 2 and was ranked in the first and third place, respectively. Compared with the ranks of *brouardc*, our algorithm obtained more consistent performance. On the other hand, our algorithm is more time efficiency which uses much less time for training and testing [68].

Comparing the results for the three tasks, it can be observed that the proposed algorithm can produce promising accuracy. In particular, the accuracy for Dmoz dataset is relatively higher than Wikipedia datasets, which may be attributed to the fact that the classification for Dmoz is a single category classification problem while the classification for Wikipedia datasets are multiple categories classification problems. On the other hand, the accuracy for Wikipedia small dataset is relatively higher than Wikipedia large dataset, which may be attributed to the fact that the more the number of categories and number of documents, the harder it is to

[4] `http://lshtc.iit.demokritos.gr/`

[5] `http://www.ecmlpkdd2012.net/info/discovery-challenge/`

Task	Algorithm	Rank	Acc	EBF	EBP	EBR	LBMaF	LBMiF	MGIE
1	Our algorithm	**2**	**0.3866**	**0.3871**	**0.3882**	**0.3866**	**0.2266**	**0.3867**	**2.8232**
	brouardc	1	0.3885	0.3890	0.3901	0.3885	0.2648	0.3886	2.8294
	k-NN baseline	15	0.1073	0.1075	0.1078	0.1073	0.0375	0.1074	4.2891
2	Our algorithm	**2**	**0.3621**	**0.4217**	**0.4785**	**0.4362**	**0.2133**	**0.3756**	**4.3635**
	brouardc	3	0.3536	0.4189	0.4744	0.4361	0.2415	0.3842	4.0764
	k-NN baseline	7	0.2491	0.3175	0.2829	0.4163	0.1757	0.2978	5.7007
3	Our algorithm	**2**	**0.3367**	**0.4116**	**0.4961**	**0.4157**	**0.1833**	**0.3345**	**4.3920**
	brouardc	-	-	-	-	-	-	-	-
	k-NN baseline	4	0.2724	0.3471	0.3627	0.3869	0.1486	0.3015	4.2883

Acc = Accuracy
EBF = Example Based F1-measure
EBP = Example Based Precision
EBR = Example Based Recall
LBMaF = Label Based Macro F1-measure
LBMiF = Label Based Micro F1-measure
MGIE = Multi-label Graph Induced Error

Table 4.2: Evaluation results for three datasets

distinguish the feature space of each category. Another trend which can be observed from the table is that KNN baseline performs better for Wikipedia datasets than Dmoz dataset, which probably because that the top categories obtained from KNN match the accuracy metric for multiple categories classification better than single category case.

Results for the ECML/PDKK Data Challenge The experimental results for the 2 tasks in Track1 compared to k-NN baseline algorithm and the Multi-stage Rocchio Classification algorithm developed by another top ranked challenge participator [69] in terms of various measures are shown in Table 4.3.

The Multi-stage Rocchio Classification algorithm, denoted as *dhlee*, is a centroids based algorithm which determines the similarity between a test document and a label based on the distance between the test document and the centroid of the label. Our algorithm outperforms *dhlee* in both tasks because we incorporated richer features besides the document similarity.

Comparing the results for the two tasks, it can be observed that the accuracy for Medium Wikipedia data set (task1) is relatively higher than Large Wikipedia data set (task2), which may be attributed to the fact that the more the number of categories and number of documents, the harder it is to distinguish the feature space

Task	1			2		
Algorithm	Our algorithm	dhlee	k-NN baseline	Our algorithm	dhlee	k-NN baseline
Rank	4	6	14	2	3	6
Accuracy	0.412	0.385	0.249	0.346	0.340	0.272
Example Based F1	0.477	0.435	0.318	0.42	0.415	0.347
Example Based Precision	0.518	0.494	0.283	0.607	0.478	0.363
Example Based Recall	0.512	0.426	0.416	0.367	0.406	0.387
Label Based Macro F1	0.245	0.282	0.176	0.17	0.256	0.149
Label Based Macro Precision	0.426	0.476	0.252	0.538	0.409	0.303
Label Based Macro Recall	0.3	0.318	0.235	0.176	0.308	0.177
Label Based Micro F1	0.419	0.421	0.298	0.345	0.345	0.302
Label Based Micro Precision	0.394	0.485	0.251	0.551	0.416	0.326
Label Based Micro Recall	0.447	0.372	0.367	0.251	0.296	0.281
Hierarchical F1	0.677	0.666	0.561	0.546	0.547	0.519
Hierarchical Precision	0.709	0.733	0.512	0.743	0.645	0.542
Hierarchical Recall	0.717	0.656	0.691	0.5	0.546	0.576

Table 4.3: Evaluation results for Track 1

of each category. Another trend which can be observed from the table is that k-NN baseline performs better for Large Wikipedia data sets than Medium Wikipedia data set, which is probably due to the fact that the top categories obtained from k-NN matches the accuracy metric for multiple categories classification better than single category case.

4.2.3.3 The influence of individual features on the accuracy

As described in Section 4.2.2.6, we observed that there are five critical distribution features affecting the classification accuracy. They are (1) the number of nearest neighbors, k; (2) the maximum score of the nearest neighbors in category c_i, $max(scores_{c_i})$; (3) the scaled aggregation of score of the nearest neighbors in category c_i, $sum(scores'_{c_i})$; (4) the scaled aggregation of score of the nearest neighbors in category $Parents(c_i)$, $sum(scores'_{p_j(c_i)})$; and (5) the ratios r obtained by dividing the number of nearest neighbors in c_i by the total number of documents in c_i.

In this subsection, we analyze the influence of each of those features using the validation data to better understand the critical features for hierarchical text classification. To simplify the analysis, we will take the Dmoz dataset as an example for the illustration.

As shown in Figure 4.3, each of the features identified in our approach has

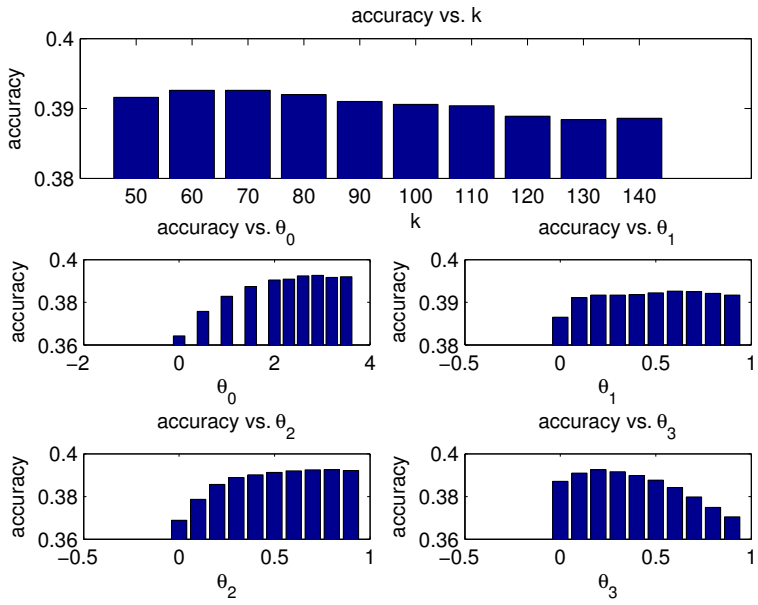


Figure 4.3: The impact of individual features on the accuracy for Dmoz dataset

different level of influence on the accuracy of the algorithm [6]. In particular, the best number of nearest neighbors, k, is 70, for Dmoz dataset. The figure also shows that $max(scores_{c_i})$ significantly affects the algorithm accuracy, as the accuracy changes rapidly as θ_0 grows.

4.2.3.4 Computation Time

The experimental environment for our evaluation is shown in Table 4.4.

OS	Windows Server 2003
Memory	8 GB
CPU	Intel Xeon 2.67 Ghz × 4

Table 4.4: Experimental environment

The computation time for our algorithm is divided into training time, during which the k-NN algorithm runs; validating time, during which the parameter estimation for each feature runs; and testing time, during which the prediction algorithm runs. The summary of the computation time for all of the three datasets are shown in Table 4.5.

[6]When estimating the influence of each features, the optimal parameters are set for the other features

Task	Training	Validating	Testing
DMOZ	19763s	202s	119s
Wikipedia Small	6525s	551s	173s
Wikipedia Large	48411s	-	4536s

Table 4.5: Computation time

4.2.4 Discussion

In this section, we proposed a multiple stages hierarchical text classification method based on k-NN algorithm. Firstly, the k-NN algorithm was performed to select a few most similar documents, whose categories become candidate categories for the classification. Secondly, several important candidate category features were extracted. Finally, the categories prediction algorithm uses a log-linear model to assign scores to the candidate categories, and the top ranked categories are chosen as the predicted categories of the query document. The weights of those features in the log-linear model are estimated through cross-validation.

We found that when calculating the similarity between two document term vectors, a variant of the $TF{\cdot}IDF$ ([65]) can perform better than the standard $TF{\cdot}IDF$ scheme. $BM25$ is combined with $TF \cdot IDF$ for better performance.

We also found that for hierarchical classification problem, the hierarchical relationships expose extra information. Incorporating such information as a candidate category feature can improve the classification performance significantly.

4.3 Collective Ontological Classification for Social Bookmarks

In this section, we introduce a consolidation method which transforms user social bookmarks into semantic user profiles.

With the rise of Web 2.0, users can contribute content to the Web besides consume resources from the Web. Therefore, social tagging systems such as Flickr,[7], Delicious,[8] and Last.fm,[9] are explored to allow users to bookmark resources on the Web (images, documents, music, and so on) using self-defined terms (namely, tags). Previously, researchers acquire user preferences [70] from implicit user behaviors such as clickthrough data [71][72] and eye-tracing [73]. However, the user's social data, like social bookmarking data, provide more practical way to derive user interests

[7]http://www.flickr.com/
[8]http://delicious.com/
[9]http://www.last.fm/

because it is easier to obtain. Tags provide a loosely textual representation of the resource's topics based on the users' perspective, which ignores the type of the corresponding resource. Compared with the tags automatically assigned by computer algorithms and the tags assigned by the authors of the resources, user-generated tags denote the user's own aspect of interests with regards to the resources. A social tagging system provides a platform for users to share their tags with the public. The collectively aggregated resources and their corresponding assigned tags constitute a folksonomy, which explicitly reveals rich information about the users' topics of interests. For instance, Yu et al. [74] analyzed user tagging behaviors in folksonomies and developed algorithms to generate and represent user's multiple interests; and Szomszor et al. [75] collected user interests across multiple social network sites.

Although harvesting of user interest profile from folksonomies and applying it to solve various information overload problems have been studied extensively, e.g., personalized search [76][77][78] and recommendation engines [79], most of them treat tags as individual terms but the semantics of the tags are not well utilized. As the tags are assigned by the users using their own word vocabulary, the aggregated tagging data may contain homonyms, which denote the tags with the same literal but are different concepts, and synonyms, which denote that a set of different tags meaning the same object. Such ambiguities may cause inefficient interpretation about a topic and misconnections among different topics[80]. Another challenge faced with generating user interest profile from folksonomy is the difficulty of modeling user's multiple diverse interests, which is common [74]. [74][81] have studied how to model the diversity of user interests using clustering algorithms. However, the results of their methods are not very promising because of the drawbacks of unsupervised approach. An extensive review on the state-of-the-art research on user interest profile can be found in our previous work [82].

To enhance existing approaches with semantic inference and diverse interests modeling, we propose a new approach for generating and representing user interests, which can provide better representation of user's diverse interests and solve the ambiguities problems in pure folksonomy-based modeling approaches. To implement this, we first map user tags in Delicious onto a Web ontology — DOMZ[10] and then build the user interest profile based on the semantic relationships inherent in the ontology. In this way, the semantics of the tags are normalized and generalized.

The proposed method takes advantage of both user-originality (user-generated tags) in folksonomy and semantics (relationships between concepts) in Web ontology. As the tags are freely chosen, they denote the corresponding user's real topics of interests, otherwise the user may not assign those tags to the resource at all.

[10]http://www.dmoz.org/

However the inter-relationships among the tags are not explicitly defined. On the other hand, as the Web ontology are created and maintained by experts with domain knowledge, the concepts and their relations are explicitly defined. Therefore the ontology is general and make it easy to spread user interests to related concepts in the ontology whenever needed. Moreover, the experimental evaluation of our user interest profiling method indicates that the proposed approach can precisely generate and represent user interest profile, which enables neighboring-concept activation in the Web ontology. We also applied the proposed user interest model to personalizing the search results for individual user on a web search engine. The evaluation results show that our user interest profiling algorithm outperforms the state-of-the-art folksonomy based user interests profiling approaches.

The rest of the section is organized as follows. Section 4.3.1 defines the basic notions and notations in folksonomy. Section 4.3.2 presents the proposed ontological user interest profiling method. (The applying of our model to personalized search with ontological activation algorithm and the evaluation results based on Yahoo Search API will be presented in Section 6.1)

4.3.1 Concepts and Notations

This subsection introduce the notions and notations of folksonomy. A folksonomy consists of a set of resources and a set of user-generated data in social tagging systems, in which users can collect their favorite Web resources and label the resources with freely chosen terms. More specifically, a typical folksonomy contains at least three sets of elements: resources[11], users, and tags. Based on [83], a folksonomy can be formally defined as follows:

Definition 4.17 *A folksonomy, denoted as F, is a tuple ($F = \{U, T, D, A\}$) in which* U *denotes a set of users,* T *denotes a set of tags,* D *denotes a set of Web documents, and $A \in U \times T \times D$ is a set of annotations.*

Within a folksonomy, we further define personomy [74], which denotes the documents, together with the tags assigned to those documents, collected by a particular user. We formally define personomy as follows:

Definition 4.18 *A personomy of user* u*, denoted as P_u, is a restriction of a folksonomy $\boldsymbol{F}$ to* u*: i.e., $P_u = (T_u, D_u, A_u)$ where*

- *T_u is the tag set of user u: $T_u = \{t \mid (t, d) \in A_u\}$,*

[11] Different social tagging systems features resources of different kinds, in this section, we focus on Web documents like those in Delicious

- *D_u is the document set of user u:* $D_u = \{d \mid (t, d) \in A_u\}$, *and*
- *A_u is the set of annotations of user u:* $A_u = \{(t, d) \mid (u, t, d) \in A\}$.

We further introduce two more notations to help further discussions. The first notation is T_d, $T_d = \{t \mid (t, d) \in A\}$, which denotes the collection of tags labeled to d by all users. The second notation is $T_{(u,d)}$, which denotes the group of tags labeled to d by user u, i.e., $T_{(u,d)} = \{t \mid (t, d) \in A_u\}$.

4.3.2 Ontological User Profile Construction in Semantic Memory

Now we present the proposed user interest profiling method. The basic idea is: we first collect all the tags of a single user from the folksonomy and group them by the resources which they are attached to, ignoring the relationships among them; then we link each of the tags to the concepts in the Web ontology to enrich the semantic relationships between the tags. In our approach, we use the Open Directory Project (ODP) taxonomy as the Web ontology. ODP is one of the largest and most comprehensive Web ontology available, which is actively maintained by a global community of domain editors. Moreover, ODP is widely used as the reference ontology by various research projects and real world applications in the area of document classification and user personalization. Because we are processing Web document, ODP is a reasonable choice of ontology for our approach.

The main obstacle faced with the mapping method is that each tag can be mapped to multiple concepts in ODP. For instance, in Figure 4.4, both *python* tag and *java* tag can be mapped to three concepts on ODP, each of which denotes different real-world objects. Therefore, the ontological mapping problem can be formulated as follows: given a collection of tags (grouped by resources) in a user's personomy P_u, how to derive an algorithm to build the mapping between each tag and one single concept in ODP[12]. The problem is hard if each tag have to be mapped individually. Luckily, the co-tagging behaviors, the phenomenon that user normally assign multiple tags to a single document, can serve as contextual clues for the mapping.

$T_{(u,d)} = \{t_1, ...t_N\}$, which has been defined in the previous subsection, denotes a co-tag set. The set of ODP concepts, the mapping targets of the tags, is represented as a concept tree $Tree$. Each concept in $Tree$ is denoted as the path from the root to a leaf in the tree. Suppose that a tag $t_i \in T_{(u,d)}$ can potentially be

[12]Although it is possible for a tag to be mapped to multiple concepts, in our approach, each tag is mapped to a single concept.

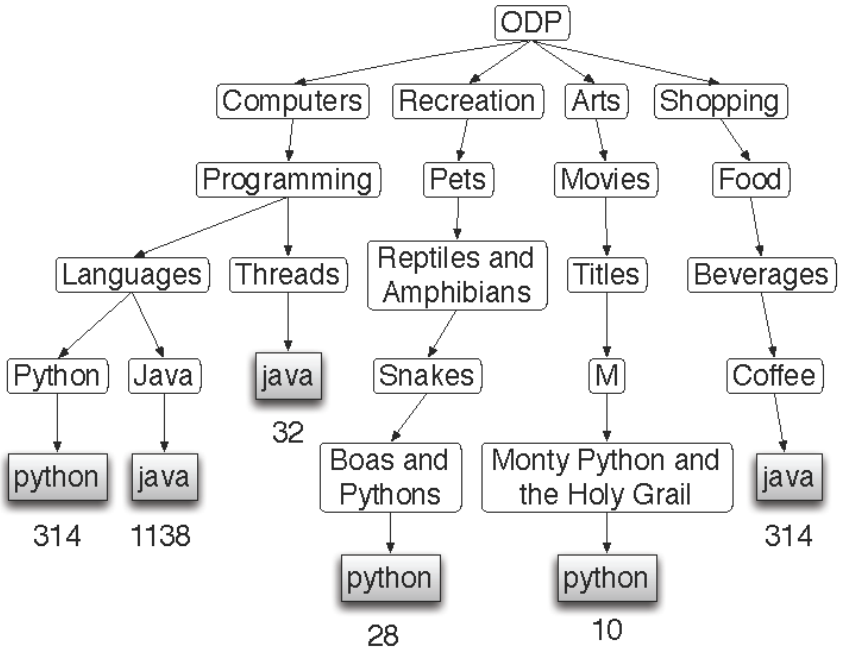


Figure 4.4: The list of mapped concepts for *python* and *java* in ODP categories

mapped to M_i candidate categories: $\{C(t_i, 1), ..., C(t_i, M_i)\}$, each with probabilities $P(C(t_i, j))$. We estimate $P(C(t_i, j))$ with the number of Web documents categorized in the corresponding concept. The best mapping for each tag is determined recursively by breadth-first visiting $Tree$ and estimating the probability at each level. For example, suppose the co-tags set for a single document is $\{python, java\}$, as shown in Figure 4.4, each of the tags can be linked to three candidate concepts. There are totally six leaf concepts in the figure. For *python*, one of the candidate category $C(python, 1)$ is the path *Computers-Programming-Languages-Python-python* with prior probability $P(C(python, 1)) = \frac{314}{314+28+10}$.

At each level l of $Tree$, the probability is estimated by combining the category's prior probability and the global occurrence of the concept in each categories at level l. For example, at the first level in Figure 4.4, the concept *Computers*'s global occurrence is 3, as there are 3 candidate mapped concepts under *Computers*. Thus, $P(python \in Computers, level = 1) = \frac{314}{314+28+10} \times \frac{3}{6} = \frac{157}{352}$. The target category at level l for the tag is the category with the highest probability, while other categories at the same level are ignored. The algorithm runs recursively until a leaf category is reached for each tag. The algorithm is described in Figure 2.

The overview of the proposed user interest profiling algorithm is depicted in Figure 4.5. The mapping result is a tree representation of the user's diverse interests

input : $T_{u,d}$ - the set of tags assigned to document d by user u;
$Tree$ - tree representation of tags in $T_{u,d}$ on ODP
output: $Categorized\_concepts$ — the set of categorized concepts

foreach $t_i \in T_{(u,d)}$ **do**
  foreach $node \in child(root(Tree))$ **do**
    $P(node) = \frac{number\_of\_categories\_bypassed\_at\_the\_node}{total\_number\_of\_categories\_in\_the\_tree}$;
    $P(C(t_i, j)) = \frac{number\_of\_documents\_indexed\_under\_categoryC(t_i,j)}{total\_number\_of\_documents\_for\_the\_concept\_in\_ODP}$;
    $node\_score = P(node) \times P(C(t_i, j))$;
  end
  $node^\star = \arg\max_{node} node\_score$;
  $Tree = Subtree(Tree, node^\star)$;
end

Algorithm 2: Collective Tags-to-Concepts Mapping Algorithm

(as shown in Figure 4.5), in which the tags are mapped to leaf categories. Semantic level inference can be conducted to enrich the user interest profile by activating neighboring concepts.

4.3.3 Discussion

Due to the shortage of training samples, we do not directly evaluate the accuracy of the mapping algorithm. However, as will be described in Section 6.1, we will apply the user interest profile generated from the mapping algorithm to the problem of personalized search. Therefore, the performance of the mapping algorithm is evaluated in the personalized search framework.

4.4 Anchor Text Based Ontological Mapping for Short Documents

Microblogs like Twitter, allow users to follow others based on their interests. The user's behaviors expressed on such platforms offer valuable information about the users' likes and preferences, which provides massive potential for accurate advertising and personalized experience. Mining and representing user interests on such platform are essential elements of successful personalized adaptive systems. However, there is very little research on modeling user interests on microblog platforms.

In this section, we investigated the problem of user interests modeling using microblogging platforms, specifically the Twitter platform. To extract user interests, the entities in the user tweets are first analyzed and linked to concepts on Wikipedia using ranking methods. Then, a *user interests model* is proposed to preserve the semantic relations exposed in these entities. And finally, we apply the user interests

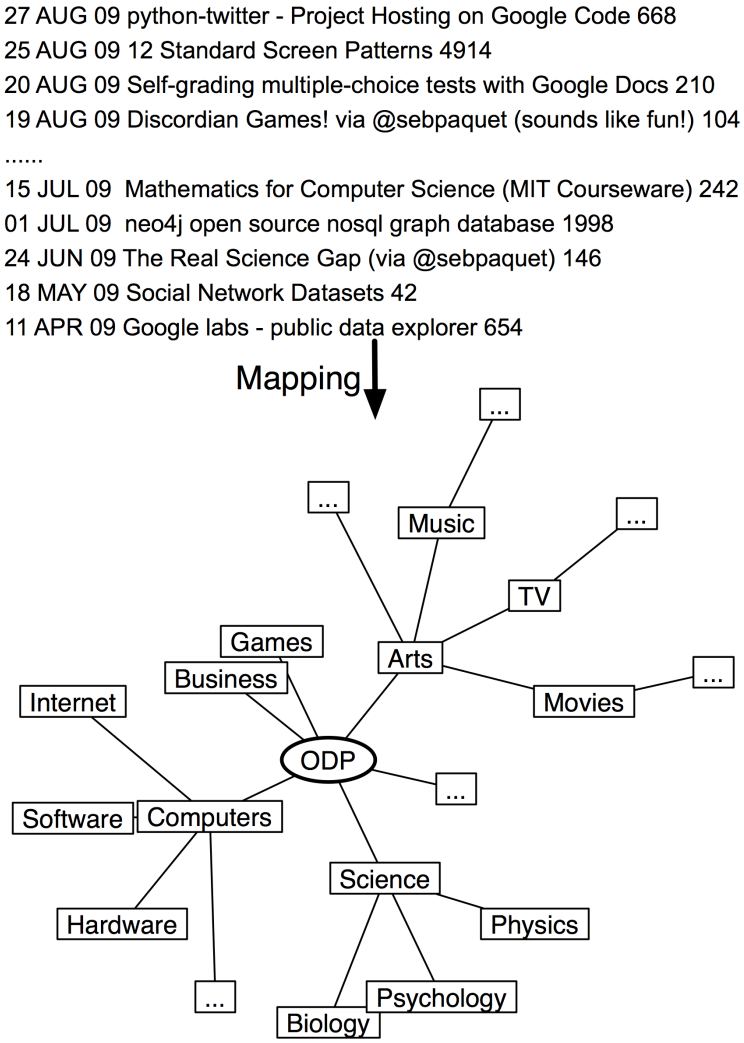


Figure 4.5: User Interest Profiling Overview

model to content recommendation. The experiments showed promising recommendation precision.

Nowadays, the amount of information on the Web is so enormous that it is becoming more and more difficult to find relevant information for individual users. Social media websites such as Facebook and Twitter exposed even more data to the user in a daily basis. While these information are valuable, the user are overwhelmed with social information overload. On the other hand, the user's activities on such platform exposed rich information on their interests and preferences. Such data can be used to build user interests model, which in return, can help solve information overload problem not only with each of individual platform, but across multiple

platforms.

We focus on studying user interests modeling problem on Twitter. More specifically, we try to identify comprehensive user interests from the tweets posted by the user and generate user interests profile which can represent the user's general interests in various entities.

