SEARCHING HETEROGENEOUS PERSONAL DATA

BY DANIELA QUITETE DE CAMPOS VIANNA

A dissertation submitted to the

School of Graduate Studies

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

Graduate Program in Computer Science

Written under the direction of

Amélie Marian

and approved by

_____

_____

_____

_____

New Brunswick, New Jersey

October, 2019

ABSTRACT OF THE DISSERTATION

Searching Heterogeneous Personal Data

by Daniela Quitete de Campos Vianna

Dissertation Director: Amélie Marian

Personal data is now pervasive, as digital devices are capturing every part of our lives. Users are constantly collecting and saving more data, either actively in files, emails, social media interactions, etc., or passively by GPS tracking of mobile devices, or records of financial transactions. Unlike traditional information seeking, which focuses on discovering new information, search on personal data is usually focused on retrieving information that users know exists in their own dataset, even though most of the time they do not have a perfect recollection of where it is stored. Attempting to retrieve and cross-reference personal information leads to a tedious process of individually accessing all the relevant sources of data and manually linking their information. In this scenario, traditional searches are often inefficient, making it critical for search tools to be capable of accessing heterogeneous and decentralized data in a flexible and accurate way by taking into consideration the additional knowledge the user is likely to have about the target information.

In this dissertation, we introduce a set of techniques that allow users to easily access their own data. We start by presenting a unified and intuitive multi-dimensional data model following a combination of dimensions that naturally summarize various aspects of the data collection: *who, when, where, what, why, how*. We then proceed by designing frequency-based scoring models that leverage the correlation between users (*who*), time (*when*), location (*where*), data topics (*what*), and provenance (*how*) to improve search over personal data. Since the scoring model proposed needs to generalize well over user-specific datasets, we extend the static scoring function by adopting a learning-to-rank approach using the state of the art LambdaMART algorithm. Due to the lack of pre-existing personal training data, a combination of known-item query generation techniques and an unsupervised ranking model (field-based BM25) is used to build our own training sets.

To validate the data and scoring models, we implemented tools for data extraction, classification, entity recognition, and topic modeling. A thorough qualitative evaluation performed over a publicly available email collection and a personal digital data trace collection from a real user show that our approach significantly improves search accuracy when compared with traditional personal search tools such as Apple's Spotlight and Apache Solr, and techniques like TF-IDF, BM25, and field-based BM25.

"We cannot expect in the immediate future that all women who seek it will achieve full equality of opportunity. But if women are to start moving towards that goal, we must believe in ourselves or no one else will believe in us; we must match our aspirations with the competence, courage and determination to succeed."

Rosalyn Yalow, medical physicist & 1977 Nobel Prize winner.

Acknowledgements

I would like to start acknowledging my advisor, Prof. Marian, for the guidance and attention over all these years. Thank you, Amelie, for the support during the most challenging years of my life, for being always patient and understanding, and for believing in my abilities to conclude this work.

Prof. Nguyen and Prof. Borgida, thank you for being a part of this incredible journey. For the most inspiring discussions and academic support.

I would not be here today if not for my dearest professors Alexandre Plastino, Otton Silveira and Vinod Rebello. Alexandre and Otton started my research career when I was still an undergraduate student; it was with their never ending support and encouragement that I took my first steps as a researcher. Vinod always had a special way to guide his students, through thought-provoking questions and great attention to details he became a reference for me. All my deep respect and appreciation to the faculty in the Department of Computer Science at Universidade Federal Fluminense (UFF).

I would like to thank the staff and faculty in the Department of Computer Science at Rutgers University for the support. Rutgers University, through the Rutgers Graduate School-New Brunswick, financially supported part of my years of study with a Teaching Assistantship.

I can not forget the New Brunswick Adult Learning Center and the English as a Second Language program. Thank you for the opportunity and for making me believe that I could one day become a Ph.D. student.

Thanks Ginger Cook, Jon, and the merry band of artists, you are so unique and so very dear to me. You never fail to make me smile. Thank you!!!

There are so many friends to thank, so let me begin by thanking my Rutgers friends Aritanan, Fabio and Ana Paula, Carlos, Janaina and Lucas. Thank you for the support and fun times! To my Rutgers labmates: Valia, Md, and Bill. To the amazing girls from the Rutgers International Women Group, in special Sahar, Hyosoo, and Suzette, you were the first ones to welcome me into my new life in the U.S., I am very thankful for each moment we have shared. My most sincere appreciation to the girls: Laura, Renata e Viviane. You are the best, your love and support carried me over those years despite the physical distance between us.