The tweets posted by the user are extremely short (limited to 140 characters) compared with traditional user generated content, which makes it difficult to disambiguate since the context available for processing by general disambiguation algorithms is quite limited. However, as the length is limited, the user tend to explicitly express the entities in their tweets.

There are many potential approaches to extract entities from the user tweets. In our approach, we rely on external ontology for entity extraction and link the entities in the tweets onto their corresponding disambiguated Wikipedia concepts. The motivation of relying on Wikipedia as linking target is based on the following observations. First, concepts on Wikipedia are always corresponding to specific entities in the world knowledge, which means the mapped concepts can represent the user's taste on the specific entities. Second, the synonym and category information of the concepts on Wikipedia can help identify the semantic relations between different concepts, which otherwise are hard to obtain in other approaches.

However, there are several challenges with this approach. First, how to identify potential entities in a specific tweet? As tweets are noisy, the performance of selecting candidate entities will dramatically affect the performance of the mapping algorithm. Second, as each entity might be linked to multiple concepts, how to solve polysemy problem in the context of other observed entities co-occurring in the same tweet? Finally, we would like to show evidence for the hypothesis that the neighboring tweets posted by the same user can be used to disambiguate entities in the current tweet to be examined.

We address these problems by utilizing a knowledge base obtained from Wikipedia and introducing feature ranking methods to select proper concepts for the tweets. Wikipedia provides background information about concepts which we use to disambiguate their appearances in the tweets. Once all the candidate entities in a tweet are disambiguated, we use the concept graph preserved in the co-occurrence relationships to represent the user's interests.

Our overall algorithm runs in two steps. In the first step, the entities in a user's tweets are identified and linked onto Wikipedia concepts. In the second step, the user interests model is constructed based on the linked concepts on Wikipedia. The framework overview is shown in Figure 4.6.

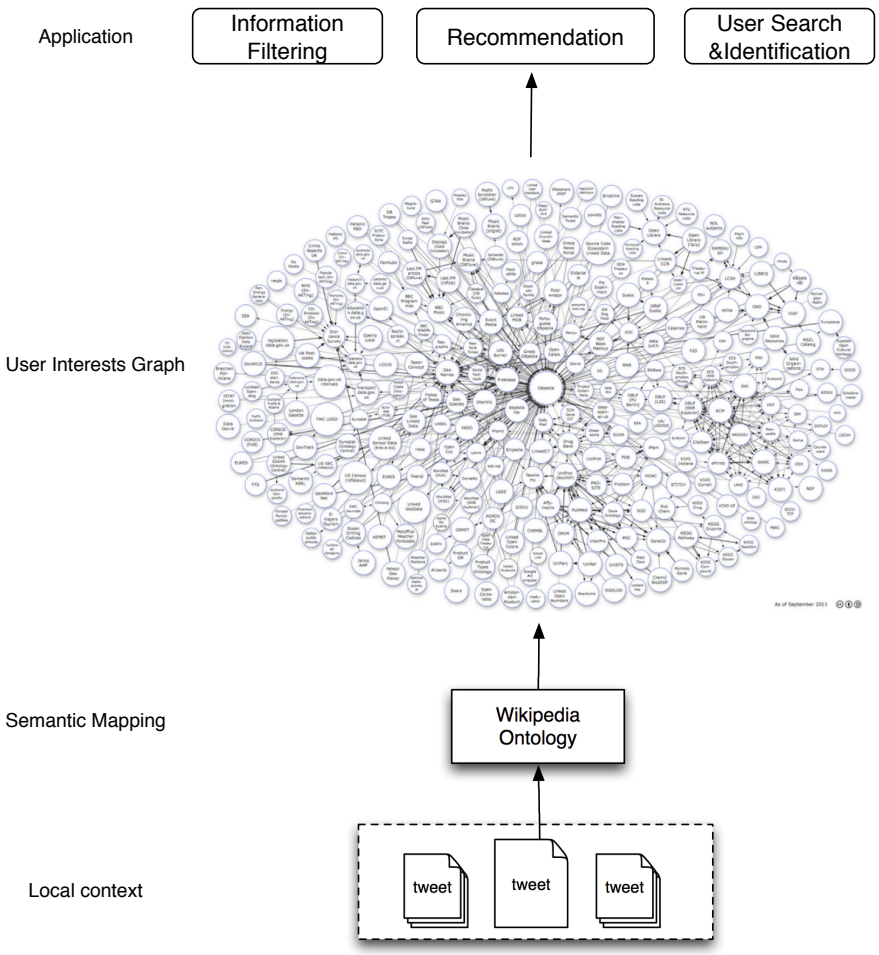


Figure 4.6: Entity linking framework

4.4.1 Linking User Tweets to Wikipedia Concepts

The most commonly used linking approach is lexical matching, which select those concepts whose titles literally matches any n-grams in a tweet [84]. However, this approach can only obtain high accuracy when the source texts are relatively clean. As have been shown in the previous section, user tweets are noisy and this approach cannot achieve high accuracy on user tweets [85]. To improve precision, Meij et al. [85] proposed an approach which involved removing mis-linked target concepts and restricting the number of n-grams utilized to create a linking. Their approach takes 2 steps. First, for each n-gram in a tweet, a ranked list of candidate concepts is generated independently. In the second step, supervised machine learning was introduced to improve precision, in which each candidate concepts is categorized as being relevant or not to the tweet. The problem with this approach is that (1)

```
I enjoyed my meeting yesterday with former President
Bill Clinton as we exchanged views on many topics
including healthcare and education
```

Figure 4.7: An example tweet

mapping commonness was the only feature used for selecting candidate concepts. However, as we will show in the later sections, commonness alone is not good for selecting significant candidate concepts; (2) the machine learning result is suffering from over-fitting problem in practice. The features selected in their approach cannot be directly applied to tweets beyond the small dataset in their experiments as diverse types of tweets exist in the real world; (3) there are some other critical features not considered in their method, but are important for linking. We will introduce these features in the following subsection.

4.4.1.1 Preliminaries

Each Twitter user u has a collection of tweets u_T obtained from user's *user timeline*[13], which is the stream of user status updates.

A collection of tweets T can be represented as a $m \times n$ matrix, in which m represents the number of tweets and n represents the total number of distinct terms appearing in the collection. An n-gram ng is a contiguous sequence of n items from a given tweet. In order to be able to match multiple term titles on Wikipedia, we further extend the matrix to n-gram based representation, in which n is the total number of distinct n-grams appeared in the collection.

Assume each individual Wikipedia article represents a concept c, identified by its URL. Wikipedia can be represented as a collection of concepts C. Wikipedia uses hypertext to enable readers to easily access relevant information on other pages. The page from which the hyperlink is activated is called the anchor; the page the link points to is called the target c_{target}. The terms used to link a page to another page within Wikipedia is called the anchor text q.

For each n-gram ng in a tweet, we generate a list of candidate concepts ng_C. We illustrate the overview linking algorithm with the sample tweet as shown in Figure 4.7. Selected n-grams in the tweet which are lexically used as interlink anchor text on Wikipedia and their corresponding interlink target concepts is listed in Table 4.7. The *score* column are the aggregated ranking score for each candidate

[13] `https://dev.twitter.com/docs/api/1/get/statuses/user_timeline`

concept using our algorithm. The top 10 ranked concepts for the tweet are shown in Table 4.6.

Ranked concepts	Labeled concepts
Bill Clinton	Meeting
Health care	Bill Clinton
President of the United States	Health care
President	Education
Yesterday (song)	
Education	
Clinton County, New York	
President of the Philippines	
Clinton County, Pennsylvania	
Yesterday (TV channel)	

Table 4.6: The comparison of the ranked list of concepts and the manually labeled list of concepts

The question is: given a tweet t, how to decide which n-grams, among all the n-grams in t can be linked to Wikipedia? For each of the candidate n-gram ng and its corresponding concept list ng_C, how to decide which one is the target concept?

4.4.1.2 Selecting candidate concepts

The first step toward linking entities in a tweet to Wikipedia concepts is to identify n-grams which may stand for entities, and for each identified entity, the list of concepts that it might be mapped to.

To identify n-grams in a tweet t, we first extract all possible n-grams from t: $NG = \{ng_i|0 < i < n\}$, where n is the number of all n-grams in t. For each n-gram ng_i, we extract all the Wikipedia concepts with which ng_i is used as an anchor text among all the interlinks on Wikipedia. The corresponding candidate set of concepts on Wikipedia is denoted as $C_i = \{c_{ij}|0 < j < m\}$, where m is the number of concepts for ng_i.

4.4.1.3 Ranking candidate concepts

For each tweet, we select the most significant n-grams, and further select the target concept for each n-gram. Therefore, the objective of the ranking approach is to determine the significance of n-grams both in the tweet corpus space and the Wikipedia concept space. In this section, we will determine the most critical factors that contribute to the ranking of the concepts.

n-gram	Wikipedia concept title	score
exchanged	Prisoner exchange	0.291967
	Exchange (chess)	0.054744
	Exchange	0.009124
president bill clinton	Bill Clinton	22111.425476
	Presidency of Bill Clinton	0.694097
views	View (database)	0.020338
	View (Buddhism)	0.015032
	View model	0.012380
bill clinton	Bill Clinton	22111.425476
topics	Topics (Aristotle)	0.040979
	Topic	0.017562
	Outline of Canada	0.011708
yesterday	Yesterday (song)	18.279802
	Yesterday (TV channel)	2.347865
	Yesterday (2004 film)	1.006228
former president	President of the United States	23.566104
	President of the Philippines	2.413109
	President of Argentina	0.004785
views on	Homosexuality and Roman Catholicism	0.001694
many	Many, Louisiana	0.002994
	Many	0.000783
	Many, Moselle	0.000184
including	Inclusion	0.000056
	Taunton sleeping car fire	0.000028
	Continuous spectrum	0.000028
healthcare	Health care	29.097476
president	President of the United States	23.566104
	President	19.830676
	President of the Philippines	2.413109
education	Education	7.042623
	Local education authority	0.427439
	Education in Ethiopia	0.134308
meeting	Meeting	0.068024
	Commonwealth Heads of Government Meeting	0.013127
	List of United States Congresses	0.002983
former	Former	0.004306
	Retirement	0.001862
	List of Prime Ministers of Pakistan	0.000698

Table 4.7: Selected n-grams and their corresponding candidate concept set

N-gram weight in the corpus We measure the weight of each n-gram ng in a tweet t using the standard vector space model, in which $TF \times iDF$ is used as weighting scheme as shown in equation Eq. 4.8.

$$weight_{(ng,t)} = tf_{(ng,t)} \times log \frac{|T|}{\{t \in T : ng \in t\}} \quad \text{(Eq. 4.8)}$$

$weight_{(ng,t)}$ is a numerical statistic on the importance of a n-gram ng to a tweet t in a collection of tweets T.

N-gram Wikipedia linking probability $linkprob_{ng_i}$ measures the probability that n-gram ng_i being used as anchor text in Wikipedia [86]. It is defined as the ratio between set of Wikipedia articles contains ng_i as anchor text and the total number of Wikipedia articles in which ng_i appeared in.

$$linkprob_{ng_i} = \frac{\sum_{c \in C_{a(ng_i)}} n(ng_i, c)}{\sum_{c' \in C_{(ng_i)}} n(ng_i, c')} \quad \text{(Eq. 4.9)}$$

$linkprob_{ng_i}$ indicates the "conceptability" of ng_i in the Wikipedia articles corpus.

N-gram concept frequency $commonness(c, q)$ is the prior probability that anchor text q is being linked to target concept c in Wikipedia. If we consider the interlink in Wikipedia as a matrix, in which each row represents an anchor text q, and each column represents a concept c, then each element in the matrix represents the frequency with which q appeared as anchor text for a concept c, denoted as $freq_{(q_i,c_j)}$. We further normalize the rows of the matrix to a length of 1 and denote the elements as $commonness(c, q)$.

$commonness(c, q)$ indicates the normalized ngram-concept linking frequency for each pair of ngram-concept mapping instance.

Wikipedia concept disambiguation diversity We define the mapping diversity $diversity_{ng_i}$ for n-gram ng_i as the ratio of the number of interlink instances in which ng_i was used as anchor text to the number of distinct target concepts among all the linking instances.

The higher $diversity_{ng_i}$ for ng_i the more difficult it is to disambiguate the n-gram, and less likely it is for the n-gram to be able to represent the meaning of its parent tweet.

Ranking scheme Based on the criterion defined in the previous sections, we derived a concept ranking scheme which incorporate the most significant factors in both the tweet corpus space and Wikipedia corpus space. The scheme is show in equation Eq. 4.10.

$$rank_{c,tweet} = weight_{(ng,t)} \times linkprob_{ng} \times commonness(c, ng) \times diversity_{ng_i} \quad \text{(Eq. 4.10)}$$

input : X — n-grams in a tweet;
C_i — candidate concepts for a selected n-gram
output: $rank_{c,tweet}$ — the ranks for each candidate concepts for a tweet

```
foreach x_i ∈ X do
    foreach c_ij ∈ C_i do
        Calculate rank_{c_ij,tweet}
    end
    Aggregate rank'_{c_ij,tweet}
end
```

Algorithm 3: Selecting and ranking candidate concepts

The selecting and ranking algorithm is shown in Algorithm 3. Top 50 ranked concepts in $rank'_{c_{ij}}$ are selected as candidate concepts.

4.4.1.4 Algorithm Performance

In this section, we describe the experiment setup and the performance of the proposed concept linking algorithm.

We use a copy of Wikipedia that is extracted on 11 Nov. 2011 and perform some pre-processing [87][88] to filter articles with one or more of the following characters: (1) articles that have fewer than 100 non stop words or fewer than 5 incoming and outgoing links; (2) articles that describes specific dates and events; (3) Wikipedia disambiguation pages; (4) articles belongs to categories related to chronology, i.e., *Year*, *Decades*, and *Centuries*; (5) the first letter is not capital letter; (6) the title is a single stopword; (7) for a multiword title, not all words other than prepositions, determiners, conjunctions, or negations are capitalized; (8) title occurs less than three times in its article.

Two datasets were introduced in our experiment for the evaluation. The first dataset is the Edgar dataset provided by [85] and the second dataset is the Tweetstream dataset we harvested from Twitter stream API. The evaluation measures include precision at rank 1 (P1), r-precision (R-prec), recall, mean reciprocal rank (MRR), and mean average precision (MAP) [89]. The top-50 candidate concepts were used for evaluation.

Evaluation on Edgar Dataset This dataset consists of the last 20 tweets from the randomly sampled users from the "verified accounts" Twitter list. The same preprocessing is performed to remove URLs, eliminate the leading "@" and "#" in mentions and hashtags respectively. The twitter dataset, which consists of 562 tweets,

	P1	Recall	MRR	MAP	R-prec
CMNS	0.6021	0.7775	0.7080	0.5853	0.5271
CMNS-RF	0.6780	0.8417	0.7676	**0.6561**	**0.5739**
TW-Ranking	**0.7122**	**0.8763**	**0.7853**	0.6264	0.5464

Table 4.8: Edgar dataset mapping performance comparison

each containing 36.5 terms on average, was manually annotated with Wikipedia concepts.

Table 4.8 shows the comparison between the different approaches — *CMNS, CMNS-RF*, and the algorithms developed in this article *TW-Ranking*. *CMNS* is a ranking algorithm only using $commonness(c, ng)$ as weighting and *CMNS-RF* is a machine learning algorithm incorporating nearly 20 features with 5-fold cross-validation. *TW-Ranking*, short for *tweet-wikipedia ranking*, is the algorithm developed in this article, which incorporate the 4 identified features in both the tweet corpus and Wikipedia corpus. It can be observed that *TW-Ranking* outperforms both *CMNS and CMNS-RF* in terms of both P1 and Recall, which are the two most important metrics. Moreover, our approach is more generalizable because much less features are used in the model.

Evaluation on Tweetstream Dataset The Tweetstream dataset is obtained by filtering out tweets that include links to Wikipedia using Twitter Streaming API[14], which consists of 27182 tweets. We then limit the tweets only to those linked to English Wikipedia concepts and the authors of which are English users. A sample tweet message in this dataset is shown as follows:

> "@DavidFCox I redirect you to the power set: http://en.wikipedia.org/wiki/Power_set"

In the sample, *@DavidFCox I redirect you to the power set:* was used as the tweet text from which to select entities and to link them to Wikipedia concepts. ***http://en.wikipedia.org/wiki/Power_set*** was used as the target linked concept for us to verify our algorithm performance. As it is impossible to link the tweet to Wikipedia if the linking URL is the only content of the tweet, we eliminated tweets with this pattern.

This dataset is much harder to work with mainly because usually the text in the tweet text may not show any clue with regards to the Wikipedia article linked in the tweet. Table 4.9 shows the performance comparison on the tweetstream dataset,

[14] `https://dev.twitter.com/docs/streaming-api/methods`

as compared with the $CMNS$ algorithm developed by Meij et al. [85]. We only implemented $CMNS$ for comparison and it can be observed that our algorithm, *TW-Ranking* significantly outperforms *CMNS* in terms of all the measures being evaluated, which means our approach is more scalable.

	P1	Recall	MRR	MAP	R-prec
CMNS	0.1761	0.4247	0.2494	0.2494	0.1761
TW-Ranking	**0.3288**	**0.5871**	**0.4130**	**0.4130**	**0.3288**

Table 4.9: TweetStream dataset mapping performance comparison

4.4.2 Graph Based Interests Representation

The concept linking algorithm developed in the previous section, *TW-Ranking*, provided the primitive mechanism to infer user interests at the individual tweet level. To analyze individual Twitter user's interests, we need to extract the snapshot of selected user's tweets. In this section, we will introduce how to aggregate the concepts into a rich user interests model.

4.4.2.1 Semantic User Interests Model

Before investigating the user interests modeling algorithm, let's define what a user interests model is. It is assumed that if two concepts appeared in the same tweet of a specific user, the user is more interested in co-occurrence of these two concepts than each individual concept.

Definition 1 *The user interests model of a user $u \in U$ ia a graph $G = (V, E)$ comprising a set V of concepts connected with a set E of edges. v_i represents the weight of the concepts for u, and e_j represents the weight of the activation strength between two concepts for u.*

A sample user interests model is shown in Figure 4.8. By modeling user interests with a graph, the relations between different concepts from the user's perspective is preserved.

4.4.2.2 Collecting interests from user tweets

Based on the user interests model defined above, we develop a user interests collecting algorithm. Various interests representation has been developed [90][91][92], of which the most widely used model is vector representation. We use vector representation as baseline and develop two advanced models.

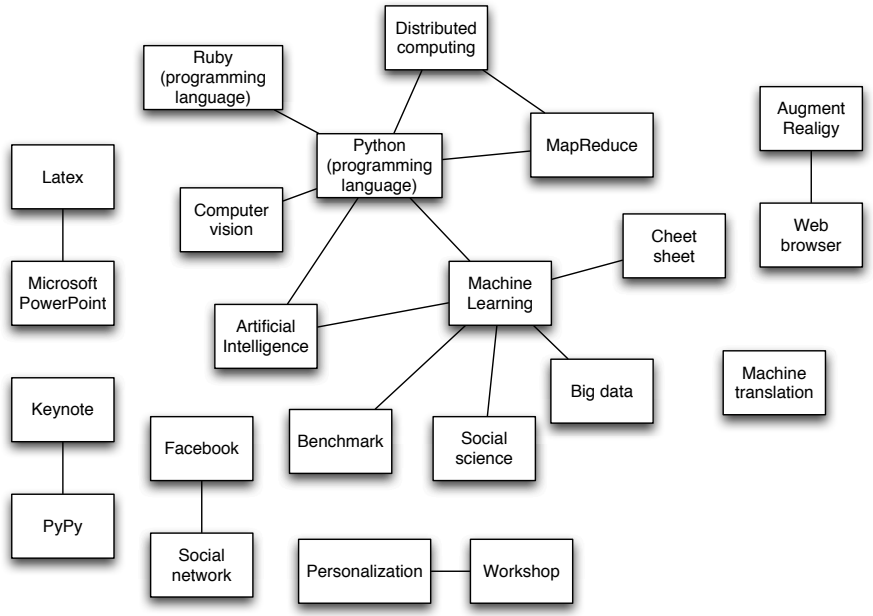


Figure 4.8: Graph representation of user interests based on concept co-occurrences extracted from user tweets

Vector representation The baseline representation considers the user interests as a term vector, in which the terms are collected from the text in user tweets. $In_{(u,vector)} = \{ngram_0, ngram_1, ..., ngram_i, ...ngram_n\}$, where $ngram_i$ corresponds to the frequency of each n-gram in the user's tweets. Similarly, the n-gram representation of a tweet being evaluated is $t_{vector} = \{ngram_0, ngram_1, ..., ngram_j, ...ngram_n\}$, where $ngram_j$ corresponds to the frequency of each n-gram in t.

The similarity between t_{vector} and a specific user's interests model $In_{(u,vector)}$ is calculated based on Euclidean dot product, as shown in Equation Eq. 4.11.

$$similarity(In_{(u,vector)}, t_{vector}) = \frac{In_{(u,vector)} \cdot t_{vector}}{\left\|In_{(u,vector)}\right\| \left\|t_{vector}\right\|} \qquad \text{(Eq. 4.11)}$$

The problem with vector model is that the term frequency does not contain semantic information and the resulting terms in the vector might contain meaningless entities. For instance, a possible vector model extracted from the tweet shown in Figure 4.7, which failed to capture lots of semantics in the original text, can be represented as follows:

v=(topics, meeting, including, healthcare, bill, education, yesterday, views, many, clinton, president, exchanged, former)

Concept representation The concept representation using the concepts linked to Wikipedia from user tweets to represent user interests. $In_{(u,c)} = \{c_0, c_1, ..., c_i, ...c_n\}$, where c_i corresponds to aggregated score of the ith concept extracted from the user tweets. Similarly, the concept representation of a tweet being evaluated is $t_c = \{c_0, c_1, ..., c_j, ...c_n\}$, where c_j corresponds to the score of each concept extracted from t.

The similarity between t_c and a specific user's interests model $In_{(u,c)}$ is calculated based on Euclidean dot product in the concept space, as shown in Equation Eq. 4.12.

$$similarity(In_{(u,c)}, t_c) = \frac{In_{(u,c)} \cdot t_c}{\left\|In_{(u,c)}\right\| \|t_c\|} \quad \text{(Eq. 4.12)}$$

With the proposed concept mapping algorithm and some post clean up, the user interests extracted from example tweet in Figure 4.7 can be represented as:

```
v = (http://en.wikipedia.org/wiki/Bill_Clinton,

http://en.wikipedia.org/wiki/Health_care,

http://en.wikipedia.org/wiki/Education )
```

Graph representation The graph representation, as defined in Definition 4.4.2.1 preserves the co-occurrence information between the concepts extracted from each tweet. $In_{(u,graph)} = G(V, E)$, in which V is the set of concepts extracted from user tweets, each with aggregated score v_i, and E is the set of edges representing the co-occurrence relations between the concepts, each with weight e_j, obtained by counting the total number of co-occurrence between the two connected nodes in all user tweets.

We represent the tweet t being evaluated as $t_{graph} = G(V', E')$, in which V' is the set of concepts in t and E' is the set of edges connecting each pair of concepts in V', as V' appeared in the same tweet t.

The similarity between t_{graph} and a specific user's interests model $In_{(u,graph)}$ is calculated based on both node similarity and connectivity similarity. The node similarity is calculated in the same way as in Equation Eq. 4.12. The connectivity similarity $connectivity(u, t_{graph})$ is calculated by aggregating the connections of V' in G. The final graph based similarity measure is shown in Equation Eq. 4.13.

$$similarity(In_{(u,graph)}, t_{graph}) = connectivity(u, t_{graph}) * \frac{In_{(u,graph)} \cdot t_{graph}}{\left\|In_{(u,graph)}\right\| \|t_{graph}\|} \quad \text{(Eq. 4.13)}$$

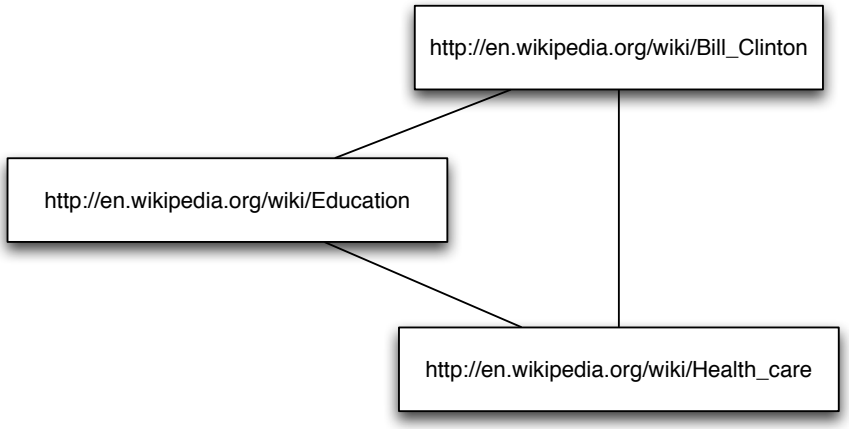


Figure 4.9: Graph based user interests representation

The graph representation of user interests for the above example is shown in Figure 4.9.

The advantage of graph representation is that the weights between concepts provides extra semantic information. For example, the user may only interested in *books about machine learning*, rather than the topic of *book* or *machine learning* individually. By preserving such information, the representation can be more precise. We will show the performance analysis in the application section.

4.4.3 Relations with Other Work

This research work is a mix of user interests modeling and entities extraction. In this section, we will review the related work in these two domains and illustrate what they have done and what are the shortcomings.

Java et al. [93] studied user intentions on Twitter and found that there are mainly four type of intentions: daily chatter, conversations, sharing information/URLs, and reporting news. Jansen et al. [94] demonstrated that out of the Twitter users analyzed, nineteen percentage of them mentioned the fifty selected brands, which motivates them to use named entities, rather than terms, to understand tweets. Kwak et al. [95] found that 85% of the trending topics are relevant to real-time breaking news, and 20% of the investigated Twitter users talked about those trending topics in their tweets. All these evidences show that the entities in tweets are useful source of information and it is reasonable to mining user interests from the tweets posted by the user.

4.4.3.1 User interests modeling on social media

In the search era, lots of studies have been carried out to develop various implicit or explicit user interest models from user browsing history, and then utilizes the information to re-rank Web search results [96] [72]. Further, utilizing domain ontology to build user interest model have been developed. In these approaches, a user interest profile is represented in terms of concepts in the ontology. Middleton et al. [97] developed two algorithms which use research paper topic ontology to construct user profiles. The entities are extracted from significant terms in the texts. Ma et al. [98] employed four approaches to map a user's interests onto appropriate ODP[15] categories using natural language processing techniques.

More recently, since social networks are becoming more popular, users engaged in the network are suffering from social information overload problem and the need to filter noises and recommend information based on the user's taste becomes very critical. Bernstein et al. [99] designed a system for organizing and displaying tweets by topic. Some researchers applied topic models microblogging platforms and social networks to model the topics hidden in tweets [100][101]. The problem with LDA on twitter is that because the data is sparse, the resulting topics are based on the mentioned terms rather than hidden concepts [91].

The length limit of social updates such as tweets on microblogging systems makes it difficult to model their topics. Bernstein et al. [99] relieved such limitation by expanding the tweet content with relevant webpages obtained from web search engines.

There are some attempts at using Twitter as platform to conduct user interests modeling research. Phelan et al. [90] utilized TFIDF to build user preference vectors. Canini et al. [91] prepared two word cloud representation for each account — TFIDF and LDA. Chen et al. [92] developed recommendation algorithms to recommend tweets to users based on his/her own tweets and information from social networks. One of their recommendation approaches is based on user topic profile, which is represented as a bag-of-words model based on the tweets of the user. Their work didn't provide qualitative analysis on their algorithm performance. But one drawback with this approach is that bag-of-words model does not work well on short documents and the generated terms are too specific to represent the topic of interests of the users.

Michelson et al. [84] developed an approach to discover Twitter user's topic of interests by leveraging Wikipedia to map the *entities* appeared in the user's tweets, and followed by analyzing category hierarchies for all of the extracted entities in the user's tweets collection. Selected categories are used to define the user's topic profile. Abel et al. [102] aimed at building interests profile for Twitter users. Their

[15]http://www.dmoz.org/

approach uses OpenCalais[16] for semantic enrichment, which allows for the detection and identification of 39 different types of entities such as persons, events, products or music groups and moreover provides unique URIs for identified entities as well as for the topics so that the meaning of such concepts is well defined. However, the performance is not provided.

Linking Text Correctly identifying entities in user tweets is critical for our user interests model. Semantical enrichment of digital assets has been studied very extensively both for multimedia [103] and text [86][104].

Gabrilovich et al. [87] enriched text by identifying the 10 most relevant encyclopedia articles for each document, without the need for deep language understanding or pre-cataloged common-sense knowledge. Algorithm:

- Centroid classifier, which represent each concept with an TFIDF vector of the article text.
- Document features at words, sentences, paragraphs, and entire document levels individual.
- Using link structure (1) anchor text as alternative names, variant spellings and related phrases for the target concept; (2)number of incoming links as significance of an article.

The problem with their approach is that for the mapping algorithm, Wikipedia category hierarchy is not considered, which was in fact rich source of information for performance boosting [105]. Hu et al. [88] developed a method to enhance content similarity measure for text clustering by leveraging Wikipedia semantics (synonym — redirect page, polysemy — disambiguation page, hypernymy — categorization, and associative relations — outlinked categories). Hu et al. [106] developed text clustering method by exploiting internal (NLP techniques to extract original features and seed phrases) and external semantics (external feature space (TFIDF) — title, anchor text, key phrases in article content), and query Wikipedia with seed phrase to retrieve the top k articles from Wikipedia. Mihalcea et al. [86] developed an approach to enrich a text with links to encyclopedic knowledge by (1) automatic keyword extraction — extracting all possible n-grams in the text that are also present in the controlled vocabulary, and ranking (TFIDF, χ^2 independent text, and Keyphraseness) the likelihood that each n-grams is a valuable keyphrase; and (2) word sense disambiguation (algorithm 1: knowledge-based approach, see our previous work [82]; algorithm 2: data-driven method, integrates both local and topical

[16] http://www.opencalais.com/

features into a machine learning classifier). The linking algorithm in [107] and [108] operated at the lexical level and matched the concept titles to terms in the text. The problem with this approach is that synonymy and homonymy are not resolved. Therefore, a lot of noises are introduced which makes it very hard to disambiguate individual entity and guarantee global precision at the tweet level. The method proposed by Milne and Witten [104] introduces a machine learning method which includes *commonness* and *relatedness* as features. However, as shown in [85], this approach does not work well on tweets because tweets are so short and noisy that the *relatedness* features cannot always be correctly obtained. Tagme [109] used Wikipedia anchor texts as spots and the pages linked to them in Wikipedia as their possible word senses and developed a voting scheme to weight the terms. Edgar et al. [85] identified a series of Wikipedia specific and twitter specific features and utilized Random Forest machine learning method to improve precision.

4.4.4 Discussion

In this section, we studied user interests modeling problem on social media, specifically Twitter. We developed a user interests model based on external ontology, Wikipedia, and showed that the proposed concept mapping algorithm outperformed previous approaches. Based on the concept mapping algorithm, we developed a graph based user interests model and applied the model to recommend tweets to users. The recommendation results were promising.

4.5 Summary

In this chapter, we studied the problem of consolidating information from episodic memory into semantic memory in order to build an ontological user interests model. This is one of the most important processes of the proposed ESRA agent model. We described three different consolidation methods (each targeting a different type of content) and evaluated their performance on different datasets. After consolidation, the agent's memory stores both the user's experience and knowledge, which then allows us to explore how to retrieve such information in the user's current context.

Chapter 5

Contextual Retrieval from the User's Multiple Information Sources

Based on the proposed remembrance agent model described in Chapter 3, the agent lives in the user's task environment and is always ready to retrieve potentially helpful resources to enhance the user's cognition for problem solving. In this chapter, we study the contextual personal information retrieval problem in the user's task environment. Because the information we are retrieving are collected from the user's own information sources, the retrieval problem in our scope is a *re-finding* problem. We will describe how re-finding is different from normal information retrieval problems. On the other hand, we treat context as a time period during which the user is trying to solve information related problems. The rich features in the task environments makes it different from session based context, which is widely studied in web search. Based on these two distinct features, we proposed a contextual information retrieval framework to retrieve the user's most relevant personal Web information from the agent's episodic and semantic memories. Without our framework, the user may not recall such information. Based on cognitive memory theories [4], two types of retrieval signal will trigger the retrieval process. The modeling of EM and SM enables us to have both types of trace available for context matching. The proposed retrieval model will consider both the episodic and semantic traces in the current context to accurately retrieve information from the memory. Based on the retrieval model, we study the information re-finding problem on the social web and developed a contextual information retrieval system — *WebESRA*.

5.1 Overview

5.1.1 Introduction

Nowadays, more and more users rely on multiple social network services to share and bookmark resources for future retrial. With the rapid growth of these resources, it is becoming more difficult for the user to actively retrieve such resources when they need them in their task environment. Task environment based re-finding is one of the main objectives for ESRA. ESRA acts as an intelligent assistant to make connections between what the user is doing now to what the user has done in the past. To make the problem more concrete, we study the problem on the social web. We regard the user's information behaviors on the social networks, like Facebook and Twitter, as information sources and integrate user's information from multiple social networks into a uniform representation. Those information are the collection of resources that will be retrieved based on the user's information need.

As the users are not as good at remembering things as computers do, we are always faced with the re-finding problems, like, "I have seen this resource before, but I cannot remember its name or description", "I have seen some resources similar this one, but I can't remember where I saw them", or, the user might bookmarked a resource which might be useful for future contextual reference, but sometime later when the user is actually in the context, he may has forgotten the existence of the resource at all. To those extent, ESRA can act as the external digital memory which extends the user's physical memory to help the user retrieve those resources in context.

Researchers have explored various sources of context to model the user's information need in information retrieval. The information retrieval process is characterized by a collection of documents and a user query. Various document-query matching algorithms [110][111] have been developed to rank the documents based on their textual similarities with the query. The problem with those approaches is that the user's context, which may contain extra information about the user's search intent, was not considered in the retrieval model and the documents extracted from those models are uniform for all users [112]. L. Tamine-Lechani [113] reviewed and summarized the five context dimensions for information retrieval utilized in various contextual IR literature [114] [115]. Those efforts bring the study of the characteristics of individual user and the task environment to assisting diverse user information need [116][117].

Retrieving information from the user's past data is known as *re-finding* problem. It is widely studied in the personal information management [33][118][119] and web search domain [120][121][122]. Deng et al. [123] observed that the limitation of human memory to remember the details of resources seen long time ago is the main

problem with information re-finding, which implies that other contextual clues are required beyond content clues for efficient retrieval. Deng [121] used time, place, and concurrent activity as clues to retrieve viewed web pages. Sawyer, B. [124] demonstrated how to re-find personal resources based on the physically interacted people and groups, which emphasizes the sisal aspects of factors for information re-finding. Yu. X. [125] studied email re-finding problem by visualizing the social network structure in the user's email archives and concluded that the graphical visualization can help users find their old emails efficiently. The problem with browsing history based re-finding is that web pages in a user's browsing history may not be a good source of information to extend user memory because viewing is not a good indicator of user interests.

Nowadays, more and more users are engaged in social networks to find, share, and publish information. For example, Twitter is regarded as an interests network on which users share the activities they are doing [93][95]; Facebook is treated as a friend network on which users interact with their friends via various sharing activities [126]; Delicious is mainly used as a social bookmarking service on which users collect web resources with the intent to reuse the resources in the future [77]. The user's behaviors on those social networks serve as practical candidate of information sources about the user's past. To get a relatively complete picture about the user's information consumption behaviors on the social web, researchers have developed algorithms to aggregate the user's data on multiple social networks to build user interests model [127][128][129] and identify unique users [130]. Wang, Q. [131] derived a user interests model from the user's social activities, specifically, the user's blogging and social bookmarking and then used the model to personalize search results. It argued that the data collected from the user's social networks is more practical than those obtained from search sessions to derive user interests model. It also showed that the user interests model obtained from multiple social networks is better than the one obtained from single social network via evaluation on personalized search.

However, there is little work done on cross social network data aggregation for context based autonomous personal information retrieval, which is the main theme of this chapter. We collect user's personal data from multiple social networks and encode the data into ESRA's episodic memory and semantic memory. On the other hand, the agent monitors the user's browser based task environment, which non-intrusively point out the connections between the contextual clues extracted from the task environment and the previously collected documents in the user's multiple social networks. The remembrance agent described in the social web scenario is named *WebESRA*, which is an implementation of what Vannevar Bush envisaged in 1945[1] and extend ESRA to the Web environment with rich and complex information.

[1] `http://www.theatlantic.com/magazine/archive/1945/07/as-we-may-think/303881/`

5.1.2 Collecting User Data from Multiple Information Sources

SNS	Data Type	Data Description
Facebook	profile	The user's profile
	friends	The user's friend list
	user feed	The list of user's own updates
	home feed	The list of friends' updates
	likes	The *Facebook pages*, stream items(updates, links, etc), and urls liked by the user
Twitter	profile	The user's profile
	friends	The users followed by the user
	tweets	The tweets posted by the user
	timeline	The tweets posted by the user's friends
	mentions	The tweets posted by other users which mentioned the user
FourSquare	profile	The user's profile
	friends	The user's friend list
	checkins	The check history of the user
	badges	The badges of the user
LinkedIn	profile	The user's profile
	friends	The list of connected users of the user
	user feed	The updates by or about the user
	home feed	The list of updates by the user's friends

Table 5.1: Information sources and the data collections

5.1.2.1 Information Sources

As the number of active social network is quite large, different users have their own choices of engagement. In our study, without loss of generality, we limit the data sources within four most popular social networks — Facebook[2], Twitter[3], FourSquare[4], LinkedIn[5]. The data we obtained from each of the social network are listed in Table 5.1.

Before storing the user data into the agent's memory, the data from different social networks are combined based on their types. E.g., friends from different social networks are merged into a single list of friends; the updates posted by the user on

[2] `https://www.facebook.com/`
[3] `https://twitter.com/`
[4] `https://foursquare.com/`
[5] `http://www.linkedin.com/`

different social networks are merged into a single list of user feed; the updates posted by the friends of the user are merged into a single list of home feed. By performing such preprocessing, we obtained a uniform representation of the user data, as shown in Table 5.2. A sample item in the user feed is shown in Figure 5.1. The item contains the information object, denoted as *body*, which is actually the content of the update and the url mentioned in the update. Besides, the item contains the time (*2010-08-30T17:16:15Z*) when the update was posted, the user (*Daniel Lemire*) who posted the update, and the social network identifier (*Twitter*) of the update.

Data Type	Data Description
friends	The friends of the user on the different SNSs
user feed	Facebook user feed, Twitter tweets, FourSquare checkins, and LinkedIn user feed of the user
home feed	The updates by the friends of the user on the different SNSs

Table 5.2: Uniform data types

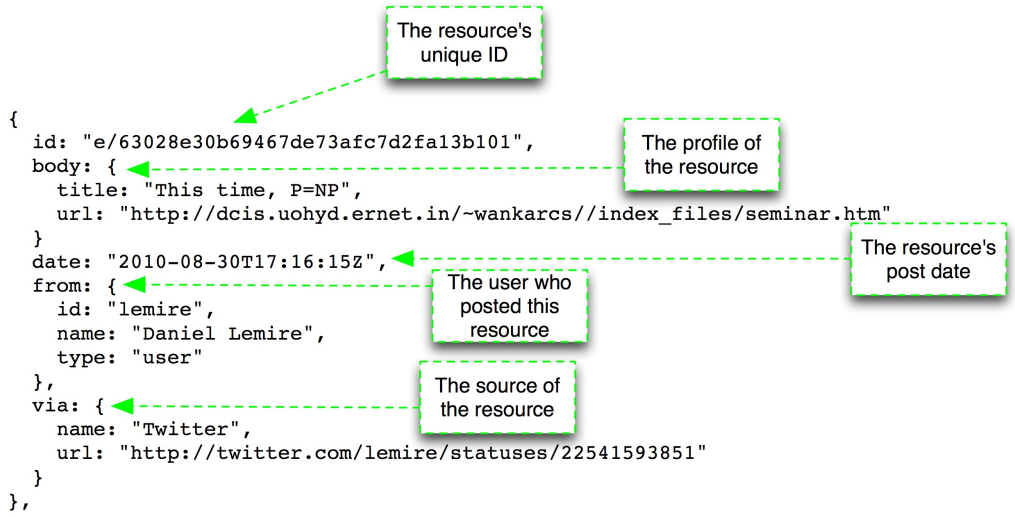


Figure 5.1: A sample encoded item in user feed

Each item in the user feed and home feed are encoded into the agent's episodic memory, indicating an interaction instance between the user and an update, together with the document, friends, time, location information embedded in the update. Based on such information, we will formalize the data representation in the next subsection.

5.1.2.2 Notions and Definitions

A social platform, s, is a social media service that the user engaged in. A contact, c, is the user's friend on social platforms. The set of user contacts forms cs. A location, l, is a checkin declared by the user. An update, d, is a status update posted by the user on a s. An episode e is defined as an update d posted by user u at time t on social platform s, optional involving the user's contacts $\mathcal{F}_u$ and the user's location l.

$$e : (d, u, t, s, \mathcal{F}_u, l) \quad \text{(Eq. 5.1)}$$

The type of an update, $type_d$, is the user's behavior type with regards to an update. E.g., *create, share, comment, bookmark*.

5.1.2.3 Consolidating Episodes

The episodes are ordered by the time when they were created. To support other contextual retrieval and exploration, the episodes are consolidated into semantic memory based on their features and social features. The classification algorithms described in Chapter 4 were used to consolidate the episodes. At the meantime, using the information obtained from the episodes, we build the interests model for individual user using the user interests modeling algorithms described in Chapter 6.

By combining the episodic and semantic information about each document, we obtained a full list of attributes with regards to the documents, as shown in Table 5.3. Among the list of attributes in the table, only the *categories* attribute comes from the semantic memory; all the other attributes are from the episodic memory. The combination provides the complete information for an episode and will be used as clues for retrieval.

5.1.3 Modeling Task Environment

Adomavicius and Tuzhilin [132] defined context in recommendation system as *any information or conditions that can influence the perception of the usefulness of an item for a user*. In this section, we will find types of context for web based task environments. It is observed that the information workers are relying on the web to accomplish their tasks and information seeking activities. For example, some users use Twitter to obtain the latest news that he is interested in; other users use Gmail to process business and personal communication with partners; some other users use search engines to seek information. When working on such tasks, they may realize (or have forgotten) that they've collected some valuable information related to their current task environment. Or, they knew that something they've done in the past are related to the current task, but have no clue how and where to find the episodes. In this section, we study the common web based task environments and their associated features.

Attribute	Description
content	the $TF \cdot iDF$ representation of an update and the content of the url embedded in the update
categories	the class labels of the update assigned in the semantic memory
document length	the number of words in the update
document type	the file extension of the document (e.g., html, pdf, doc, and etc
action	the user's action on the document (e.g., share, like, comment, bookmark)
social network	the social network where the document is from
timestamp	the time when the document was added
location	the geolocation where the user interacted with the document (may not available for all documents)
domain	the domain name where the document is from
feed source	which feed does the document from, user feed or home feed

Table 5.3: Document attribute list

5.1.3.1 Characterizing Task Environment

Although the web based task environments are quite different from each other, some of the elements are commonly shared among them. In this subsection, we identify the common features in web based task environment which can be utilized for contextual retrieval. Because the user's activities in the task environment is a special case of user's information behaviors, we suppose that a task environment, *taskenv*, can be regarded as a special kind of episode, or a series of episode. Therefore it shares some common elements with an episode. It is time to describe the features in *taskenv*:

- the domain of *taskenv*, denoted as $s_{taskenv}$, is the root domain of the current webpage. Its value could be, but not limited to, one of the root domain of the user data sources. This attribute implies the type of the task and is useful for contextual matching. For example, in the task of composing email using Gmail[6] service, $s_{taskenv}$ = "*www.gmail.com*".

[6] `https://www.gmail.com`

- the content in $taskenv$, denoted as $d^i_{taskenv}$, is the textual content identified in the current webpage based which to calculate the context relevance. All the visible texts can be considered as candidate content features, whereas the most significant content can be captured by tracking the location of the mouse cursor and using the text surrounding the mouse cursor as the latest content. Content feature is special in that it is continuously changing. We will describe it in details in the next subsection. In the email composing task, the content is the mixture of the *title* and *body* of the email.

- the other users in $taskenv$, denoted as $\mathcal{F}_{taskenv}$, is the list of other users involved in the task. This feature is quite useful as clue for people based re-finding. It enables cross-social-networks user matching to find out all the documents posted by the same friend. In the email composing task, $\mathcal{F}_{taskenv} = \{$the list of recipients of the email$\}$.

- the time when the user does the task, denoted as $t_{taskenv}$, is the date and time duration of the task. This attribute is useful for temporal feature analysis and can be used for temporal pattern based matching.

- the location of the user, denoted as $l_{taskenv}$, is the physical location where the user executes the task. As users usually perform different type of tasks at different locations, it indicates the type of the task. This attribute can be used for geolocation analysis for location based matching.

- the user self, denoted as u, is the information about the current user.

Based on the elements described above, the context in a task environment can be formalized as shown in Eq. 5.2. The set of context is formalized as shown in Eq. 5.3

$$context^i_{taskenv} : (d^i_{taskenv}, u, t_{taskenv}, s_{taskenv}, \mathcal{F}_{taskenv}, l_{taskenv}) \quad \text{(Eq. 5.2)}$$

$$taskenv = \{context^i_{taskenv} | i \in N\} \quad \text{(Eq. 5.3)}$$

where N is the total number of context identified in $taskenv$.

5.1.3.2 Modeling Incremental Context Switching

One of the significant differences between the context defined in the task environment and the context as defined in web search is that the information need in the task environment is expressed as a series of queries, rather than a single query. Furthermore, the context in the task environment is characterized by different dimensions of context and the mixing of static context and dynamic context. Therefore, when determining each query for individual retrieval instance, the weights of context used in antecedent queries is modeled by a decay factor.

$$\begin{aligned} context^{i}_{taskenv} &= 1 * context^{i-1}_{taskenv(static)} \\ &+ decay * context^{i-1}_{taskenv(dynamic)} \\ &+ (1 - deday) * context^{i}_{taskenv(dynamic)} \end{aligned} \quad \text{(Eq. 5.4)}$$

where the static elements of the task environment is $(u, t_{taskenv}, s_{taskenv}, cs_{taskenv}, l_{taskenv})$, whereas the dynamic element is $d^{i}_{taskenv}$.

5.1.4 Retrieval Model

Given the definitions of episode e and the set of contexts in the task environment *taskenv*, the objective of the retrieval process is to activate the most *context relevant* documents and display them non-intrusively in the user's task environment. The ranking of episode e given task environment *taskenv* can be measured by the similarity between e and *taskenv*, as denoted in Eq. 5.5. In this section, we first describe a linear retrieval model which incorporates the features of the documents and the features of the contexts in the task environment. Furthermore, we describe how the user can refine the retrieval results by filtering. Finally, we develop a learning algorithm to incorporate user feedback into the re-finding system.

$$ranking(e|context^{i}_{taskenv}) = sim(e, context^{i}_{taskenv}) \quad \text{(Eq. 5.5)}$$

5.1.4.1 Episodic and Semantic Features Based Ranking

In this subsection, we describe each of the similarity measures with regards to the common features in the document and the context. Based on the measures derived, we develop a linear ranking model to rank documents based on their similarity with a context.

Content Relevance The content relevance measures the textual similarity between $context^i_{taskenv}$ and each e, which is denoted as $content\_sim(context^i_{taskenv}, e)$. We represent the text content of $context^i_{taskenv}$ and e as term vectors and use the $TF \cdot iDF$ weighting scheme to calculate $content\_sim(context^i_{taskenv}, e)$. The detailed calculation was described in Chapter 4.

Categorical Relevance The categorical relevance measures the categorical similarity between the information object in the context and the information object in an episode, e , which is denoted as $cat\_sim(context^i_{taskenv}, e)$ as defined in Eq. 3.6. Because the category taxonomy used in our work is a hierarchical tree, we extend Eq. 3.6 by utilizing the concept similarity measure introduced by Ganesan [133] to calculate the similarity between two concepts, c_1 and c_2, in the hierarchical tree, as shown in Eq. 5.7.

$$\delta(c_1, c_2) = \frac{2 \times depth(LCA(c_1, c_2))}{depth(c_1) + depth(c_2)}, \qquad \text{(Eq. 5.6)}$$

where c_1 and c_2 are two categories of a tree, $LCA(c_1, c_2)$ is the lowest common ancestor of c_1 and c_2, and $depth(c_1)$ and $depth(c_2)$ are the depth (from root) of these two categories in the tree. Suppose each information object is classified into a single category, and we denote the category for $context^i_{taskenv}$ and e as $c_{context^i_{taskenv}}$ and c_e respectively, then the categorical similarity as $cat\_sim(context^i_{taskenv}, e)$ is calculated as:

$$cat\_sim(context^i_{taskenv}, e) = \delta(c_{context^i_{taskenv}}, c_e) \qquad \text{(Eq. 5.7)}$$

Social Relevance The social relevance measures the intersection between users involved in $context^i_{taskenv}$ and users involved in e, denoted as $social\_sim(context^i_{taskenv}, e)$ as defined in Eq. 3.3.

Geographical Relevance The geographical relevance measures the correlation between the geolocation of (or mentioned in) the task environment $taskenv$ and the geolocation of (or mentioned in) the episode e, denoted as $geo\_sim(context^i_{taskenv}, e)$ and calculated with Eq. 3.4.