My parents, Antônio Cláudio (*in memory*) and Licia, your unconditional love brought me here. You taught me to fight for my dreams and you gave me all the tools and emotional strength to succeed in life. Dad, you have always celebrated my achievements and was one of the greatest supporters of my doctorate, your love and memory kept me going during the most challenging times. To my brothers Flávio and Luis Felipe, sister-in-laws Rita and Ana Claudia, niece Maria Eduarda, and nephews Matheus, Antônio Cláudio, and Pedro, you make my life lighter, full of love and fun. To my grandparents Neusa (*in memory*) and Carlos (*in memory*), and Alacyr (*in memory*) and Alamir (*in memory*). I love you all very much!!! To my uncles, aunts and cousins who have always cheered for me. To my in-laws Rubens and Ana Maria, and sister-in-law Ana Silvia. You are all very special to me.

Finally, I would like to thank my boys, Rodrigo, Daniel and Rafael. Daniel and Rafael, my Rutgers boys. Daniel, you came into my life when I was at the beginning of my Ph.D. journey; Rafael, I qualified while carrying you. It was scary, challenging, exhilarating, enthralling... I have learned so much from you

two and I cannot imagine my life without you. For you, I wanted to succeed; because of your love, I found a way through the most difficult times. My dear husband, my friend, my beloved colleague, my mountaineering partner, the most inspiring person in my life. Rodrigo, I cannot find words to express all my love for you. Together, with companionship and respect for each other's dreams, we braved the new life as parents and Ph.D. students. You were by my side every step of the way, from taking care of our babies to late hours of studies, making it possible for me to be the mom that I always dreamed to be without having to give up my professional dreams. Thank you, my love, for being the most incredible partner in this unpredictable and captivating journey that is life.

Dedication

To my husband and sons. I love you!

To my Mom, Dad (in memory), and Brothers.

Table of Contents

List of Tables

List of Figures

Chapter 1
Introduction

Personal data of our lives are constantly being produced and saved by users, either actively in files, emails, social media interactions, multimedia objects, calendar items, contacts, etc., or passively via various applications such as GPS tracking of mobile devices, records of usage, records of financial transactions, web search records or quantified self-sensor usage. These "personal digital traces" are typically (but not always) smaller, heterogeneous, and accessible through a wide variety of different portals and interfaces, such as web forms, APIs or email notifications; or directly stored in files used by apps on our devices. These traces reflect a chronicle of the user's life, keeping record of where the user went, who the user interacted with (online or in real-life), what the user did, and when. However, the large quantity of personal data available, and the fact that data may be stored in multiple decentralized systems, in heterogeneous formats, makes it challenging for users to interact with their data and perform even simple searches.

Personal Information Management is complicated by the sheer amount of data available, and by the fact that data is decentralized and heterogeneous. Attempting to retrieve and cross-reference personal information leads to a tedious process of individually accessing all the relevant sources of data, and manually linking their information. Under these circumstances, users have no hope of being able to easily locate past information unless they have a perfect recollection of where it is stored – an unlikely proposition when the amount of data stored is so

large, and may even be recorded without the user's input (e.g., GPS location).

Work in Cognitive Psychology [84, 17, 72, 49] has shown that contextual cues are strong triggers for autobiographical memories. Abowd *et al.* [3] and Dey [24] define context as any information that can be used to characterize *the situation of an entity* (person, place, object,...). This suggests that a natural way to remember and learn from past events is to include any pertinent contextual information when organizing and searching personal data. Personal information can be modeled, and indexed following six dimensions that mirror the basic interrogative words: *what, who, when, where, why, and how*. Each personal digital trace is a source of knowledge and can be related to different data traces by shared common information. For instance, a simple Facebook post may contain enough information to identify where a user went, what they did, who they interacted with, and when. Multiple traces, from the same or different data sources, are often related to each other. The correlation between data traces can be identified through common information such as time and location. Even though multiple data traces may share common information, they may have significantly different structures. This heterogeneity presents a major challenge.

Search of personal data is usually focused on retrieving information that users know exists in their own dataset, even though most of the time they do not know in which source or device they have seen the desired information. Personal search have been throughout studied in specific real-life scenarios as desktop search [29] and email search [42]. Current search tools such as Spotlight and Gmail search are not adequate to deal with this scenario where the user has to perform the same search multiple times on different services or/and devices rather than search over just a single service. Besides, traditional searches are often inefficient as they typically identify too many matching documents.

Our goal, with this dissertation, is to give back to individual users easy and flexible access to their own data by proposing a data model, search methodologies, and a series of tools that let user retrieve, store and organize their digital traces *on their own devices*, guaranteeing some clear privacy and security benefits.

1.1 Contributions

The process of unifying a user's personal information is an important step to address the decentralized and heterogeneous nature of personal data. In this dissertation, we present a unified and intuitive multidimensional data model following a combination of dimensions that naturally summarize various aspects of the data collection: *who, when, where, what, why, how*. The data model, called *w5h*, is used both to unify heterogeneous digital trace data from different sources, and to create links within the data, connecting relevant pieces of information together and also identifying possibly new connections within user data.