Temporal Relevance The temporal relevance measures the relative strength of retention between two episodes. An intuitive solution is to assign higher weights to more recent episodes. We consider the time of the context as reference timestamp, then the relative decay of the document in e, denoted as $decay'(e, context^i_{taskenv})$, can be calculated with Eq. 3.5.

Retrieval Model Based on the relevance measures introduced above, we derived a contextual episodes retrieval model by incorporating those relevance measures linearly with an extra decaying function.

$$\begin{aligned} score(taskenv, e) = {} & decay'(doc, context^{i}_{taskenv}) \cdot \\ & (w_1 \cdot content\_sim(taskenv, e) \\ & + w_2 \cdot social\_sim(taskenv, e) \\ & + w_3 \cdot temporal\_sim(taskenv, e) \\ & + (1 - w_1 - w_2 - w_3) \cdot geo\_sim(taskenv, e)) \end{aligned} \tag{Eq. 5.8}$$

The top-k documents with the highest ranking scores are selected and displayed to the user.

5.1.4.2 Interactive Filtering

The default ranked list of documents has provided the foundations for the retrieval model. However, it is very hard to capture the user's most active clues for retrieval in specific context. The user might have very specific contextual clues for the retrieval. Therefore, we provide interactive filtering in the user interface to assist the user to select the context actively.

Filtering by document meta attributes The user may rely on the document length and document type to explore the documents in the retrieval results to further refine the query. If the user knows the document type or approximate length, then this filtering will help user in locating the documents efficiently.

Filtering by user action The user's action space — share, like, comment, and bookmark, provides strong clues for the user to explore. If the user has such clues in mind, then this filtering can reduce the search space for locating the resource.

Filtering by information source and domain If the user knows the information sources of the document, or the domain of the document, then the search space can be reduced.

Filtering by timestamp and location Timestamp and location at which the user interacted with the document are also string clues. Time based filtering can reduce the search space to very specific time interval patterns.

Filtering by feed source The fact whether the documents are seen in the user feed or the home feed can help reduce the search space.

Each of the individual filter described above can help locating the target documents to some extent. Moreover, the combination of two and more filters can further improve the re-finding efficiency.

5.2 Case Study on Social Networks

People use a variety of social networking services to collect and organize web information for future reuse. However, when such contents are needed as reference to reply a post in an online conversation, the user may not be able to retrieve them with proper clues or may even forget their existence at all. In this section, we study this problem in the online conversation context and investigate how to automatically retrieve the most context-relevant previously-seen web information without user intervention. We propose a Context-aware Personal Information Retrieval (CPIR) algorithm, which considers both the participatory and implicit-topical properties of the context to improve the retrieval performance. Since both the context and the user's web information are usually short and ambiguous, the participatory context is utilized to formulate and expand the query. Moreover, the implicit-topical context is exploited to implicitly determine the importance of each web information of the targeting user in the given context. The experimental results using real-world dataset show that CPIR can achieve significant improvements over several baselines.

5.2.1 Introduction

Information reuse and re-finding are very common behaviors in both desktop based personal information management systems [134][135] and Web search engines [136]. Many information related activities involve referring to and integrating previously-seen information. For example, when replying to questions on question answering websites or posts on Social Networking Services (SNSs), the user may need such previously-seen information as reference to support reuse. Previous studies [134] have shown that 58%-81% of web pages accessed are re-visits to pages previously seen. Traditionally, people organize and store the interested information in their desktop and then use classical information retrieval technologies to retrieve them for reuse. With the exponential growth of Web 2.0 services, people tend to utilize SNSs to collect and share previously-seen information [137][93][138]. Typical examples of such services include microblogging (e.g., twitter), social network (e.g., Fackbook) and social bookmarking (e.g., Delicious).

Personal information indexing and re-finding have been studied extensively in the literature of extending human memory [139]. Staff I've Seen [134] was developed to help information re-use on the desktop by indexing the documents that the user has seen and utilizing various contextual clues such as date, document type, and author to assist retrieval in the search interface. SenseCam [140] captures people's everyday life using a wearable camera to support people's memory for the past and personal events. However, the scope of these work are limited to desktop or single device. The information re-finding problem in social networks is very challenging due to the information fragmentation problems [141]. As the user's data is distributed on different web sites, the platform diversity and heterogeneity degrade data connectivity and cleanness [142]. Moreover, the social networks also bring new features via so called collective intelligence, which are significantly different fromtraditional personal information management systems. For instance, the social tagging generated by the community in Folksonomy [143] and knowledge aggregated in community question answer websites [144] have provided extra metadata about entities in which each individual user involved. Finally, evidence shows that the user's documents on different platforms share a common vocabulary on multiple social bookmarking systems [145][146] and multiple social network sites [102]. Therefore, the data needs to be aligned before integration.

We use Personal Web Information (PWI) to indicate the previously-seen information that have been collected and shared by a user on different SNSs. Practically, it is a very important and challenging task to make connections between the user's context and his PWIs automatically and implicitly [134][1], especially when the PWIs spreads across multiple SNSs. For example, a film lover, who has reviewed a classical movie on Facebook a few years ago, can provide potentially valuable comments to his friend's recent post about that movie on Twitter. However, the user may not be able to retrieve the review with proper clues or may have totally forgotten it.

In this section, we formulate the problem of automatic personal web information retrieval. The study addresses how to build a query by implicitly capturing the individual user's *information need* in the on-line conversation context and how to retrieve the user's most relevant PWIs for information reuse. However, the task is non-trivial. Although the social aggregation services (SASs) can gather the web information of individual users from different SNSs and the conversations in SASs provide ideal environment to simulate user context, there still exists the following challenges. First, the posts in the conversations are usually short and ambiguous [147], which cannot provide enough clues for PWIs retrieval. Other than that, since the users' documents in SNSs are noisy and complex [84][148], the pairwise relevance measuring alone cannot capture the similarities between the query generated

from the context and the user PWIs. Figure 5.2 shows an example conversation[7] extracted from FriendFeed .

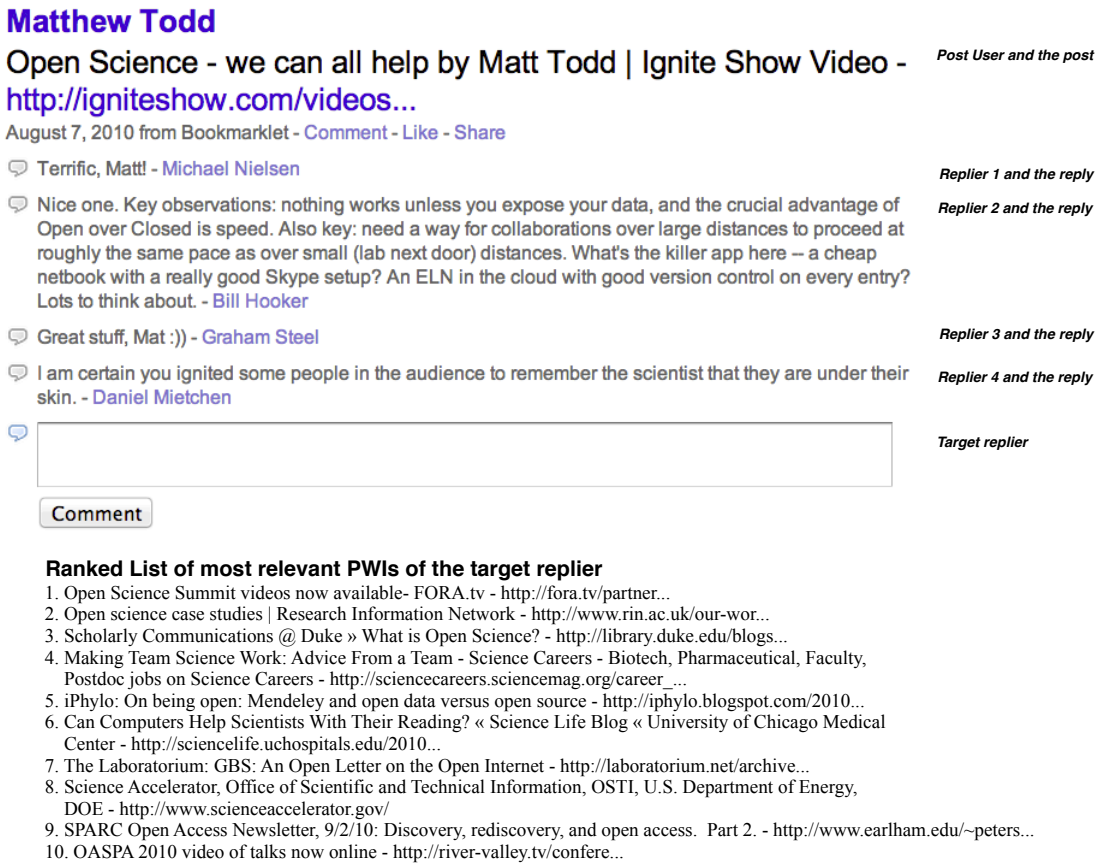


Figure 5.2: A conversation on FriendFeed, with the ranked list of top 10 most relevant PWIs of the targeting replier from our algorithm. In this conversation, *Matthew Todd* wrote a post on "open science" and four repliers have responded to the post. Our objective is to automatically retrieve context-relevant documents from the targeting replier's own PWIs before the targeting replier composing the response message for reuse. The most relevant ones retrieved by our algorithm are shown at the bottom.

Through data analysis described in follow sections, it is found that (i) the replies and PWIs of all users participating in a conversation provide additional information that may be used to expand the query. For example, a football fan usually posts football related news on different SNSs. Therefore, in a football related conversation, all football related documents of all repliers can be mined to expand the

[7]The conversation is from http://friendfeed.com/science-2-0/ea2aadbb/open-science-we-can-all-help-by-matt-todd-ignite

context; and (ii) the users (e.g., the initiator, the existing repliers, and the targeting replier) involved in the same conversation share common interests — the topic of the conversation. From this perspective, there exists a subset of the PWIs of all participating users, which is related to the topic. As a result, making use of these implicit relationships make it possible to reveal the subset, and thus obtain the list of relevant PWIs for the targeting user.

Based on the analysis, a two-step ranking approach called CPIR is developed to solve the problem and obtain better retrieval results. We treat an online conversation as a session and the post created by the initiator in a session as the initial query. First, the participatory context, which includes the replies and PWIs of all participating users, is exploited. KL-Divergence [149] model with customized smoothing method is developed to find the most relevant replies and PWIs of the existing repliers to expand the initial query. Moreover, the implicit-topical context, in which a graph-based algorithm is employed to deduce the implicit relationships among the PWIs of all participating users, is introduced to rank those documents so that documents with higher similarities with each other are assigned with higher scores. The relevance score of each PWI measured by its similarity with the query and the importance score of each PWI obtained from the graph-based algorithm are combined to produce the final ranking score for each PWI of the targeting replier subsequently. In our study, FriendFeed[8], a well-known social aggregation service which aggregates users' PWIs from different SNSs, is used as the testbed . Given a replier who is trying to reply a post in a conversational session in FriendFeed, the concerned problem is how to efficiently capture the context of the session and retrieve the most relevant PWIs of the targeting replier for reuse.

5.2.2 Context-aware Personal Information Retrieval

5.2.2.1 Problem Statement

Given a session and the targeting replier for whom we will make recommendation, the context-aware personal information retrieval problem studied in this section is to model the context of the session and generate a query so as to retrieve the most relevant PWIs from the user's document collection.

For clarity, the key notations used in this section are listed in Table 5.4. In particular, the definition of *Session* is as follows:

Definition 5.19 *A **Session** ($\mathcal{S}$), is an on-line conversation with an initial post p and a set of replies $\mathcal{R} = \{r_1, \cdots, r_i, \cdots, r_n\}$.*

[8]FriendFeed: www.friendfeed.com

Examples of $\mathcal{S}$ include conversations in SNSs, such as a tweet posted on Twitter with replies from fellow followers, or a question posted on a question answering website with answers from other users.

Table 5.4: Symbols and Definitions

Symbol	Definition
$\mathcal{S}$	Session
p	Initial post in $\mathcal{S}$
Q	Expanded query
$\mathcal{R}$	Collection of existing replies, $\mathcal{R} = \{r_i\}_{i=1}^{n}$
$\mathcal{U}$	User collection within $\mathcal{S}$, $\mathcal{U} = \{u_p, u_1, ..., u_n, u_t\}$
D_i	PWIs set of i-th replier, $D_i = \{d_j^i\}_{j=1}^{m_i}$
$\mathcal{D}$	Collection of PWIs of all the users involved in $\mathcal{S}$, $\mathcal{D} = \{D_p, D_1, ..., D_n, D_t\}$

In a Session ($\mathcal{S}$), p denotes the initial post. $\mathcal{U}$ indicates the set of users involved in $\mathcal{S}$, i.e., $\mathcal{U} = \{u_p, u_1, \cdots, u_i, \cdots, u_n, u_t\}$. u_p is the creator of p, u_t is the targeting replier to whom we will make recommendations, and u_i ($u_i \notin \{u_p, u_t\}$ and $i = 1...n$) is an existing replier. $\mathcal{R} = \{r_1, \cdots, r_i, \cdots, r_n\}$ is the set of existing replies, assuming that each user gives solely one reply. $\mathcal{D} = D_p \cup \{D_{1i}\}_{i=1}^{n} \cup D_t$ is the entire collection of PWIs of all the users within $\mathcal{S}$, where D_p, D_t, and D_i ($D_i \notin \{D_p, D_t\}$) are the sets of PWIs of u_p, u_t, and u_i respectively. In addition, $D_i = \{d_j\}_{j=1}^{m_i}$ is the PWIs of u_i, where m_i is the number of u_i's PWIs.

The content of all the documents in $\mathcal{S}$ are represented by the Vector Space Model [150], i.e., $\mathbf{v}_d = [w_{1,d}, \cdots, w_{k,d}]^T$, in which each term is weighted by its *tfidf* score [151].

5.2.2.2 Model Overview

Normally, information retrieval systems use solely the terms in the query and the document collection to decide the relevance, while the information in the specific context is ignored [152]. Automatic query generation [153] is the task of identifying the most representative texts in the context for information retrieval. Query expansion is the process of enriching the original query by adding extra content words deduced from the context [154][155]. In order to implicitly query relevant resources based on user's current computing activities, [19][156] use *tfidf* based *cosine* similarity to rank candidate resources based on their similarities with the document in the context. In our work, the query is generated by considering the post, replies to the

post, and the PWIs of all the participating users, which captures richer information regarding both the context and the user. Retrieving relevant documents from a collection of candidate documents can be considered as a classification problem or a ranking problem. We treat the personal information retrieval problem as a ranking problem in which two properties of relevance are explored and combined.

CPIR is mainly decomposed into two sub-steps: (i) query formulation and expansion: building query by extracting important terms from the session; and (ii) PWIs ranking: extracting the most relevant PWIs of the targeting replier according to the query. The overview of CPIR is illustrated in Figure 5.3.

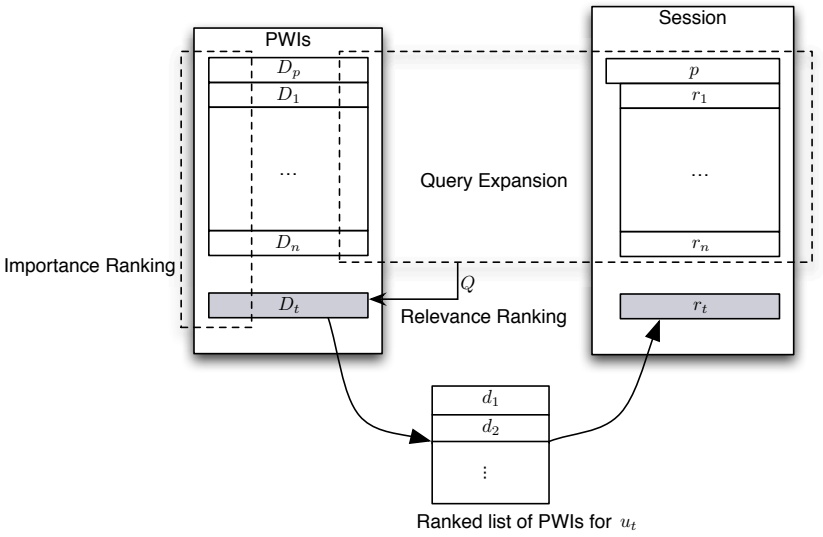


Figure 5.3: Diagram of Context-aware Personal Information Retrieval Framework. The framework takes a session and users' PWIs as input. It works by first building a query Q from the session using the information in both the session and users' PWIs. A graph-based algorithm is employed to derive the importance scores for each PWI of u_t, which is then combined with the relevance scores to obtain the final ranked list of documents for u_t.

In the first step, the participatory context is used to formulate and expand the query by considering both the replies and the PWIs of all participating users. An intuitive idea is to use the initial post p and the existing replies $\mathcal{R}$ as the context to build the query since they are the basic available context information in $\mathcal{S}$. However, the posts in SNSs are usually short and ambiguous, which are not sufficient to characterize different properties of the session. Therefore, the PWIs of the creator and the existing repliers are utilized to obtain richer information. The main reason is that users participating in the same session usually share common interests related

to p, and they might have posted similar documents in other SNSs. Therefore, the PWIs of the creator and the existing repliers can be used as the complementary context information.

In the second step, the shared interests among the targeting replier, the creator, and the existing repliers are considered, which forms the implicit-topical context. One of the common interests of all the participating users is the topic of the conversation they are involved in. By implicitly inferring the subset of documents on the topic in an unsupervised manner, the relevant PWIs of the targeting user can be collected.

In the remaining parts of this section, we will describe the two steps in details and then combine the ranking results from these two steps to obtain the final ranking scores for each PWI of the targeting user.

5.2.2.3 Utilizing Participatory Context for Query Expansion

The context of the session $\mathcal{S}$ consists of an initial post p, existing replies, and the PWIs of the creator and the existing repliers. To accurately model the context of session $\mathcal{S}$ so as to build a comprehensive query for covering the different properties of context, we need to extract as many clues as possible from them.

To model the context, the query Q is built by modeling the session in two levels. Formally, the initial p is treated as the basic query. We first combine the replies of existing repliers with p, since these replies are the responses to p and thus can provide extra information about $\mathcal{S}$. As different replies have different levels of correlation with p, the replies are weighted according to their similarities with p. The expanded query is calculated as follows:

$$\mathbf{v}_{Q_{p+R}} = \alpha \mathbf{v}_p + (1-\alpha) \sum_{r_i \in \mathcal{R}} sim(\mathbf{v}_p, \mathbf{v}_{r_i}) \cdot \mathbf{v}_{r_i} \qquad \text{(Eq. 5.9)}$$

where α ($0 \leq \alpha \leq 1$) is a trade-off parameter to control the contribution of replies. $sim(.,.)$ is the similarity metrics to measure the relevance between the initial post p and the replier r_i ($r_i \in \mathcal{R}$).

Among the different similarity measures, it has been shown that probabilistic methods like KL-divergence [157] can achieve better results than vector space based measures [158][159], especially for short text [160] like the PWIs discussed in this section. However, as the vocabulary in PWIs is sparse, smoothing techniques are always introduced to take the entire vocabulary into consideration to compare two distributions. We introduce the translation-based language model [161] with WordNet[9] as an external source to expand the documents before calculating their similarities.

[9]http://wordnet.princeton.edu/

The KL-divergence between p and r_i, $D_{KL}(\mathbf{v}_p||\mathbf{v}_{r_i})$ is calculated as follows:

$$D_{KL}(\mathbf{v}_p||\mathbf{v}_{r_i}) = \sum_{w_{i,p}} P'(w_{i,p}|\mathbf{v}_p) \log \frac{P'(w_{i,p}|\mathbf{v}_p)}{P'(w_{i,p}|\mathbf{v}_{r_i})} \quad \text{(Eq. 5.10)}$$

in which $P'(w|\mathbf{v})$ is the expanded distribution. $P'(w|\mathbf{v})$ is calculated as follows:

$$P'(w|\mathbf{v}) = \sum_{w' \in \mathbf{v}} f(w'|w)P(w'|\mathbf{v}) \quad \text{(Eq. 5.11)}$$

where $P(w'|\mathbf{v})$ denotes the *tfidf* score of w' in $\mathbf{v}$ and $f(w'|w)$ is the translation probability of word w to word w' calculated using WordNet sense similarity.

The similarity between p and r_i can be calculated as:

$$sim(\mathbf{v}_p, \mathbf{v}_{r_i}) = e^{-|\frac{1}{2}(D_{KL}(\mathbf{v}_p||\mathbf{v}_{r_i}) + D_{KL}(\mathbf{v}_{r_i}||\mathbf{v}_p))|} \quad \text{(Eq. 5.12)}$$

To further expand the query, we also consider the PWIs of the creator and the existing repliers. However, not all of them are incorporated into query Q which otherwise will result in a very long query. Instead, the top k most relevant ones are selected to expand $Q_{p+\mathcal{R}}$. The expanded query can be represented as,

$$\mathbf{v}_Q = \beta \mathbf{v}_{Q_{p+\mathcal{R}}} + (1-\beta) \sum_{d \in D_i} sim(\mathbf{v}_d, \mathbf{v}_{Q_{p+\mathcal{R}}}) \cdot \mathbf{v}_d \quad \text{(Eq. 5.13)}$$

where β $(0 \leq \beta \leq 1)$ is a trade-off parameter and $D_i \in \mathcal{D}$ but not D_t.

5.2.2.4 PWIs Ranking

Utilizing Implicit-Topical Context for Importance Ranking We have shown thatAs aforementioned, the users involved in the same session $\mathcal{S}$ share common interests, at least including the topic of S. We employ a Markov random walk model to infer the implicit relationship between the web information of the users and find the subset of PWIs which are more relevant to the topic of the session.

Let $G(N, E)$ be a graph of documents where $N = \{n_1, \cdots, n_{|N|}\}$ is a set of vertices and $E = \{e_1, \cdots, e_{|E|}\}$ is a set of edges. In G, a vertex $n_i \in N$ is an PWI $d \in D_i$ ($D_i \in \mathcal{D}$, and $D_i \neq D_p$). The transition probability matrix of G is represented by $P = [p_{ij}]$, in which each transition probability from node n_i to node n_j is given by,

$$p_{ij} = \frac{sim(\mathbf{v}_{n_i}, \mathbf{v}_{n_j})}{\sum_k sim(\mathbf{v}_{n_i}, \mathbf{v}_{n_k})} \quad \text{(Eq. 5.14)}$$

where $sim(.,.)$ is defined as Eq. (Eq. 5.12).

To overcome the "dangling link" while conducting a random walk on graph G, the similarity scores between the generated query Q and each PWI in G as the reset probability are used to allow the random walk process jumping from a node to an arbitrary node with a small probability. For node n_i, the reset probability x_i is calculated as follows:

$$x_i = sim(\mathbf{v}_{n_i}, \mathbf{v}_Q) \quad \text{(Eq. 5.15)}$$

To make the sum of all the elements in $\mathbf{x}$ equal to 1.0, each $x_i \in \mathbf{x}$ will be normalized by $x_i = \frac{x_i}{\sum_j x_j}$. With the transition matrix P and the reset probability vector $\mathbf{x}$, the stationary eigenvector π can be computed iteratively using power method [162].

Final Ranking for User PWIs The ranking obtained from the random walk model denotes the importance of the document in the collection of PWIs. The similarity between the expanded query Q and each document measures the relevance of the document for the session. We use a linear combination of these two ranking scores to obtain the final score for each candidate $d_i \in D_t$, which is calculated as follows:

$$Score(d_i) = \lambda cosine(d_i, Q) + (1 - \lambda)\pi_i \quad \text{(Eq. 5.16)}$$

where λ ($0 \leq \lambda \leq 1$) is a combining parameter. Note that *cosine* measure is used to calculate the similarities between the Q and the user document d_i. It is primarily due to the fact that Q is usually quite longer than PWI since it is the combination of multiple PWIs, *cosine* is more suitable in this case. Finally, the top ranked PWIs ($d_i \in D_t$) are selected as the recommendation results to the targeting replier. The algorithm is described in Algorithm 4.

5.2.3 Experiments and Analysis

A set of experiments were designed to evaluate the retrieval algorithm. In this section, we describe the analysis on the dataset and discuss the performance of the proposed CPIR algorithm by comparing it with other baselines.

5.2.3.1 Experiment settings

Data Description Celli et al. [163] provided a FriendFeed dataset[10], which was collected by monitoring the data stream on FriendFeed from 01/08/2010 to 30/09/2010.

[10]FriendFeed Dataset: http://larica.uniurb.it/sigsna/data/

input : $p,\ R,\ \mathcal{D} = \{D_p, D_t, D_1, \cdots, D_n\}$
output: Document set $D_t^* = \{d_i\}_{i=1}^{K},\ d_i \in D_t$, which contains the top-k relevant documents with regarded to the topic of Session $\mathcal{S}$

Query Expansion:
$v_{Q_p} \leftarrow v_p$;
foreach $r_i \in \mathcal{R}$ **do**
  Compute $sim(v_p, v_{r_i})$ according to Equation (Eq. 5.10), (Eq. 5.11) and (Eq. 5.12);
  $v_{Q_{p+R}} \leftarrow \alpha v_{Q_p} + (1-\alpha) sim(v_{Q_p}, v_{r_i}) \cdot v_{r_i}$;
end
foreach $r_i \in \mathcal{R}$ **do**
  $D_i^* \leftarrow \arg\max_{X, d \in D_i} sim(v_{Q_{p+R}}, v_d)$;
  foreach $d \in D_i^*$ **do**
    Compute $sim(v_{Q_{p+R}}, v_d)$ according to Equation (Eq. 5.10), (Eq. 5.11) and (Eq. 5.12);
    $v_Q \leftarrow \beta v_{Q_{p+R}} + (1-\beta) sim(v_{Q_{p+R}}, v_d) \cdot v_d$;
  end
end
Importance Ranking:
Construct the probability matrix $\mathbf{P}$ according to Equation (Eq. 5.14);
Construct the reset probability vector $\mathbf{x}$ according to Equation (Eq. 5.15);
Compute iteratively $\pi^{(t+1)} = \gamma \mathbf{P} \pi^{(t)} + (1-\gamma)\mathbf{x}$;
Final Ranking:
foreach $d \in D_t$ **do**
  $\text{Score}(d_i) = \lambda \text{cosine}(d_i, Q) + (1-\lambda)\pi_i$;
end
$D_t^* \leftarrow \arg\max_K \text{Score}(d),\ (d \in D_t)$;
Return D_t^*;

Algorithm 4: Context-aware Personal Information Retrieval (CPIR)

Note that the set of comments for a single entry only contains the initiator's comments , while the comments provided by other users are not included. In order to obtain complete conversations, we re-extracted all the conversations in the dataset via the FriendFeed API[11].

Table 5.5 summarizes the basic information about the conversations and their replies, including the number of repliers involved in these conversations and the aggregated PWIs for those users. From these conversations, we select the post-reply pairs written in English[12] and the repliers of which have at least 50 PWIs.

Table 5.5: Basic Statistics of the FriendFeed Dataset

Number of conversations	56,460
Number of replies	749,369
Number of repliers	14,443
Number of PWIs	637,320

To obtain manual annotation results for evaluation , we randomly sampled 105 post-reply pairs. Those replies are posted by 73 unique users. Each user has 316 PWIs on average. Two volunteers manually labeled the 23,046 PWIs of the repliers as relevant or irrelevant for the given conversations.

Before applying the models on the documents, tokenization and part-of-speech tagging are performed to eliminate terms with non-functional tags. In addition, stop words are removed and terms are stemmed using Porter Stemmer [164].

Data Analysis Figure 5.4(a) shows the distribution of the number of repliers per conversation. Figure 5.4(b) shows the distribution of unique replies per conversation. We can observe that 98% of conversations have at least 3 replies and 78% of conversations have at least 3 unique repliers. This confirms the feasibility of utilizing the conversations to model the task environment so as to retrieve historical information.

The distribution of the number of aggregated services for the repliers is depicted in Figure 5.5(a), which shows that more than 65% users use at least two services. It confirms that the retrieving document corpus is extracted from diverse information sources. The distribution of the number of aggregated PWIs for the users is depicted

[11]FriendFeed API: http://friendfeed.com/api/

[12]The language detection tool: guess-language (https://github.com/dsc/guess-language) is used for language detection

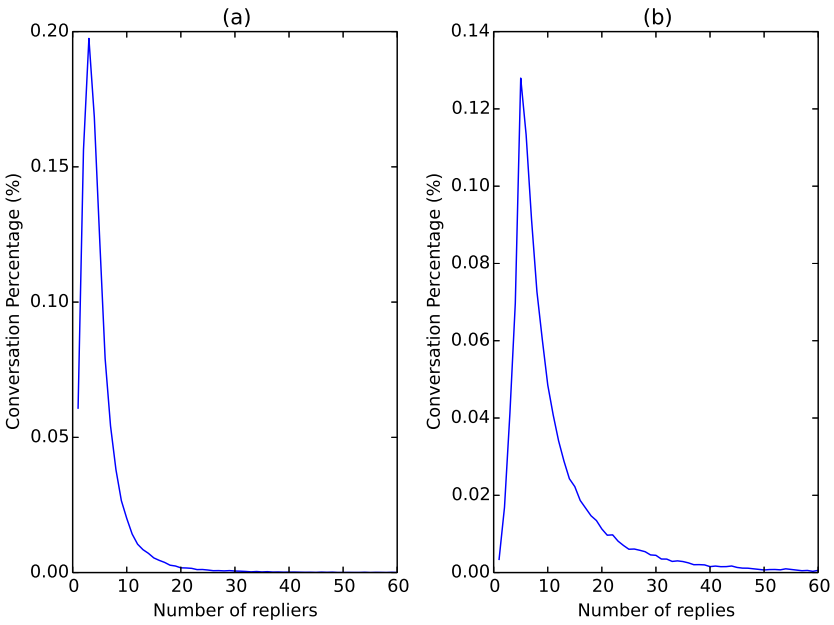


Figure 5.4: Conversation engagement — (a) Distribution of the number of repliers per conversation in the corpus, (b) Distribution of the number of replies per conversation in the corpus.

in Figure 5.5(b). It can be observed that 63% users posted more than 10 PWIs. These observations motivate us to use the PWIs of the users in the conversation to expand the query and improve the retrieval performance.

The top services by total number of PWIs are depicted in Figure 5.6. It can be observed that the majority portion of PWIs are aggregated from the popular services like FriendFeed, Twitter, and Google Reader.

Evaluation Metrics The performance of our algorithm is evaluated against six widely used metrics — precision at rank 1 (P@1), precision at rank 5 (P@5), recall, mean average precision (MAP), mean reciprocal rank (MRR), and F-measure [89]. The top 50 retrieved PWIs for each post-reply pair are used for evaluation.

Baselines Our algorithm is compared with three baselines. They include:

- using the initial query to build the session query, denoted as *post*;
- using $p{+}R$ to build query, with *cosine* and KL to measure similarities between two vectors, denoted as *prcos* and *prkl* respectively;

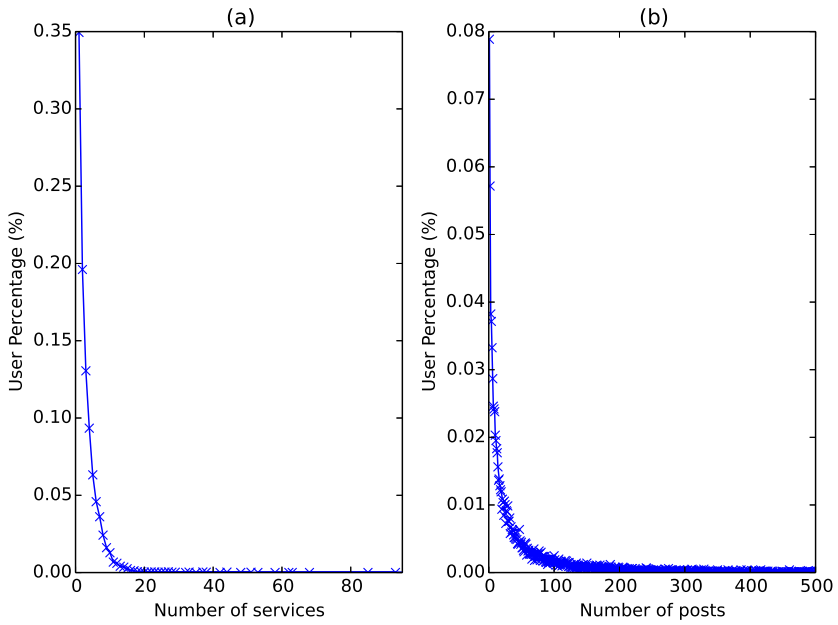


Figure 5.5: User engagement — (a) Distribution of the number of services per user in the corpus, (b) Distribution of the number of PWIs per user in the corpus.

- using $p{+}R{+}D$ to build query, but with *cosine* as similarity measure, denoted as $prcos{-}prdcos$.

Those baselines are compared against the relevance ranking scheme in CPIR, denoted as $\mathsf{CPIR}_{\lambda=1}$.

Moreover, we combine each of the baselines with the importance scores calculated from the random walk model and obtain another set of baselines, which are denoted as $post{+}graph$, $prcos{+}graph$, $prkl{+}graph$, and $prcos{-}prdcos{+}graph$ respectively. These baselines are compared against CPIR.

Table 5.6: Retrieval results of expanding the session query, compared with baselines

Algorithms	P@1	P@5	Recall	MAP	MRR	F-measure
post	0.6476	0.4667	0.7639	0.4395	0.7608	0.2197
prcos	0.6476	0.4914	0.8009	0.4651	0.7720	0.2316
prkl	0.7143	0.5162	0.8263	0.4949	0.8176	0.2390
prcos-prdcos	0.7619	0.5505	0.8272	0.5290	0.8336	0.2414
$\mathsf{CPIR}_{\lambda=1}$	**0.8000**	**0.5562**	**0.8529**	**0.5466**	**0.8620**	**0.2497**

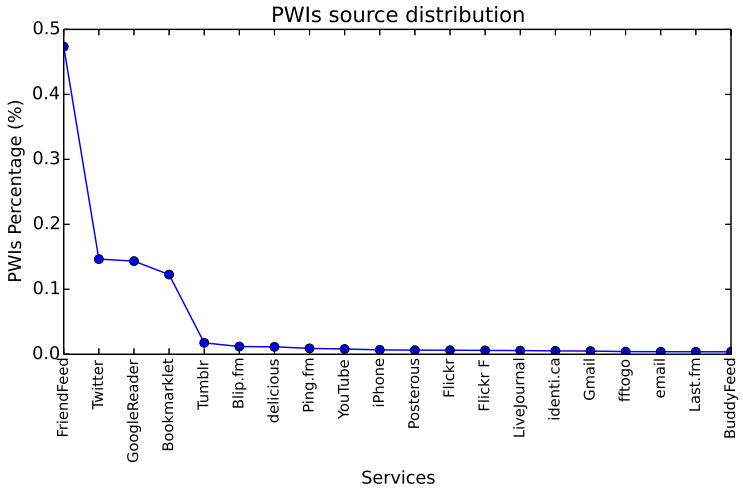


Figure 5.6: Top 20 services by total number of PWIs.

Table 5.7: Retrieval results of combining relevance and importance

Algorithms	P@1	P@5	Recall	MAP	MRR	F-measure
post+graph	0.6571	0.4762	0.7881	0.4618	0.7693	0.2306
prcos+graph	0.6762	0.4933	0.8191	0.4835	0.7857	0.2394
prkl+graph	0.7238	0.5200	0.8457	0.5035	0.8232	0.2471
prcos-prdcos+graph	0.7714	0.5524	0.8489	0.5401	0.8412	0.2494
CPIR	**0.8000**	**0.5619**	**0.8619**	**0.5528**	**0.8618**	**0.2543**

5.2.3.2 Retrieval Performance

Table 5.6 shows the comparison between $\mathsf{CPIR}_{\lambda=1}$ and the corresponding baseline algorithms. $\mathsf{CPIR}_{\lambda=1}$ achieves significant improvement over the baseline methods with respect to all the six metrics. Expanding the initial query with the replies in the conversation enhanced the context clues, while adding selected PWIs further captured the context information. It is also observed that our KL based measure outperforms *cosine* based measure to calculate document similarities.

Table 5.7 compares the retrieval results of CPIR and the corresponding baselines. It can be observed that CPIR, which combines $\mathsf{CPIR}_{\lambda=1}$ and graph ranking, can produce best performance compared with the combination of graph ranking with other baselines. When compared with the corresponding baselines without graph ranking in Table 5.6, graph-based ranking algorithm can find the PWIs with common topics among the users involved in the same conversation so as to further improve the performance. Specifically, CPIR outperforms $\mathsf{CPIR}_{\lambda=1}$ in five out of the six metrics.

Table 5.8: Parameter settings

	Value	Description
α	0.6	p and R combination controller
β	0.6	pr and D_{p+R} combination controller
k	15	number of top PWIs for $\mathsf{CPIR}_{\lambda=1}$
λ	0.75	relevance ranking and importance ranking combination controller

A specific example conversation with the retrieval results obtained from CPIR and two baselines is shown in Figure 5.7. The post and selected replies are shown on the top left, while the manually labeled relevant PWIs of the targeting user are shown on the top right. The retrieval results by *prcos*, *prcos-prdcos*, and CPIR are shown at the bottom, in which the indexes of the matched PWIs are marked in **Bold**. It is observed that CPIR found 6 matched PWIs, while *prcos* and *prcos-prdcos* found 1 and 3, respectively.

We also analyze the distribution of the retrieved documents in different social network sites. The top five social network sites with the largest number of retrieved documents are shown in Figure 5.8. It shows that the number of retrieved documents is proportional to the total number of documents in those platforms.

5.2.3.3 Parameter Settings

Table 5.8 shows the main parameters and their settings in our experiments. The optimal parameter for each variable are obtained by fine tuning. For example, The optimal α in our experiment is 0.6, which implies that the weights of the terms in p is slightly more important than the terms in R. The optimal k is 15, which means the best performance is obtained by only extending pr with selected documents in D_{p+R}.

One of the most important parameter in our experiments is λ, which controls how to combine the ranking scores from the query expansion results with the ranking scores from the random walk model. Figure 5.9 shows the effect of varying λ on MAP with step $\Delta\lambda = 0.02$. The best MAP is obtained by setting λ to 0.75. The optimal settings for other parameters are obtained in similar way.

5.3 User Studies

To evaluate *WebESRA* in real world scenarios, we created a web application to aggregate user data from different information sources via their APIs. The users

Post
ekins older scientists more likely to prefer non collaborative lab notebooks as opposed to wiki jcbradley thoughts acs_boston
Replies
- confounded somewhat by effect of tenure some older scientists are very into open foo but recognize that earlier in their careers they might have been more cautious
- i think it would be hard to get meaningful statistics on this numbers are small and how do you define scientist undergrads don t typically have problem with sharing notebook again have to clearly define what sharing means but most of them don t become scientists by most definitions
- bill makes an excellent point that younger scientists may not have the same choice as older ones
- but this is all going to change no those who begin their career with this technology will always use it in 30 years that will be everyone

Manually labeled relevant PWIs
0. discusses lab notebook requirements and discusses the scrap of paper disallowed mistake acs_boston acsrdf2010 any literature on such
1. biomed central is going to support open data
2. open science microformats initial thoughts jessy s acceptable
3. why i blog because i need thoughts out of my system like with getting things done write it down know it s organized
4. introducing the mcprinciples for open cheminformatics
5. biomed central is going to support open data via
6. dutch intelligence service raises awareness with universities of the risk of espionage opendata science
7. linked open data and pavlova
8. molecular networks is discussing all the benefits of open standards and open source acs_boston are they changing their own north too
9. sharing detailed research data is associated with increased citation rate

Top 10 PWIs retrieved by *prcos*
0. discusses lab notebook requirements and discusses the scrap of paper disallowed mistake acs_boston acsrdf2010 any literature on such
1. jeremy talks about computers and wet lab chemistry acsrdf2010 acs_boston
2. great drawing of benzene but not quite the way you re used to it acs_boston
3. rich showed some usage stats on one new during during his talk acs_boston
4. the life scientist room at cliqset
5. more acs_boston slides online if anyone at acs listens we need central repository for this
6. oh and jeremy is profechem so feel free to contact the speaker directly acsrdf2010 acs_boston but not until the end of his talk
7. lovely regex logo on slides by lezan acsrdf2010 acs_boston
8. leonid shows nice get coffee slide the soap and the semantic version acsrdf2010 acs_boston
9. the list of 128 papers that cite one or two of the main cdk papers

Top 10 PWIs retrieved by *prcos-prdcos*
0. discusses lab notebook requirements and discusses the scrap of paper disallowed mistake acs_boston acsrdf2010 any literature on such
1. jeremy talks about computers and wet lab chemistry acsrdf2010 acs_boston
2. great drawing of benzene but not quite the way you re used to it acs_boston
3. the life scientist room at cliqset
4. molecular networks is discussing all the benefits of open standards and open source acs_boston are they changing their own north too
5. rich showed some usage stats on one new during during his talk acs_boston
6. open science microformats initial thoughts jessy s acceptable
7. acsrdf2010 acs_boston
8. collaborative development of predictive toxicology applications
9. more acs_boston slides online if anyone at acs listens we need central repository for this

Top 10 PWIs retrieved by CPIR
0. discusses lab notebook requirements and discusses the scrap of paper disallowed mistake acs_boston acsrdf2010 any literature on such
1. the life scientist room at cliqset
2. molecular networks is discussing all the benefits of open standards and open source acs_boston are they changing their own north too
3. jeremy talks about computers and wet lab chemistry acsrdf2010 acs_boston
4. biomed central is going to support open data
5. biomed central is going to support open data via
6. open science microformats initial thoughts jessy s acceptable
7. rich introduces the problem that science is too specialized for not using online tools just like i just did for cheminformatics acs_boston
8. linked open data and pavlova
9. jeremy discusses that chemistry is really slow with open scientific dissemination behind biology for example acsrdf2010 acs_boston

Figure 5.7: An example conversation and its retrieval results (documents are truncated)

are asked to authorize their accounts on the those social networks. Thereafter, our web application will work in the backend to pull the data that the users shared on those social networks. The user data are organized in ESRA's episodic and semantic memory, which are performed offline. Based on the information in the agent's memory, the server provide an interface to query the documents based on

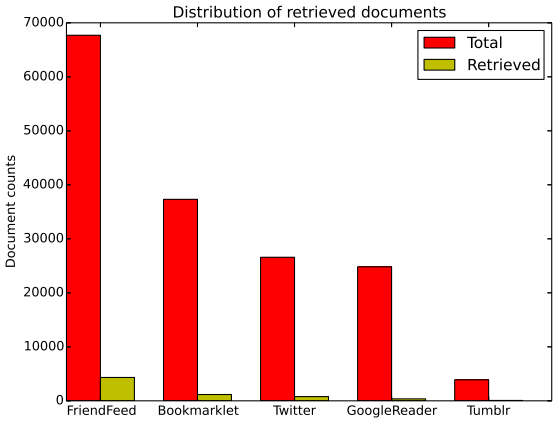


Figure 5.8: Distribution of retrieved documents in different SNSs

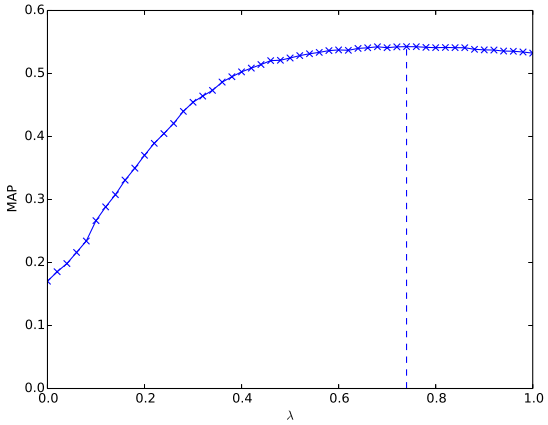


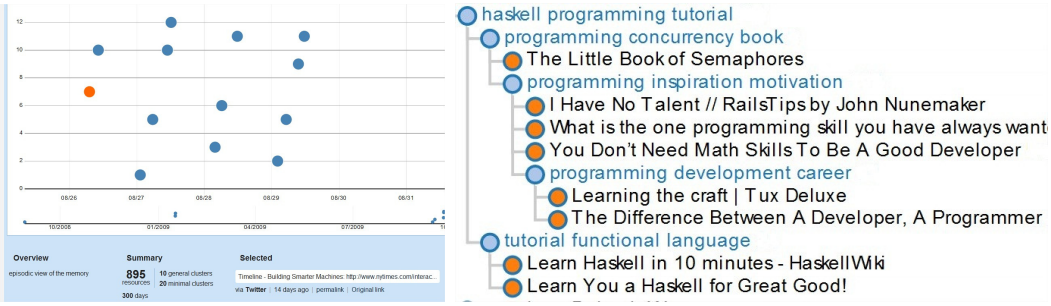
Figure 5.9: Effect of varying λ on MAP.

user context.

The sample user data stored in the agent's episodic memory and semantic memory are shown in 5.10. We also provide an interface for the user to explore the two different memories.

A browser extension[13], denoted as $WebESRAClient$, is developed to monitor the user information consumption behaviors in the web browser. Once the user installed the $WebESRAClient$n in his web browser, $WebESRAClient$ will act as a callback agent, which proactively monitor the user behaviors in the task environments. E.g., composing an email using Gmail, or reading news on NYTimes.

[13] `http://en.wikipedia.org/wiki/Browser_extension`



5.10.a: *EM* visualization. The points in the figure represent the resources collected by a user. The horizontal and vertical axis are the date and the time of the day at which the resources are collected. When a resource was selected, the detailed information about the resource is shown at the bottom

5.10.b: *SM* visualization. The texts in blue color are class labels of the categories and the texts in black color are documents classified in the categories

Figure 5.10: Visualization of information in *EM* and *SM*

Once such user behaviors are detected, The *WebESRAClient* will extract the context in the task environment and send the context to the server. The server analyzes the context and retrieves the most relevant documents from the agent's memory and returns the documents to the client for display. Figure 5.11 shows a task environment in which the user is composing an email.

We also collected the implicit and explicit feedback from the users with regards to relevancy and usefulness. For implicit feedback, we assume that if the user's click behavior - clicking on any episode in the top ranked list to expand for details - as the indicator that the episode is relevant to the context. For explicit feedback, we attach a five-star voting bottom to each top ranked episode and collect user's rating on each episode as the indicator of usefulness.

5.4 Summary

In this chapter, we described a web based contextual information retrieval agent *WebESRA* and illustrated how it can help retrieve the user's past contents and

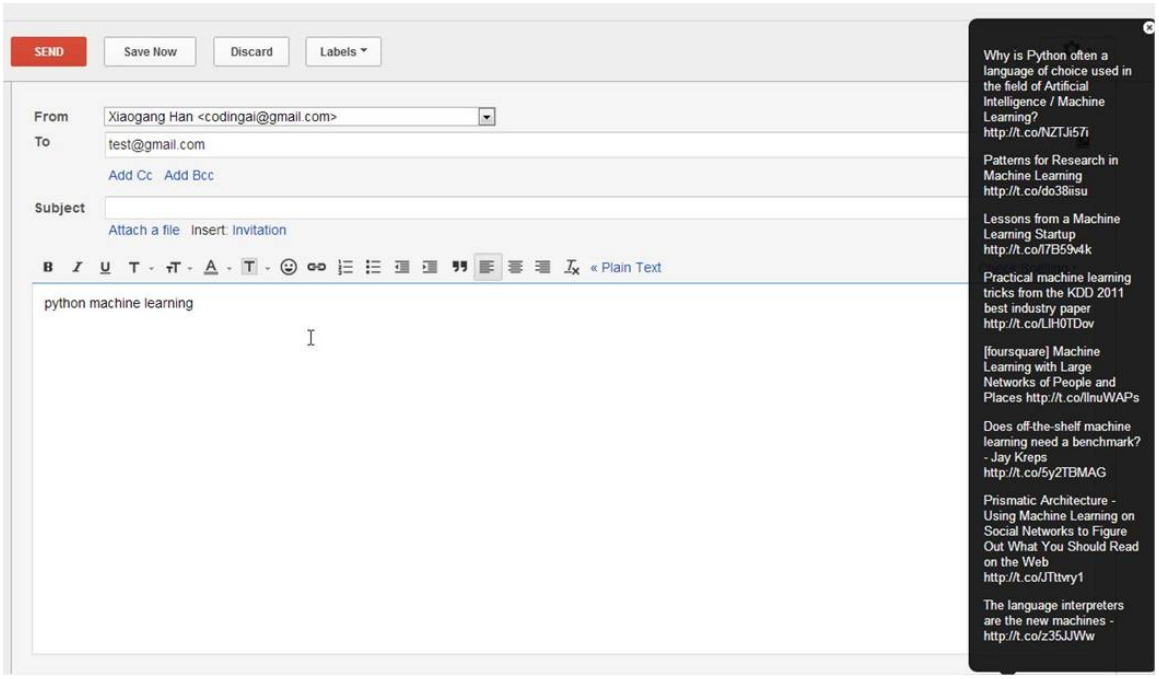


Figure 5.11: ESRA retrieving relevant user data when the user composes an email. When the user composes words, the agent will detect the latest composed text and pop up a window showing the most relevant topics and resources

make connections between those contents with the user's current task environment. A retrieval model incorporating multiple relevance measures was proposed to rank the most relevant historical episodes based on user context. We formulated the problem of automatic context-aware personal information retrieval to enhance user memory in the online conversation environment. By analyzing the FriendFeed data,, we found that the participating users and their PWIs possibly provided rich information for query expansion and implicit PWIs ranking. We employed such information to develop a two-step algorithm, namely CPIR, so as to solve the retrieval problem by considering both the participatory property and implicit-topical property of the context. In the first step, the query in the session is expanded with extra information from both the replies and the PWIs of the participating users. A customized smoothing method is developed to extract semantic information from the short texts. In the second step, a graph-based algorithm is employed to reveal the implicit relationship among the PWIs of all participating users and extract the user PWIs which match the topic of the session.

The experiments conducted on the real-world dataset collected from FriendFeed

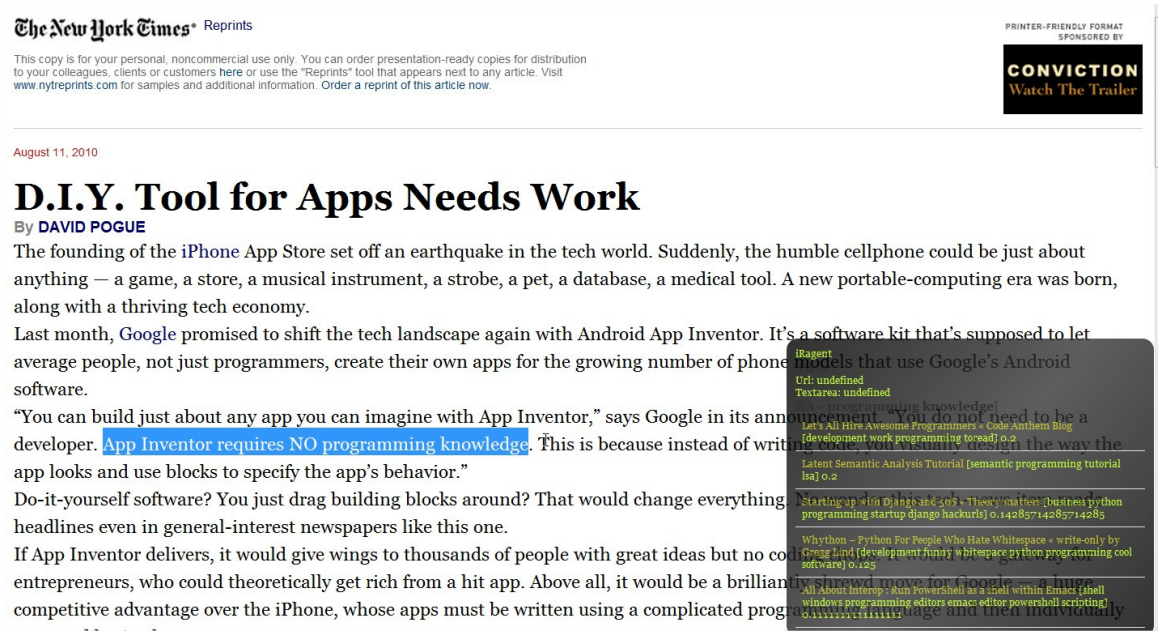


Figure 5.12: ESRA retrieving relevant user data when the user is reading NYTimes. When the user selects content, the agent will analyze the context and show relevant topics and resources

demonstrated that CPIR outperforms several baseline methods significantly. We also created a web application to evaluate *WebESRA*.

Chapter 6

Applications of ESRA

In this chapter, we investigate how ESRA and the knowledge obtained from individual users can be used to enhance the cognition of the users. In Chapter 5, we have demonstrated how the contextual retrieval framework of the ESRA work and how it can enhance the user's cognitive ability to satisfy his/her real-time information need in context. In this chapter, we study how the knowledge about the user obtained in the agent's knowledge base, especially in SM, enables the agent to provide personalized information services to the user .

More specifically, we apply the proposed ESRA in three real world applications:

- Personalized search based on user interests

  With the user profile with regards to general world knowledge constructed in SM, we build a personalized search framework to re-rank search results based on the user's preferences. We use the user profile obtained in Section 4.3 as the user's interests model and evaluated its efficiency using a personalized search system built on Yahoo search API. We will describe how the system work in Section 6.1.

- Recommending new tweets based on user interests exhibited in the user's Twitter timeline.

  Based on the short text classification algorithm developed in Section 4.4, we propose a tweets recommendation algorithm using the existing tweets posted by each user on Twitter. The recommendation algorithm is presented in Section 6.2.

- Providing contextual information help to students in virtual learning environments.

  We also have the rare opportunity to apply the ESRA to a 3D virtual environment, in which the agent observes the student's interactions with various

information objects and provides contextual help when the student is stuck at certain problems. The concrete problems the ESRA trying to solve in the virtual learning environment and how it is deployed in the environment is presented in Section 6.3.

6.1 Personalized Search Based on Semantic Memory

6.1.1 Personalized Search Based on Ontological User Profile

We now describe how to apply the user interest profile model developed in section 4.3 to personalize web search. The objective is to re-rank the web search results for a particular query q based on the similarity between q and each document d in a set of documents D. The degree of personalization is measured by calculating the similarity between u and each document d.

One of the problems with existing user interest profiling methods such as that of Noll and Meinel's [78] is that in their approaches the user interests are represented as a tag frequency vector, which cannot be decomposed to adapt to each individual context. In the personalized web search scenario, it is desired that the top ranked documents should be similar to both the user's interests and the search query. However, with Noll and Meinel's method, those documents which are more relevant to the user might be ranked high, the influence of the search query might be ignored. In our approach, we address the problem with a contextual user interest activation mechanism.

6.1.1.1 Contextual User Interests Activation

User interests are diverse and contextual. When matching between user interests and a context, only a small part of the user's interests are activated[165]. In the web search context, the most relevant and easy to get context is the user's query keywords. In our research, we regard the user's query as context and develop a contextual user interest activation algorithm.

In our ontological user interest profile, each interest is denoted by a leaf node in the ontological tree T. Suppose we identified N interests for a user, which is denoted as $\overrightarrow{In} = \{In_i \mid i = 0, ..., N\}$. Each In_i is a path from the root of T to a leaf category. A search context consists of a set of K keywords, which are also mapped to ODP and represented as concept vectors $\overrightarrow{Q} = \{q_j \mid j = 0, ..., K\}$.

For each $In_i \in \overrightarrow{In}$, the corresponding activation score in the context $\overrightarrow{Q}$ is calculated as follows:

$$activation\_score(\overrightarrow{In_i}, \overrightarrow{Q}) = \sum_{j=0}^{K} \delta(In_i, q_j) \quad \text{(Eq. 6.1)}$$

where δ is the similarity scheme developed by Ganesan et al. [133] to measure the relational similarity between two concepts in a hierarchy. We have:

$$\delta(l_1, l_2) = \frac{2 \times dep(LCA_U(l_1, l_2))}{dep(l_1) + dep(l_2)}, \quad \text{(Eq. 6.2)}$$

where l_1 and l_2 denote two concepts in a tree, $LCA_U(l_1, l_2)$ is the lowest common ancestor of l_1 and l_2, and $dep(l_1)$ and $dep(l_2)$ are the depth of these two concepts. After obtaining the activation scores for each In_i in the context Q, we sort the interests based on their scores in descending order. The top k interests are marked as activated.

6.1.1.2 User-Document Similarity

Noll and Meinel [78] developed a dimensionless measure to calculate the similarity between user interests and a document, which is denoted as: $\theta(u, d) = V_u \times |V_d|$. V_u denotes the user interest profile, in which each element represents the frequency of a tag assigned by user u, while V_d denotes the profile of document d, in which each element denotes the frequency of a tag labeled to document d by all users. All vectors are normalized using L2 normalization. Vallet [76] measured the similarity with the *term frequency-inverse document frequency (tf-idf)* weighting scheme, which is define as $similarity(u, d) = \sum_i tf\text{-}idf_{u_i} \times tf\text{-}idf_{d_i}$. Compared with traditional vector space model, their tf-idf scheme eliminates the user interests and document length normalization factors. This is because that while a document's popularity score is a good source of relevancy [5], it would be penalized by introducing the length normalization factor.

Our similarity measure differs with the two measures described above in that we take the context and the semantic interpretation into consideration to measure the relevance between the portion of user interests (activated in the context) and each document in the search result list. Our measure method is extended from *Generalized Cosine-Similarity Measure (GCSM)* proposed by Ganesan et al. [133], which is developed to measure the similarities among different concepts in hierarchies.

GCSM itself is a semantic extension of the *tf-idf* model. Suppose the set of activated user interests is denoted as *Active_In*, and a search result document is denoted as d, *GCSM* computes the similarity between $\overrightarrow{d}$ and $\overrightarrow{Activated\_In}$ as:

$$\theta(\overrightarrow{d}, \overrightarrow{Activated\_In}) = \sum_{i=1}^{n} \sum_{j=1}^{m} a_i b_j d_i \cdot Activated\_In_j, \quad \text{(Eq. 6.3)}$$

where a_i, b_j are the *tf-idf* for d_i and $Activated\_In_j$, respectively. Finally, $GCSM$ is normalized as:

$$\theta_n(\overrightarrow{d}, \overrightarrow{Activated\_In}) = \frac{\overrightarrow{d} \cdot \overrightarrow{Activated\_In}}{\sqrt{\overrightarrow{d} \cdot \overrightarrow{d}}\sqrt{\overrightarrow{Activated\_In} \cdot \overrightarrow{Activated\_In}}} \quad \text{(Eq. 6.4)}$$

For the same reason as described in Vallet [76], we also eliminate the length normalization factor. We denote the activation score for $Active\_In_j$ in query context Q as α_j. Finally, our similarity measure can be defined as follows:

$$\theta'(d, Active\_In) = \sum_{i=1}^{n}\sum_{j=1}^{m} \alpha_j a_i b_j d_i \cdot Active\_In_j, \quad \text{(Eq. 6.5)}$$

6.1.2 Experimental Evaluation

6.1.2.1 Personalized Web Search Evaluation

We now discuss how we evaluate the proposed user interest profiling method by utilizing it to provide personalized search results for individual user. A well defined evaluation framework for evaluating folksonomy based Web search personalization algorithms has been developed by David and Cantodor [76]. We adopt their framework to evaluate the method developed in our approach and compare our method with their approach with regards to personalization performance.

Within the evaluation framework, a user's social bookmarks are divided into two groups. The bookmarks in the first group is used as training dataset to create the user interest profile, while the bookmarks in the second group is the testing dataset used to test the user interest profile accuracy. For each bookmark in the testing dataset, the most popular tags by all users are extracted, which are used to represent the features of the corresponding document. A Web search is launched using the top tags as querying keywords and the search results are collected subsequently.

We assume that the bookmarked documents in a user's testing dataset fall in the user's topics of interests, otherwise the user would not have bookmarked them in the first place. Given a list of search result documents (using the popular tags of a document bookmarked by the user and it is supposed that the document is also in the search result list), the objective of an efficient user interest modeling algorithm is to rank the document with higher score compared with other result documents.

The evaluation for each document d in a user's testing dataset is performed in the following steps: 1) representing the features of d using the top k most popular tags assigned by all users; (2) querying the web search engine with the k tags as

keywords and collect the top R documents in the search results list; (3) if d is found in the R documents, applying the user interest profiling algorithm to each of the R documents; (4) calculating the original ranking position γ_1 and the new ranking position γ_2 of d and computing the Reciprocal Ranking of d. After performing these steps for all documents in a user's testing dataset, we calculate the Mean Reciprocal Ranking score, which is the average of reciprocal ranks for all documents in the testing dataset.

In our experiment, k is set to 3 and R is set to 500, which is the same settings as in [76]. In the case that document d does not appear in the top R result list, the document is ignored.

To measure the performance of proposed user interest profiling method, we need to re-rank each document in the search results list by combining its rank score assigned by the search system and its rank score assigned by personalization approaches. We choose to use CombSUM ranking aggregation algorithm [166] to combine the search engine ranking score and user interest profile based ranking score. Let τ denote the results list from the search engine for a query, $\tau(i)$ denote the position of document i in τ, $s^\tau(i)$ denote the ranking score assigned by the search engine to $i \in \tau$. The normalized scores [166] are as below, for an item $i \in \tau$:

- Rank normalization score:

$$\omega_R^\tau(i) = 1 - \frac{\tau(i) - 1}{|\tau|} \qquad \text{(Eq. 6.6)}$$

- Score normalization score:

$$\omega_S^\tau(i) = \frac{s^\tau(i) - min_{j \in t} s^\tau(j)}{max_{j \in t} s^\tau(j) - min_{j \in t} s^\tau(j)} \qquad \text{(Eq. 6.7)}$$

- The CombSUM score:

$$s_C^\tau(i) = \omega_S^\tau(i) + \omega_R^\tau(i) \qquad \text{(Eq. 6.8)}$$

For the document ranking by the search engine, as we can only get the ranking position $\tau(i)$, rather than the merged score, so rank normalization is used to normalize the score. For document ranking based on user interest profile, the ranking score $s^\tau(i)$ can be directly calculated. Therefore score normalization is applied.

6.1.2.2 Dataset Used in the Experiments

The dataset is extracted from 100 users' bookmarks on Delicious using its API[1], in which each user has at least 300 bookmarks. 80% of the bookmarks are used to create the user interest profile, and the other 20% are used for querying and evaluation as described above. In order to keep more valid documents in the testing dataset, before splitting the dataset, we query the search engine with the top popular tags of each document d in the bookmark collection for a user, and verify if d included in the top R results. Only documents which appeared in the results list are included in the testing dataset.

For the evaluation search engine, we use Yahoo Search Engine, which provide API through the Yahoo Boss search API [2]. Yahoo Search Boss API allows unlimited number of web queries and allows search results re-ordering, which makes it a great search engine to test personalization experiments. For the ODP ontology, we search on its site and extract the categories using HTTP requests.

6.1.2.3 Validity of Our Approach

Because the users choose the tags from their own vocabularies, it is not guaranteed that every tag has corresponding mapping on ODP. We define the percentage of mappable tags out of all the tags as *mapping rate*. Our analysis show that the *mapping rate* for 98% users are above 80%.

6.1.2.4 Comparing the Performance of Personalized Web Search Approaches

We now compare the performance of the personalization approaches. Table 6.1 shows the MRR scores for both the state-of-the-art approaches and our approach. *Noll* [78] and *Vallet* [76] in Table 6.1 are the two state-of-the-art methods. Our algorithm is denoted as *TagOnto*[3].

We can see from Table 6.1 that the MRR score of *Noll* is below the *Baseline*. It might be due to fact that the uniform term vector representation of user interests may give higher priorities to documents which are more similar to the user interests but not the current search context. The similar problem also happens to the *Vallet* method but not our method. Although the *tf-idf* weight scheme gives bias to each element in the user interests and the resource features, it is not directly based on the search context. Further, the semantic similarity information cannot be measured in their approach. The performance of the approach proposed in this work, *TagOnto*, is better than *Noll* and *Vallet* in terms of MRR.

[1] http://delicious.com/help/api

[2] http://developer.yahoo.com/search/boss/

[3] *TagOnto* — abbreviation for combination of tag and ontology based algorithm

	Baseline	Noll	Vallet	TagOnto
MRR	0.2547	0.2271	0.2876	0.3348

Table 6.1: Personalized search performance comparison

6.1.3 Discussion

In this section, we developed a contextual retrieval model for personalized search based on ontological user profiling. The ontological user profile is constructed by mapping the features (semantic tags) of users' information collection onto an ontology with a probability tree model. The experiment shows that personalized search that uses our user profile model outperforms a number of previous approaches (especially the one in [76] published in 2010). This improvement is mainly due to our introduction of semantic information inference via the mapping of folksonomy onto a domain ontology. Therefore, our model of diverse user interests is more accurate and semantically inferable.

The user profile generation algorithm described in this chapter is similar to the consolidation algorithm developed in Chapter 4, except that the one described in this chapter depend on an external ontology for the categorization. It is left for future work to investigate the comparison between these two approaches for user profile generation.

6.2 Recommending Short Texts on Microblogging Services

Using the graph based user interest model developed in Section 4.4, we develop a content-based tweet recommendation system. The preference of the user is derived from the user's historical tweets and represented with the proposed graph based representation method. The motivation is that the more similar between a tweet and a user's profile, the more likely that the user will be interested the tweet.

We define the similarity between user u and tweet t as the aggregated co-occurrence count for all the tweet concepts in user interests.

$$sim(u,t) = sim(In_u, t) = \sum_{0<i<N}^{N} freq(In_u, c_i) \qquad \text{(Eq. 6.9)}$$

6.2.1 User Interest Profile Based Recommendation Algorithm

The recommendation algorithm is shown in Algorithm 5. The predicted target user for a tweet t is the one whose interests has the highest similarity score with the concepts extracted from t.

input : T_{test} — Testing tweets;
U - targeting users
output: $u_{predict}$ — the predicted targeting user for a given tweet

```
foreach t ∈ T_test do
    foreach u ∈ U do
        calculate sim(u, t)
    end
    u_predict = arg max_u sim(u, t)
end
```

Algorithm 5: User Interests Based Recommendation

We studies the user interests based recommendation problem as the classification of randomly selected tweets into classes which are labeled by individual Twitter users. We split the user tweets into training set (80%) and testing set (20%), and the tweets in the training set are used to build the user interests model.

- Binary classification — In this experiment, we only use two users and their tweets for calculation. Given a tweet, the algorithm will predict which of the two users it belonging to.
- Multi-class classification — It is known that as the number of label increase, the prediction accuracy will reduce.

6.2.2 Experimental Evaluation

We randomly selected 14 users and collected their most recent 1000 tweets, of which 80% were used for building the user interests model and 20% were for evaluation. We tread the prediction problem as a classification problem and the users are regarded as target classes. The task is to classify new tweets in the 20% tweets space into one of the classes.

Table 6.2 shows the performance of the two classification algorithms *concept*, *graph* and a baseline algorithm, which use the vector representation, concept representation and graph representation of user interest model as described in section 4.4.2, respectively The first experiment defines a 2 classes classification problem. In each

batch, 2 users and their tweets are used for the evaluation. The final classification precision score is obtained by averaging on all the batches. The second experiment defines a 14 classes classification problem. In our experiment, both *concept* and *graph* based algorithms can obtain high precision as compared with the baseline *vector* representation. *graph* algorithm further outperforms *vector* due to the deeper semantic information retained. The results imply that the *graph* based representation and the corresponding classification algorithm work well to identify tweets for different users.

Algorithm	No. Class	Precision
vector	2	0.6525
	14	0.2466
concept	2	0.8097
	14	0.4722
graph	2	0.8365
	14	0.5210

Table 6.2: Classification algorithm performance

6.3 Helping Students Remembering Things in Virtual Learning Environments

6.3.1 Learning in Virtual Environments

The online virtual worlds were made available due to the advances in computer graphics and network technologies. Since then it has been widely applied to domains like education, tourism, e-commerce, and many other fields. Lots of research work [167][168] have demonstrated the great potential of leveraging virtual worlds as the next generation learning environments, because virtual worlds can offer a sense of immersion, which is important for engagement and learning, by enhancing the three aspects of education: allowing multiple perspectives, situated learning, and transfer. Numerous other studies in education research have also demonstrated the advantages of utilizing virtual worlds as learning environments, such as increasing motivation[169] [170] and enhancing the learning affordance [171] [172]. James Lee argued [173] that game, especially educational game, are incorporated with good learning principles. For example, good games give information on demand and just-in-time within the contexts of actual use or people's purposes and goals, which frequently happen in schools. People are quite poor at understanding and remembering information they have received out of context or for too long a period before they can make use of it.

Virtual learning environments are designed to enhance learning beyond classroom settings, guided by the learning-by-doing principle. Usually the students enter into the virtual world with some learning goals and predefined tasks. The environment provides the students with various learning scenarios in which the students can chat with non-player characters, do experiments, and take notes. On the other hand, the students are required to accomplish certain tasks.

One of the problems with learning in virtual worlds is that due to the complexity of the environments, the player is always required to learn some fundamental knowledge and later apply the acquired knowledge to accomplish some missions. There is a gap between the abstract knowledge and the immediate problem to be solved at hand. It is not uncommon that the player may fail to remember the previously acquired information, especially if the information is not visualized. The player will become frustrated when he fails to relate the problem to be solved at hand to existing information.

Many studies have shown the advantages of using intelligent software agents to achieve different educational goals in computer-based learning environment [174] [175]. Based on such observations, we have conducted preliminary research on incorporating ESRA in a 3D virtual learning environment to assist the game players organize the knowledge acquired and enhance their problem solving ability by helping them connecting the new situation to their prior knowledge.

During the past few years, we had developed a learning companion augmented virtual world- the Chronicles of Singapura (CoS), which is designed for lower secondary school students in Singapore to learn the transport systems in the plant. CoS has been deployed in two local secondary schools in Singapore and involved over 500 students to participate in the study. We implemented the ESRA as a remembrance companion and deployed it into CoS to enhance students' cognitive load [176] and help the learner to organize the knowledge acquired and making connections between new situations with previously acquired knowledge.

6.3.2 The Chronicles of Singapura (CoS)

The Chronicles of Singapura (CoS), which is designed based on 19th century Singapore, is a virtual world based learning environment designed for lower secondary school students to learn the science knowledge, specifically, the plant transport systems. In the virtual environment, the students control their avatars to explore, acquire knowledge, and solve simulated problems.

The primary objective the CoS is to provide an immersive environment for the students to learn science knowledge. In CoS, we model the world knowledge WK as an ontology, which consists of a set of concepts in the plant domain, and the

relationships between those concepts. Each node represents a key concept c, which denotes a virtual object in the virtual world. Each edge represents the relationships r between two concepts. Via interacting with the virtual objects and undertaking learning tasks, the users will build their own knowledge about the domain, which is denoted as UK. Partial of world knowledge in CoS is shown in Figure 6.1

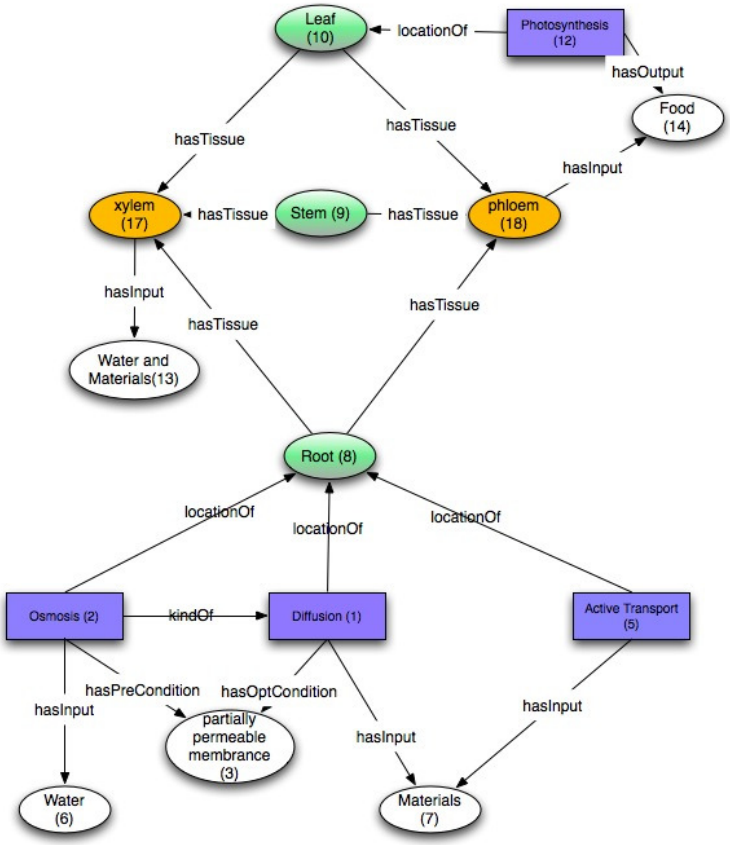


Figure 6.1: Partial knowledge base represented in CoS

6.3.3 Deploying ESRA in CoS

In CoS, ESRA continuously monitors the player's information consumption behaviors, and then organizes the information in its knowledge base. When the player is detected stuck at certain problems, relevant information will be retrieved and presented to the player. The player can also occasionally explore the temporal-spatial structured or topical clustered knowledge base.

The remembrance companion also perceives three sources of information: the predefined world knowledge WK, the user knowledge UK and the user context

UC. The agent proactively monitors the user's behavior in UC. When the user is detected being stuck, the agent will retrieve the most relevant concepts to help the user solve the problems.

As mentioned in the former sections, in the virtual environment, the world knowledge WK is represented as a set of concepts. For the remembrance companion, the user's learning behaviors are represented by episodes, denoted as E. Each episode, denoted as e, represents a learning event via which the user u interacted with a concepts $set(c)$ at time t and location $l_{x,y,z}$. Optionally, an episode may also includes the user's friends fs and the set of concepts cs_e. The player's real-time context, which is a special kind of episode, is denoted as e_c.

$$e = (u, t, l_{x,y,z}, set(c), fs)$$

The episodes are encoded and stored in the agent's episode memory EM.

6.3.3.1 Memory Consolidation

The episodes stored in EM can serve as evidences on the user's interaction with the learning concepts. Based on the world knowledge WK and the user behavior obtained in the EM, we derived an memory consolidation algorithm to estimate the user's knowledge with regards to the concepts defined in WK. We store the user's knowledge in the agent's semantic memory SM.

The consolidation algorithm regards the consolidating problem as a multi-label episode classification problem, in which each concept in WK is considered as a class label. The objective of the classification algorithm is to classify each episode in EM into one or more concepts in WK, given the features of the episode.

6.3.3.2 Retrieval Model

ESRA situated in the virtual environment will continuously watch the player's problem solving activities and pro-actively retrieve relevant information based on local context. The local context can be simply represented as the evidence (e.g., the player is stuck, answering questions, asking for help, or spending a long time in solving a problem) observed by the agent. The agent records the user's exploration history and finds the user's information need patterns in real-time.

When such patterns are found in certain context, e_c, the agent will extract relevant episodes in the EM or learning objects in the SM, and present them to the player in a non-intrusive manner. The player can also occasionally explore EM and SM to review the events he had experienced in the past and the knowledge he has obtained so far.

id	event_type	event_title	event_content	time	location	player	participants
1	Action	LeaveGame		2010-10-26 05:51:43			
2	Message	Received System	Welcome to a Torq...	2010-10-26 05:51:51		chrishan	
3	Action	EnterGame		2010-10-26 05:52:31	Singapore 249.31 11...	chrishan	
4	Action	ObjectClicked		2010-10-26 05:53:57	Singapore 271.966 9...	chrishan	Ahmad bin Sulaiman
5	Action	ObjectClicked		2010-10-26 05:54:45	Singapore 167.348 1...	chrishan	Lothaire
6	Message	Received NPC	Lothaire: Hi, I am Br...	2010-10-26 05:54:45	Singapore 167.348 1...	chrishan	Lothaire
7	Message	Received NPC	Lothaire: Find the d...	2010-10-26 05:54:48	Singapore 167.348 1...	chrishan	Lothaire
8	Gui	Map	Open Singapore	2010-10-26 05:54:48	Singapore 167.348 1...	chrishan	
9	Action	ObjectClicked		2010-10-26 05:56:56	Singapore 269.486 2...	chrishan	Miss Lee
10	Message	Received NPC	Miss Lee: I am Miss ...	2010-10-26 05:56:56	Singapore 269.486 2...	chrishan	Miss Lee
11	Action	ObjectClicked		2010-10-26 05:57:13	Lab	chrishan	TableStaticData
12	Gui	OsmosisA	Open	2010-10-26 05:57:13	Singapore 275.336 2...	chrishan	
13	Gui	OsmosisA	Close	2010-10-26 05:58:08	Singapore 273.802 2...	chrishan	
14	Action	ObjectClicked		2010-10-26 05:58:12	Lab	chrishan	TableStaticData
15	Gui	OsmosisExperiment	Open	2010-10-26 05:58:12	Singapore 273.953 2...	chrishan	
16	Gui	OsmosisExperiment	Close	2010-10-26 06:01:08	Singapore 273.953 2...	chrishan	
17	Message	Received System	The sky starts to get...	2010-10-26 06:05:35	Singapore 273.953 2...	chrishan	
18	Action	ObjectClicked		2010-10-26 06:10:53	Lab	chrishan	TableStaticData
19	Gui	OsmosisExperiment	Open	2010-10-26 06:10:53	Singapore 273.953 2...	chrishan	
20	Gui	OsmosisExperiment	Close	2010-10-26 06:10:59	Singapore 273.953 2...	chrishan	
21	Action	ObjectClicked		2010-10-26 06:11:03	Lab	chrishan	TableStaticData
22	Gui	OsmosisA	Open	2010-10-26 06:11:03	Singapore 273.953 2...	chrishan	
23	Gui	OsmosisA	Close	2010-10-26 06:11:04	Singapore 273.953 2...	chrishan	
24	Action	ObjectClicked		2010-10-26 06:11:06	Lab	chrishan	TableStaticData
25	Gui	OsmosisSimulation	Open	2010-10-26 06:11:07	Singapore 273.953 2...	chrishan	
26	Gui	OsmosisSimulation	Close	2010-10-26 06:11:09	Singapore 273.953 2...	chrishan	
27	Action	ObjectClicked		2010-10-26 06:12:02	Singapore -10.4903 ...	chrishan	Uncle Ben
28	Message	Received NPC	Uncle Ben: Welcom...	2010-10-26 06:12:02	Singapore -10.4903 ...	chrishan	Uncle Ben
29	Message	Received NPC	Uncle Ben: Look, th...	2010-10-26 06:12:04	Singapore -10.4903 ...	chrishan	Uncle Ben

Figure 6.2: User behaviors recorded by ESRA

Episodic Retrieval When the student is trying to solve a simulated problem, he may have difficulties recalling that he has interacted with the knowledge objects required to solve the problem. In such cases, rather than pointing out the solution, ESRA will trigger episodic retrieval to identify the most relevant episodes in the EM. The ranking of episode e, given user context e_c, can be measured by the similarity between e and e_c and calculated on the three relevance measures defined in chapter 3.

Semantic Retrieval There are cases when the student cannot solve a problem because he hasn't obtained the knowledge required yet. In such cases, ESRA will trigger the semantic retrieval to identify the most relevant concepts in the SM with regards to the user context. The ranking of concept c, given user context e_c is measured by the similarity between c and the concepts in e_c, which is defined in chapter 3. After retrieval, the top-k concepts with the highest ranking scores are selected. Their definitions and relations with each other are displayed to the user. Furthermore, the location at which the students can acquire those knowledge objects are also shown to the user.

A screenshot of ESRA in CoS is shown in Figure 6.3. We implemented the agent as a diary book, in which it records the student's activities in CoS, including the virtual objects he interacted with. In the figure, the left side shows the ontology of

Figure 6.3: The remembrance learning companion embodied by a diary book

the plant knowledge in the virtual world. The grey blocks are the not-interacted-with-yet virtual objects, whereas the colored blocks are virtual objects with which the student has interacted with. The right side shows the screenshots captured when the user interacts with certain concepts in the virtual world, together with all the relevant informations involved in the interaction. The diary book is minimized in the interface by default. When the student is detected having problems, the agents will determine whether there are any relevant information that can help and what information to provide to help. Its icon will flicker in the bottom toolbox and the student can decide whether to open the diary book.

6.4 Summary

In this chapter, we described three application scenarios for the proposed ESRA. It is observed that the ESRA model is very general and can be easily applied to different application settings.

Chapter 7

Conclusion and Future Work

In this chapter, we conclude the thesis by highlighting the main contributions and identify several research directions to work on in the future to improve and enhance the current model.

7.1 Summary of Contributions

Remembrance agents are agents with sophisticated models to solve the information overloading problem. This thesis started with a discussion on the limitations of existing RAs and proposed a novel ESRA to eliminate these limitations subsequently.

The main contributions of this research are:

- The ESRA inspired by cognitive memory theories

  The ESRA is proposed in Chapter 3, which extends the existing RA by explicitly modeling EM and SM and their interaction in the agent's memory. In this way, both episodic information and categorical semantic information of the user documents are retained (in EM and SM, respectively). The categorical information identified for each document reveals the hidden topics of the corresponding document, which, in turn, enrich the representation of the document with background ontological knowledge. This improves the accuracy of retrieving the user's documents.

- Memory consolidation methods for ontological user interests modeling

  In the proposed ESRA model, the enrichment mentioned above is achieved through classifying the documents and mapping the entities in the documents onto external Web ontologies. As it is assumed that the user documents might be collected from diverse information sources, we investigate three typical types of user documents on the Web and implemented three enrichment algorithms to consolidate them. The three algorithms are:

- k-NN based hierarchical text classification for normal document

  Although document classification has been studied extensively for decades, the size of the Web ontologies and the number of documents produced by the user on the Web have made large-scale hierarchical text classification a challenging problem with regards to accuracy and scalability. We incorporate the hierarchical relationships between different categories as extra features into the k-NN algorithm. The experiment results show that the proposed algorithm can achieve significantly higher accuracy compared with several baseline algorithms.

- Collective ontological classification for social bookmarks

  Social tagging is another popular information source, in which the users assign tags to their favorite documents. We study the problem of mapping the tags onto Web ontologies to determine the categories of the corresponding documents. We build a collective tag mapping algorithm by utilizing the neighboring tags assigned to the same document. Our approach takes advantage of both the folksonomy and Web ontology, which can accurately capture the semantics of the tags and solve the redundancy problems facing purely folksonomy-based approaches.

- Anchor text based on ontological mapping for short texts on microblgging services

  Social updates on microblogging services such as Twitter and Facebook provide another source of user documents. As the content of the social updates are naturally short, we study the problem of identifying the categories of social updates through the entity linking approach. Our approach determines the significance of n-grams both in the tweet corpus space and the ontology space by incorporating various features with regards to the two spaces. The evaluations on two datasets show that the accuracy of our *TW-Ranking* algorithm outperforms two state of the art algorithms.

As the ESRA maintains the user's historical documents in its memory, it has rich information on the preferences of the user. To model user interests in ESRA, we take advantage of the categorical information of the user documents to derive the user interest profile. We developed an ontological user interest profiling algorithm by aggregating the mapped categories for all the user documents and assigning them with weighted activation scores. The resulting user interest profile is represented as a set of tuples. The first element

denotes a category name in the Web ontology, while the second element is a score denoting the user's interest intensity on the category.

Compared with existing user interest profiling approaches, our method retains more semantic level information by taking advantage of the hierarchical relations between different categories in the ontology. The evaluation experiment shows that our method can accurately capture user interests and enable semantic reasoning in the ontology. Based on the ontological user profile, semantic inferences can be made to provide a wide range of personalized recommendations to the user.

- Contextual memory retrieval from the user's multiple information sources

  This thesis studies the information source diversity problem in real world. This is motivated by the observation that users' documents are usually distributed in multiple information sources (such as different social network sites). In order to unify all those documents from different information sources, we developed a retrieval algorithm which incorporates both episodic and semantic features into a linear ranking model. A conversation based evaluation framework was developed using the social conversations on a popular social network site — FriendFeed. The experimental results show that given a context modeled by a social conversation, the proposed algorithm can accurately identify the user's most relevant documents on multiple social network sites to reduce the user's cognitive load.

- Applying ESRA to real-world problems

  To study the usefulness of the proposed ESRA in practice, this thesis applies ESRA to three real-world applications.

  - Personalized search based on ontological user profile obtained from social tagging services

    Utilizing the ontological user interest profile obtained from the user's social tagging behaviors, we developed a personalized Web search algorithm. Compared with the *Noll* algorithm, our algorithm can activate the user interests partially based on the search context. The ontological representation of the user interest profile also makes it possible to determine document similarity with *GCSM* measures. The evaluation results obtained using the Yahoo Search API show that our *TagOnto* algorithm outperforms *Noll* and *Vallet* algorithm in terms of MRR.

- Recommending social updates on microblogging services

  To evaluate the user interest profile obtained from the entity linking algorithm using the user's social updates, we study the task of social updates recommendation on Twitter. Our algorithm leverages the mapped entities obtained from the user's tweets as the primary features to match a new tweet with the ontological user interest profile. The experiment results show that the ontological user profile approach can produce higher recommendation precision compared with two baseline approaches.

- Helping students remember things in VLEs

  In VLEs, the students are usually required to acquire knowledge and then apply it in a coherent manner to accomplish certain tasks. The problem is that the student may become frustrated when he/she failed to associate the current task with the past knowledge he/she has encountered in previous scenes. In order to solve this problem, we deployed ESRA in a learning companion augmented virtual world — *CoS*. The ESRA continuously monitors the student's behaviors and records the information objects experienced by the student. When the student is detected to be stuck at certain problem, the agent will retrieve relevant information objects to help the student solve his/her problems.

7.2 Future Work

ESRA provides an initial evaluation of a proactive information retrieval system which emphasizesemphasizing personal, social and proactive aspects of a memory support system. In this section, we describe possible extensions for ESRA.

1. Large scale general knowledge extraction from unstructured data General world knowledge is a fundamental building block for remembrance agent , based on which the agent can make inference on the user to derive user preferences. However, such knowledge is very difficult to collect. Some researchers [177][178][179] start to seek ways to extract structured information from unstructured web pages. Our future work on this topic will develop a novel approach for knowledge extraction. Specifically, we plan to extract general knowledge from open source codebase. The motivation of the idea is based on the observation of growth and accessibility of open source software development and the potential to mining the codebase at large scale. The main difference between the existing work on source code mining [180][181] and our approach is that the existing work mainly focused on closing the gap between the semantics of the code and the ontology standards, like OWL,

whereas our objective is to build general world knowledge base using statistical approaches. The to-be-built knowledge base can benefit the knowledge based inference for remembrance agent.

2. Extending ESRA to non-declarative memory The personal memory extender — Memex, was never implemented due to two limitations: the need for very, the large storage and easily accessible anywhere context. However, these two limitations are no longer the problem nowadays. Smart phones and tablets are replacing PC as the primary information processing devices for the users. The availability of physical location information on mobile devices enables us to build context-aware remembrance agent which acts as intelligent reminders to provide real-time recommendations for the user. We plan to develop a procedural memory (PM), in parallel with EM and SM, to store the user's actions and goals. The actions are collected from the observed user daily routines and stored in the form of time series. The goals are recognized by mining the user actions to find common patterns. We will extend the consolidation methods developed in Chapter 4 to design the goal recognition algorithm. Based on the actions and goals stored in PM, we will extend ESRA with the ability to recommend the most likely goals for the user after each observation. With the specific goal in mind, the ESRA will recommend the most suitable actions to achieve the goal. Incorporating with the knowledge with regards to the environment and the user stored in SM, ESRA can further help decide the best time to achieve selected goals so as to maximize the user's benefits.

3. Topical news recommendation based on user interests analysis One of the main inference abilities of ESRA is to derive user preferences model in its semantic memory. In this future research work, we will study how to incorporate the information (about the individual user) in episodic memory and semantic memory to derive a personalized recommendation engine. The recommendation engine takes a novel approach to identify evolutionary authority community users to rank top stories on each topic. Thereafter, the recommendation engine will further delivery the stories to individual users based on their long-term and short-term preferences.

Appendix A

Publication List

Journal Articles

(i) Xiaogang Han, Wei Wei, Chunyan Miao, and Jian-Ping Mei, "Context-Aware Personal Information Retrieval From Multiple Social Networks," *IEEE Computational Intelligence Magazine*, 2014 (to appear)

Conference Articles

(i) Qiong Wu, Xiaogang Han, Han Yu, Zhiqi Shen, and Chunyan Miao, "The innovative application of learning companions in virtual singapura," *Proc. of the 12th International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS13)*, pp.1171–1172, 2013.

(ii) Xiaogang Han, Zhiqi Shen, and Shaohua Li, "A k-NN Method for Large Scale Hierarchical Text Classification at LSHTC3," *ECML-PKDD Data Challenge Workshop*, 2012.

(iii) Xiaogang Han, Junfa Liu, Zhiqi Shen, and Chunyan Miao, "An Optimized K-Nearest Neighbor Algorithm for Large Scale Hierarchical Text Classification," *ECML-PKDD PASCAL Workshop*, pp. 2–12, 2011.

(iv) Xiaogang Han, Zhiqi Shen, Chunyan Miao, and Xudong Luo, "Folksonomy-Based Ontological User Interest Profile Modeling and Its Application in Personalized Search," *Proceedings of the 6th international conference on Active media technology*, pp. 34–46, 2010.

References

[1] P. Maes, "Agents that reduce work and information overload," *Communications of the ACM*, vol. 37, no. 7, pp. 30–40, 1994.

[2] B. Rhodes, *Just-in-time information retrieval*. PhD thesis, Massachusetts Institute of Technology, 2000.

[3] B. Rhodes, "Margin notes: Building a contextually aware associative memory," in *Proceedings of the 5th international conference on Intelligent user interfaces*, pp. 219–224, 2000.

[4] Y. Miyashita, "Cognitive memory: cellular and network machineries and their top-down control," *Science*, vol. 306, no. 5695, p. 435, 2004.

[5] A. Hotho, R. Jaschke, C. Schmitz, and G. Stumme, "Information retrieval in folksonomies: Search and ranking," *The Semantic Web: Research and Applications*, pp. 411–426, 2006.

[6] V. Bush and A. Think, "As we may think," *The Atlantic Monthly*, vol. 176, no. 1, pp. 101–108, 1945.

[7] V. Bush, "As we may think," *Reading digital culture*, p. 9, 2001.

[8] V. Bush, "Memex revisited," in *From Memex to hypertext*, ch. Memex revisited, pp. 197–216, Academic Press Professional, Inc., 1991.

[9] M. Lamming and M. Flynn, "Forget-me-not: Intimate computing in support of human memory," in *Proceedings of International Symposium on Next Generation Human Interface*, vol. 94, pp. 2–4, 1994.

[10] H. Byun and K. Cheverst, "Exploiting user models and context-awareness to support personal daily activities," in *Workshop in UM2001 on User Modelling for Context-Aware Applications*, 2001.

[11] B. Rhodes and T. Starner, "Remembrance Agent: A continuously running automated information retrieval system," in *Proceedings of the First International Conference on the Practical Application of Intelligent Agents and Multi Agent Technology*, pp. 487–495, 1996.

[12] B. Rhodes, "The wearable remembrance agent: A system for augmented memory," *Personal and Ubiquitous Computing*, vol. 1, no. 4, pp. 218–224, 1997.

[13] B. Rhodes, "Using physical context for just-in-time information retrieval," *IEEE Transactions on Computers*, vol. 52, no. 8, pp. 1011–1014, 2003.

[14] D. Leake, R. Scherle, J. Budzik, and K. Hammond, "Selecting task-relevant sources for just-in-time retrieval," in *Proceedings of the AAAI-99 Workshop on Intelligent Information Systems*, 1999.

[15] J. Budzik and K. Hammond, "User interactions with everyday applications as context for just-in-time information access," in *Proceedings of the 5th international conference on Intelligent user interfaces*, pp. 44–51, 2000.

[16] H. Lieberman, "Autonomous interface agents," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 67–74, 1997.

[17] C. Xia and P. Maes, "The design of artifacts for augmenting intellect," in *Proceedings of the 4th Augmented Human International Conference*, pp. 154–161, 2013.

[18] E. Ofek, S. T. Iqbal, and K. Strauss, "Reducing disruption from subtle information delivery during a conversation: mode and bandwidth investigation," in *Proceedings of the 2013 ACM annual conference on Human factors in computing systems*, pp. 3111–3120, 2013.

[19] S. Dumais, E. Cutrell, R. Sarin, and E. Horvitz, "Implicit queries (iq) for contextualized search," in *Proceedings of the 27th annual international ACM conference on Research and development in information retrieval*, pp. 594–594, 2004.

[20] T. Ayodele and S. Zhou, "Applying machine learning techniques for e-mail management: solution with intelligent e-mail reply prediction," *Journal of Engineering and Technology Research*, vol. 1, no. 7, pp. 143–151, 2009.

[21] P. N. Garner, M. Yazdani, A. Nanchen, and A. Popescu-Belis, "A speech-based just-in-time retrieval system using semantic search," in *Proceedings of the ACL-HLT 2011 System Demonstrations (49th Annual Meeting of the Association for Computational Linguistics)*, 2011.

[22] S. Hunter, P. Maes, S. Scott, and H. Kaufman, "Memtable: an integrated system for capture and recall of shared histories in group workspaces," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 3305–3314, 2011.

[23] Y. Liu, D. Edge, and K. Yatani, "Sidepoint: a peripheral knowledge panel for presentation slide authoring," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 681–684, 2013.

[24] B. Hecht, J. Teevan, M. Morris, and D. Liebling, "Searchbuddies: Bringing search engines into the conversation," in *International AAAI Conference on Weblogs and Social Media*, 2012.

[25] J. Brandt, M. Dontcheva, M. Weskamp, and S. R. Klemmer, "Example-centric programming: integrating web search into the development environment," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 513–522, 2010.

[26] J. Matejka, T. Grossman, and G. Fitzmaurice, "Ambient help," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 2751–2760, 2011.

[27] J. Guerreiro, T. Guerreiro, and D. Gonçalves, "Surpassing farley files: opportunities and challenges on obtaining personally relevant information," in *Proceedings of the 12th international conference on Human computer interaction with mobile devices and services*, pp. 385–386, 2010.

[28] L. Jalali and R. Jain, "Building health persona from personal data streams," in *Proceedings of the 1st ACM international workshop on Personal data meets distributed multimedia*, pp. 19–26, 2013.

[29] H. Zeng, Q. He, Z. Chen, W. Ma, and J. Ma, "Learning to cluster web search results," in *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, p. 217, 2004.

[30] D. Ramage, P. Heymann, C. Manning, and H. Garcia-Molina, "Clustering the tagged web," in *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, pp. 54–63, 2009.

[31] K. Myers and N. Yorke-Smith, "Proactivity in an Intentionally Helpful Personal Assistive Agent," in *Proceedings of the AAAI Spring Symposium on Intentions in Intelligent Systems*, 2007.

[32] M. Pollack, L. Brown, D. Colbry, C. McCarthy, C. Orosz, B. Peintner, S. Ramakrishnan, and I. Tsamardinos, "Autominder: An intelligent cognitive orthotic system for people with memory impairment," *Robotics and Autonomous Systems*, vol. 44, no. 3-4, pp. 273–282, 2003.

[33] J. Gemmell, G. Bell, and R. Lueder, "MyLifeBits: a personal database for everything," *Communications of the ACM*, vol. 49, no. 1, p. 95, 2006.

[34] J. Anderson, *Language, memory, and thought*. Lawrence Erlbaum, 1976.

[35] E. Tulving, *Elements of episodic memory*. Clarendon Oxford, 1983.

[36] E. Tulving, *Episodic and semantic memory*, ch. 10, pp. 381–403. New York: Academic Press, 1972.

[37] D. Kolb *et al.*, *Experiential learning*. Prentice-Hall Englewood Cliffs, NJ, 1984.

[38] M. Cravo and J. Martins, "SNePSwD: A newcomer to the SNePS family," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 5, no. 2, pp. 135–148, 1993.

[39] G. Sartori and L. Lombardi, "Semantic relevance and semantic disorders," *Journal of Cognitive Neuroscience*, vol. 16, no. 3, pp. 439–452, 2004.

[40] L. Barsalou, "The content and organization of autobiographical memories," *Remembering reconsidered: Ecological and traditional approaches to the study of memory*, pp. 193–243, 1988.

[41] M. Conway and C. Pleydell-Pearce, "The construction of autobiographical memories in the self-memory system," *Psychological Review*, vol. 107, no. 2, pp. 261–288, 2000.

[42] K. Saywitz, R. Geiselman, and G. Bornstein, "Effects of cognitive interviewing and practice on children's recall performance," *Journal of Applied Psychology*, vol. 77, no. 5, pp. 744–756, 1992.

[43] D. Rubin and A. Wenzel, "One hundred years of forgetting: A quantitative description of retention.," *Psychological Review*, vol. 103, no. 4, p. 734, 1996.

[44] J. Wixted, "The psychology and neuroscience of forgetting," *Annual Review of Psychology*, vol. 55, pp. 235–269, 2004.

[45] D. Vernon, G. Metta, and G. Sandini, "A survey of artificial cognitive systems: Implications for the autonomous development of mental capabilities in computational agents," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 2, pp. 151–180, 2007.

[46] P. Langley, J. Laird, and S. Rogers, "Cognitive architectures: Research issues and challenges," *Cognitive Systems Research*, vol. 10, no. 2, pp. 141–160, 2009.

[47] A. Newell, *Unified theories of cognition*, vol. 187. Harvard University Press, 1994.

[48] J. Laird, A. Newell, and P. Rosenbloom, "Soar: An architecture for general intelligence," *Artificial intelligence*, vol. 33, no. 1, pp. 1–64, 1987.

[49] P. Rosenbloom, J. Laird, and A. Newell, *The Soar papers: Research on integrated intelligence, Vols. 1 & 2.* The MIT Press, 1993.

[50] J. Laird and P. Rosenbloom, "The evolution of the soar cognitive architecture," *Mind matters: A tribute to Allen Newell*, pp. 1–50, 1996.

[51] A. Nuxoll, *Enhancing intelligent agents with episodic memory.* PhD thesis, The University of Michigan, 2007.

[52] A. Nuxoll and J. Laird, "Extending cognitive architecture with episodic memory," in *Proceedings of the National Conference on Artificial Intelligence*, vol. 22, pp. 1560–1565, 2007.

[53] Y. Wang and J. Laird, "Integrating semantic memory into a cognitive architecture," tech. rep., University of Michigan, 2006.

[54] W. Duch, R. Oentaryo, and M. Pasquier, "Cognitive architectures: Where do we go from here?," *Frontiers in artificial intelligence and applications*, vol. 171, p. 122, 2008.

[55] K. Panton, C. Matuszek, D. Lenat, D. Schneider, M. Witbrock, N. Siegel, and B. Shepard, "Common sense reasoning–from cyc to intelligent assistant," *Ambient Intelligence in Everyday Life*, pp. 1–31, 2006.

[56] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann, "Dbpedia - a crystallization point for the web of data," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 7, no. 3, pp. 154–165, 2009.

[57] F. Suchanek, G. Kasneci, and G. Weikum, "Yago: A large ontology from wikipedia and wordnet," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 6, no. 3, pp. 203–217, 2008.

[58] G. Stillman, "Impact of prior knowledge of task context on approaches to applications tasks," *The Journal of Mathematical Behavior*, vol. 19, no. 3, pp. 333–361, 2000.

[59] J. Bransford, *How people learn: Brain, mind, experience, and school*. National Academies Press, 2000.

[60] Y. Ozuru, K. Dempsey, and D. McNamara, "Prior knowledge, reading skill, and text cohesion in the comprehension of science texts," *Learning and Instruction*, vol. 19, no. 3, pp. 228–242, 2009.

[61] L. Squire, C. Stark, and R. Clark, "The medial temporal lobe," *Annual Reviews*, pp. 279–306, 2004.

[62] A. Greve, M. C. Van Rossum, and D. I. Donaldson, "Investigating the functional interaction between semantic and episodic memory: Convergent behavioral and electrophysiological evidence for the role of familiarity," *Neuroimage*, vol. 34, no. 2, pp. 801–814, 2007.

[63] G. Salton, A. Wong, and C. Yang, "A vector space model for automatic indexing," *Communications of the ACM*, vol. 18, no. 11, pp. 613–620, 1975.

[64] P.-N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*. Addison-Wesley, 2005.

[65] Y. Yang and X. Liu, "A re-examination of text categorization methods," in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 42–49, 1999.

[66] A. Singhal, C. Buckley, and M. Mitra, "Pivoted document length normalization," in *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 21–29, 1996.

[67] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Random k-labelsets for multilabel classification," *IEEE Transactions on Knowledge and Dat Engineering*, 2010.

[68] C. Brouard, "Echo at the LSHTC Pascal Challenge 2," in *Joint ECML-PKDD PASCAL Workshop on Large-Scale Hierarchical Classification*, pp. 49–57, 2011.

[69] D.-H. Lee, "Multi-stage Rocchio Classification for Large-scale Multi-labeled Text data," in *ECML-PKDD 2012 PASCAL Workshop on Large-Scale Hierarchical Classification*, 2012.

[70] D. Kelly and X. Fu, "Eliciting better information need descriptions from users of information search systems," *Information Processing & Management*, vol. 43, no. 1, pp. 30–46, 2007.

[71] X. Shen, B. Tan, and C. Zhai, "Implicit user modeling for personalized search," in *Proceedings of the 14th ACM international conference on Information and knowledge management*, pp. 824–831, 2005.

[72] J. Teevan, S. Dumais, and E. Horvitz, "Personalizing search via automated analysis of interests and activities," in *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 449–456, 2005.

[73] T. Joachims, L. Granka, B. Pan, H. Hembrooke, F. Radlinski, and G. Gay, "Evaluating the accuracy of implicit feedback from clicks and query reformulations in web search," *ACM Transactions on Information systems*, vol. 25, no. 2, 2007.

[74] C. Yeung, N. Gibbins, and N. Shadbolt, "A study of user profile generation from folksonomies," in *Proceedings of the Workshop on Social Web and Knowledge Management*, 2008.

[75] M. Szomszor, H. Alani, I. Cantador, K. O'Hara, and N. Shadbolt, "Semantic modelling of user interests based on cross-folksonomy analysis," in *7th International Semantic Web Conference*, pp. 632–648, 2008.

[76] D. C. I. J. M. Vallet, "Personalizing web search with folksonomy-based user and document profiles," in *The Annual European Conference on Information Retrieval 2010*, 2010.

[77] S. Xu, S. Bao, B. Fei, Z. Su, and Y. Yu, "Exploring folksonomy for personalized search," in *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 155–162, 2008.

[78] M. Noll and C. Meinel, "Web search personalization via social bookmarking and tagging," in *The Semantic Web*, vol. 4825, pp. 367–380, 2007.

[79] P. Chirita, S. Costache, W. Nejdl, and S. Handschuh, "P-tag: Large scale automatic generation of personalized annotation tags for the web," in *Proceedings of the 16th International Conference on World Wide Web*, pp. 845–854, 2007.

[80] S. Golder and B. A. Huberman, "The structure of collaborative tagging systems," *Journal of Information Science*, vol. 32, no. 2, pp. 198–208, 2005.

[81] A. Shepitsen, J. Gemmell, B. Mobasher, and R. Burke, "Personalized recommendation in social tagging systems using hierarchical clustering," in *Proceedings of the 2008 ACM conference on Recommender Systems*, pp. 259–266, 2008.

[82] X. Han, Z. Shen, C. Miao, and X. Luo, "Folksonomy-based ontological user interest profile modeling and its application in personalized search," in *Proceedings of the 6th international conference on Active media technology*, pp. 34–46, 2010.

[83] P. Mika, "Ontologies are us: A unified model of social networks and semantics," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 5, no. 1, pp. 5–15, 2007.

[84] M. Michelson and S. Macskassy, "Discovering users' topics of interest on twitter: a first look," in *Proceedings of the fourth workshop on Analytics for noisy unstructured text data*, pp. 73–80, 2010.

[85] E. Meij, W. Weerkamp, and M. de Rijke, "Adding semantics to microblog posts," in *Proceedings of the fifth ACM international conference on Web search and data mining*, pp. 563–572, 2012.

[86] R. Mihalcea and A. Csomai, "Wikify!: linking documents to encyclopedic knowledge," in *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pp. 233–242, 2007.

[87] E. Gabrilovich and S. Markovitch, "Overcoming the brittleness bottleneck using wikipedia: Enhancing text categorization with encyclopedic knowledge," in *Proceedings of the National Conference on Artificial Intelligence*, vol. 21, p. 1301, 2006.

[88] J. Hu, L. Fang, Y. Cao, H. Zeng, H. Li, Q. Yang, and Z. Chen, "Enhancing text clustering by leveraging wikipedia semantics," in *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 179–186, 2008.

[89] C. Manning, P. Raghavan, and H. Schutze, *Introduction to information retrieval*, vol. 1. Cambridge University Press Cambridge, 2008.

[90] O. Phelan, K. McCarthy, and B. Smyth, "Using twitter to recommend real-time topical news," in *Proceedings of the third ACM conference on Recommender systems*, pp. 385–388, 2009.

[91] K. Canini, B. Suh, and P. Pirolli, "Finding credible information sources in social networks based on content and social structure," in *SocialCom-11*, 2011.

[92] J. Chen, R. Nairn, L. Nelson, M. Bernstein, and E. Chi, "Short and tweet: experiments on recommending content from information streams," in *Proceedings of the 28th international conference on Human factors in computing systems*, pp. 1185–1194, 2010.

[93] A. Java, X. Song, T. Finin, and B. Tseng, "Why we twitter: understanding microblogging usage and communities," in *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*, pp. 56–65, 2007.

[94] B. Jansen, M. Zhang, K. Sobel, and A. Chowdury, "Twitter power: Tweets as electronic word of mouth," *Journal of the American society for information science and technology*, vol. 60, no. 11, pp. 2169–2188, 2009.

[95] H. Kwak, C. Lee, H. Park, and S. Moon, "What is twitter, a social network or a news media?," in *Proceedings of the 19th international conference on World wide web*, pp. 591–600, 2010.

[96] H. Lieberman *et al.*, "Letizia: An agent that assists web browsing," in *International Joint Conference on Artificial Intelligence*, vol. 14, pp. 924–929, 1995.

[97] S. Middleton, N. Shadbolt, and D. De Roure, "Ontological user profiling in recommender systems," *ACM Transactions on Information Systems (TOIS)*, vol. 22, no. 1, pp. 54–88, 2004.

[98] Z. Ma, G. Pant, and O. Sheng, "Interest-based personalized search," *ACM Transactions on Information Systems (TOIS)*, vol. 25, no. 1, p. 5, 2007.

[99] M. Bernstein, B. Suh, L. Hong, J. Chen, S. Kairam, and E. Chi, "Eddi: interactive topic-based browsing of social status streams," in *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, pp. 303–312, 2010.

[100] J. Weng, E. Lim, J. Jiang, and Q. He, "Twitterrank: finding topic-sensitive influential twitterers," in *Proceedings of the third ACM international conference on Web search and data mining*, pp. 261–270, 2010.

[101] D. Ramage, S. Dumais, and D. Liebling, "Characterizing microblogs with topic models," in *International AAAI Conference on Weblogs and Social Media*, 2010.

[102] F. Abel, Q. Gao, G. Houben, and K. Tao, "Analyzing user modeling on twitter for personalized news recommendations," in *User Modeling, Adaption and Personalization*, 2011.

[103] C. Snoek, B. Huurnink, L. Hollink, M. De Rijke, G. Schreiber, and M. Worring, "Adding semantics to detectors for video retrieval," *IEEE Transactions on Multimedia*, vol. 9, no. 5, pp. 975–986, 2007.

[104] D. Milne and I. Witten, "Learning to link with wikipedia," in *Proceedings of the 17th ACM conference on Information and knowledge management*, pp. 509–518, 2008.

[105] X. Han, J. Liu, Z. Shen, and C. Miao, "An optimized k-nearest neighbor algorithm for large scale hierarchical text classification," in *ECML/PKDD PASCAL Workshop*, 2011.

[106] X. Hu, N. Sun, C. Zhang, and T. Chua, "Exploiting internal and external semantics for the clustering of short texts using world knowledge," in *Proceeding of the 18th ACM conference on Information and knowledge management*, pp. 919–928, 2009.

[107] S. Dill, N. Eiron, D. Gibson, D. Gruhl, R. Guha, A. Jhingran, T. Kanungo, S. Rajagopalan, A. Tomkins, J. Tomlin, *et al.*, "Semtag and seeker: Bootstrapping the semantic web via automated semantic annotation," in *Proceedings of the 12th international conference on World Wide Web*, pp. 178–186, 2003.

[108] P. Mendes, A. Passant, P. Kapanipathi, and A. Sheth, "Linked open social signals," in *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*, vol. 1, pp. 224–231, 2010.

[109] P. Ferragina and U. Scaiella, "Tagme: on-the-fly annotation of short text fragments (by wikipedia entities)," in *Proceedings of the 19th ACM international conference on Information and knowledge management*, pp. 1625–1628, 2010.

[110] G. Salton, *The SMART Retrieval System: Experiments in Automatic Document Processing*. Prentice-Hall, Inc., 1971.

[111] S. Robertson and K. Jones, "Relevance weighting of search terms," *Journal of the American Society for Information science*, vol. 27, no. 3, pp. 129–146, 2007.

[112] V. Vieira, P. Tedesco, A. Salgado, and P. Brézillon, "Investigating the specifics of contextual elements management: the cemantika approach," *Modeling and Using Context*, pp. 493–506, 2007.

[113] L. Tamine-Lechani, M. Boughanem, and M. Daoud, "Evaluation of contextual information retrieval effectiveness: overview of issues and research," *Knowledge and Information Systems*, vol. 24, no. 1, pp. 1–34, 2010.

[114] R. White, P. Bailey, and L. Chen, "Predicting user interests from contextual information," in *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pp. 363–370, 2009.

[115] W. Weerkamp, K. Balog, and M. De Rijke, "Using contextual information to improve search in email archives," *Advances in Information Retrieval*, pp. 400–411, 2009.

[116] D. Robins, "Interactive information retrieval: context and basic notions," *Informing Science*, vol. 3, no. 2, pp. 57–62, 2000.

[117] I. Ruthven, "Interactive information retrieval," *Annual review of information science and technology*, vol. 42, no. 1, pp. 43–91, 2009.

[118] D. Karger, K. Bakshi, D. Huynh, D. Quan, and V. Sinha, "Haystack: A customizable general-purpose information management tool for end users of semistructured data," in *Proceedings of of CIDR*, 2005.

[119] P. Chirita, S. Costache, W. Nejdl, and R. Paiu, "Beagle++: Semantically enhanced searching and ranking on the desktop," *The Semantic Web: Research and Applications*, pp. 348–362, 2006.

[120] S. Tyler, J. Wang, and Y. Zhang, "Utilizing re-finding for personalized information retrieval," in *Proceedings of the 19th ACM international conference on Information and knowledge management*, pp. 1469–1472, 2010.

[121] T. Deng, L. Zhao, H. Wang, Q. Liu, and L. Feng, "Refinder: A context-based information re-finding system," *IEEE Transactions on Knowledge and Data Engineering*, 2012.

[122] T. Deng, L. Zhao, L. Feng, and W. Xue, "Information re-finding by context: a brain memory inspired approach," in *Proceedings of the 20th ACM international conference on Information and knowledge management*, pp. 1553–1558, 2011.

[123] T. Deng and L. Feng, "A survey on information re-finding techniques," *International Journal of Web Information Systems*, vol. 7, no. 4, pp. 313–332, 2011.

[124] B. Sawyer, F. Quek, W. Wong, M. Motani, S. Chu Yew Yee, and M. Perez-Quinones, "Using physical-social interactions to support information re-finding," in *Proceedings of the 2012 ACM annual conference extended abstracts on Human Factors in Computing Systems Extended Abstracts*, pp. 885–910, 2012.

[125] X. Yu, M. Alkandari, P. Liu, and M. Pérez-Quiñones, "Visualizing a personal social network of email archives for re-finding," in *Proceedings of the 2nd Invitational Workshop on Personal Information Management at SIGIR*, vol. 2006, 2006.

[126] K. Lewis, J. Kaufman, M. Gonzalez, A. Wimmer, and N. Christakis, "Tastes, ties, and time: A new social network dataset using facebook.com," *Social Networks*, vol. 30, no. 4, pp. 330–342, 2008.

[127] F. Abel, S. Araújo, Q. Gao, and G. Houben, "Analyzing cross-system user modeling on the social web," *Web Engineering*, pp. 28–43, 2011.

[128] M. Szomszor, H. Alani, I. Cantador, K. O'Hara, and N. Shadbolt, "Semantic modelling of user interests based on cross-folksonomy analysis," in *Proceedings of the 7th International Conference on the Semantic Web*, pp. 632–648, 2008.

[129] F. Abel, E. Herder, G. Houben, N. Henze, and D. Krause, "Cross-system user modeling and personalization on the social web," *User Modeling and User-Adapted Interaction (UMUAI), Special Issue on Personalization in Social Web Systems*, vol. 22, no. 3, pp. 1–42, 2011.

[130] F. Carmagnola, F. Osborne, I. Torre, and B. White, "Cross-systems identification of users in the social web," in *Proc. of the IADIS International Conference*, pp. 129–134, 2009.

[131] Q. Wang and H. Jin, "Exploring online social activities for adaptive search personalization," in *Proceedings of the 19th ACM international conference on Information and knowledge management*, pp. 999–1008, 2010.

[132] G. Adomavicius and A. Tuzhilin, "Context-aware recommender systems," *Recommender Systems Handbook*, pp. 217–253, 2011.

[133] P. Ganesan, H. Garcia-Molina, and J. Widom, "Exploiting hierarchical domain structure to compute similarity," *ACM Transactions on Information Systems*, vol. 21, no. 1, pp. 64–93, 2003.

[134] S. Dumais, E. Cutrell, J. Cadiz, G. Jancke, R. Sarin, and D. C. Robbins, "Stuff i've seen: a system for personal information retrieval and re-use," in *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pp. 72–79, 2003.

[135] E. Cutrell, D. Robbins, S. Dumais, and R. Sarin, "Fast, flexible filtering with phlat," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 261–270, 2006.

[136] J. Teevan, E. Adar, R. Jones, and M. A. S. Potts, "Information re-retrieval: repeat queries in yahoo's logs," in *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 151–158, 2007.

[137] C. Marlow, M. Naaman, D. Boyd, and M. Davis, "Ht06, tagging paper, taxonomy, flickr, academic article, to read," in *Proceedings of the seventeenth conference on Hypertext and hypermedia*, pp. 31–40, 2006.

[138] R. Wetzker, C. Zimmermann, and C. Bauckhage, "Analyzing social bookmarking systems: A del.icio.us cookbook," in *Proceedings of the ECAI 2008 Mining Social Data Workshop*, pp. 26–30, 2008.

[139] H. Bruce, W. Jones, and S. Dumais, "Keeping and re-finding information on the web: What do people do and what do they need?," *Proceedings of the American Society for Information Science and Technology*, vol. 41, no. 1, pp. 129–137, 2004.

[140] A. Sellen, A. Fogg, M. Aitken, S. Hodges, C. Rother, and K. Wood, "Do life-logging technologies support memory for the past?: an experimental study using sensecam," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 81–90, 2007.

[141] E. L. S. Araujo, Q. Gao and G. Houben, "Linking personal data: towards the web of your digital memories," in *Proceedings of the WebSci10: Extending the Frontiers of Society On-Line*, 2010.

[142] M. Tungare, P. Pyla, M. Sampat, and M. Perez-Quinones, "Defragmenting information using the syncables framework," in *Proceedings of the 2nd Invitational Workshop on Personal Information Management at SIGIR*, 2006.

[143] V. Robu, H. Halpin, and H. Shepherd, "Emergence of consensus and shared vocabularies in collaborative tagging systems," *ACM Transactions on the Web*, vol. 3, no. 4, pp. 1–34, 2009.

[144] D. Schall and F. Skopik, "An analysis of the structure and dynamics of large-scale q/a communities," in *Advances in Databases and Information Systems*, pp. 285–301, 2011.

[145] M. Szomszor, H. Alani, I. Cantador, K. O'Hara, and N. Shadbolt, "Semantic modelling of user interests based on cross-folksonomy analysis," in *Proceedings of the 7th International Conference on The Semantic Web*, pp. 632–648, 2008.

[146] T. Iofciu, P. Fankhauser, F. Abel, and K. Bischoff, "Identifying users across social tagging systems," in *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media*, pp. 522–525, 2011.

[147] K. Gimpel, N. Schneider, B. O'Connor, D. Das, D. Mills, J. Eisenstein, M. Heilman, D. Yogatama, J. Flanigan, and N. A. Smith, "Part-of-speech tagging for twitter: annotation, features, and experiments," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 42–47, 2011.

[148] O. Owoputi, B. OConnor, C. Dyer, K. Gimpel, N. Schneider, and N. A. Smith, "Improved part-of-speech tagging for online conversational text with word clusters," in *Proceedings of NAACL-HLT*, pp. 380–390, 2013.

[149] R. M. Fano, "Transmission of information: A statistical theory of communications," *American Journal of Physics*, vol. 29, pp. 793–794, 1961.

[150] G. Salton, A. Wong, and C. S. Yang, "A vector space model for automatic indexing," *Commun. ACM*, vol. 18, pp. 613–620, Nov. 1975.

[151] G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., 1986.

[152] X. Shen, B. Tan, and C. Zhai, "Context-sensitive information retrieval using implicit feedback," in *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 43–50, 2005.

[153] X. Xue and W. Croft, "Automatic query generation for patent search," in *Proceedings of the 18th ACM conference on Information and knowledge management*, pp. 2037–2040, 2009.

[154] S. Riezler, A. Vasserman, I. Tsochantaridis, V. Mittal, and Y. Liu, "Statistical machine translation for query expansion in answer retrieval," in *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pp. 464–471, 2007.

[155] K. Massoudi, M. Tsagkias, M. de Rijke, and W. Weerkamp, "Incorporating query expansion and quality indicators in searching microblog posts," *Advances in Information Retrieval*, pp. 362–367, 2011.

[156] J. Shen, W. Geyer, M. Muller, C. Dugan, B. Brownholtz, and D. R. Millen, "Automatically finding and recommending resources to support knowledge workers' activities," in *Proceedings of the 13th international conference on Intelligent user interfaces*, pp. 207–216, 2008.

[157] S. Kullback, *Information theory and statistics*. Courier Dover Publications, 1968.

[158] B. Bigi, "Using kullback-leibler distance for text categorization," in *Advances in Information Retrieval*, pp. 305–319, Springer-Verlag, 2003.

[159] X. Zhou, X. Zhang, and X. Hu, "Semantic smoothing of document models for agglomerative clustering.," in *IJCAI*, pp. 2928–2933, 2007.

[160] D. Metzler, S. Dumais, and C. Meek, "Similarity measures for short segments of text," in *Advances in Information Retrieval*, pp. 16–27, Springer-Verlag, 2007.

[161] J. Jeon, W. B. Croft, and J. H. Lee, "Finding similar questions in large question and answer archives," in *Proceedings of the 14th ACM international conference on Information and knowledge management*, pp. 84–90, 2005.

[162] J. E. Gubernatis and T. E. Booth, "Multiple extremal eigenpairs by the power method," *J. Comput. Physics*, vol. 227, no. 19, pp. 8508–8522, 2008.

[163] F. Celli, F. Di Lascio, M. Magnani, B. Pacelli, and L. Rossi, "Social network data and practices: The case of friendfeed," *Advances in Social computing*, pp. 346–353, 2010.

[164] M. F. Porter, "An algorithm for suffix stripping," *Program: electronic library and information systems*, vol. 14, no. 3, pp. 130–137, 1980.

[165] A. Sieg, B. Mobasher, and R. Burke, "Web search personalization with ontological user profiles," in *Proceedings of the 16 ACM Conference on Information and Knowledge Management*, pp. 525–534, 2007.

[166] M. Renda and U. Straccia, "Web metasearch: rank vs. score based rank aggregation methods," in *Proceedings of the 2003 ACM Symposium on Applied Computing*, p. 846, 2003.

[167] C. Dede, "The evolution of constructivist learning environments: Immersion in distributed, virtual worlds," *Educational Technology*, vol. 35, no. 5, pp. 46–52, 1995.

[168] S. Riley, "Teaching in virtual worlds: Opportunities and challenges," *The Journal of Issues in Informing Science and Information Technology*, vol. 5, p. 127, 2008.

[169] B. Choi and Y. Baek, "Exploring factors of media characteristic influencing flow in learning through virtual worlds," *Computers & Education*, vol. 57, no. 4, pp. 2382–2394, 2011.

[170] M. Roussos, A. Johnson, T. Moher, J. Leigh, C. Vasilakis, and C. Barnes, "Learning and building together in an immersive virtual world," *Presence: Teleoperators & Virtual Environments*, vol. 8, no. 3, pp. 247–263, 1999.

[171] B. Dalgarno and M. Lee, "What are the learning affordances of 3-d virtual environments?," *British Journal of Educational Technology*, vol. 41, no. 1, pp. 10–32, 2010.

[172] A. Johnson, M. Roussos, J. Leigh, C. Vasilakis, C. Barnes, and T. Moher, "The nice project: Learning together in a virtual world," *Virtual Reality Annual International Symposium*, pp. 176–183, 1998.

[173] J. Gee, "What video games have to teach us about learning and literacy," *Computers in Entertainment (CIE)*, vol. 1, no. 1, p. 20, 2003.

[174] W. Johnson, J. Rickel, R. Stiles, and A. Munro, "Integrating pedagogical agents into virtual environments," *Presence*, vol. 7, no. 6, pp. 523–546, 1998.

[175] K. Leelawong and G. Biswas, "Designing learning by teaching agents: The betty's brain system," *International Journal of Artificial Intelligence in Education*, vol. 18, no. 3, pp. 181–208, 2008.

[176] F. Paas, A. Renkl, and J. Sweller, "Cognitive load theory and instructional design: Recent developments," *Educational psychologist*, vol. 38, no. 1, pp. 1–4, 2003.

[177] A. Carlson, J. Betteridge, R. C. Wang, E. R. H. Jr., and T. M. Mitchell, "Coupled semi-supervised learning for information extraction," in *Proceedings of the Third ACM International Conference on Web Search and Data Mining (WSDM 2010)*, 2010.

[178] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. H. Jr., and T. M. Mitchell, "Toward an architecture for never-ending language learning," in *Proceedings of the Twenty-Fourth Conference on Artificial Intelligence (AAAI 2010)*, 2010.

[179] B. Dalvi, W. Cohen, and J. Callan, "Websets: Extracting sets of entities from the web using unsupervised information extraction," in *Proceedings of the fifth ACM international conference on Web search and data mining*, pp. 243–252, 2012.

[180] C. Huang, T. Hong, and S. Horng, "Mining knowledge from object-oriented instances," *Expert Systems with Applications*, vol. 33, no. 2, pp. 441–450, 2007.

[181] S. Tirmizi, J. Sequeda, and D. Miranker, "Translating sql applications to the semantic web," in *Database and Expert Systems Applications*, pp. 450–464, 2008.