Based on the *w5h* data model we designed frequency-based scoring models that leverage the correlation between users (*who*), time (*when*), location (*where*), data topics (*what*), and provenance (*how*) to improve search over personal data. The first scoring model, called *w5h-f*, is a static function that focused around personal digital traces and as such includes specific group of correlations in the scoring. The second scoring model, called *w5h-l2r*, takes into consideration the fact that the scoring model proposed needs to generalize well over user-specific data sets, and so, we extend the static scoring function by adopting a learning-to-rank approach using the state of the art LambdaMART algorithm. The *w5h-l2r* approach uses a compact and efficient frequency-based feature space to rank query results over personal digital traces.

Learning-to-rank approaches have been very successful in solving real-world

ranking problems. However, the existing models for ranking are trained on either explicit relevance judgments (crowdsourced or expert-labeled) or clickthrough logs. In our scenario (personal digital traces), none of these is available nor pursuable. Human-labeled training sets are not available, in addition, there is a dearth of synthetic personal datasets and benchmarks. To overcome those challenges, the learning-to-rank approach, *w5h-l2r*, relies on a combination of known-item query generation techniques and an unsupervised ranking model (field-based BM25) to heuristically build our own training sets.

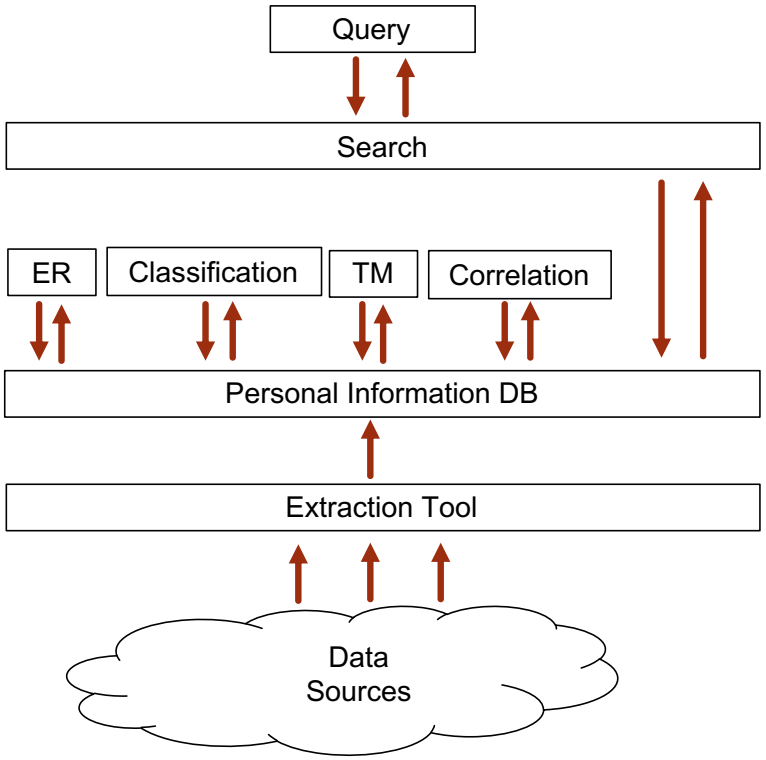


Figure 1.1: Architecture.

To validate the data and scoring models, we implemented tools for data extraction (Extraction Tool), classification (Classification), entity recognition (ER), and topic modeling (TM) as illustrated in Figure 1.1. The data extraction tool retrieves and stores the digital life of an user creating a personal information

database (PIM) that is robust, reliable and secure. For data classification we designed a machine learning multi-class classifier that automatically maps the raw data retrieved from each source into the *w5h* dimensions. The classification process is done without requiring user intervention. Entity resolution, topic modeling, and data traces correlation (Correlation) are pre-computed to support the proposed search approaches.

A thorough qualitative evaluation is performed using two different datasets: a real user dataset composed by data from a variety of data sources such as Facebook, Dropbox, and Gmail; and, the Enron Corporation dataset, composed by emails from around 158 employees. The efficacy of the *w5h-f* scoring models is evaluated by comparing its performance with two popular existing search tools, Solr [6] (using different scoring methodologies: TFIDF, BM25, and field-based BM25), and Spotlight [1]. We observe that by including pertinent contextual information when searching personal data, the *w5h* approaches can significantly improve accuracy. Also, the *w5h* approaches benefit for their ability to disambiguate/link people from different sources of data and, for including frequency information as part of their scoring results. The results for the *w5h* learning-to-rank approach, *w5h-l2r*, show that moderately large datasets can benefit from learning-to-rank techniques when paired with a representative feature set built from a novel frequency based feature space introduced in this dissertation.

1.2 Organization

This dissertation is organized as follows:

- In Chapter 2 we survey related work.
- A unified and intuitive multidimensional data model to link and represent

heterogeneous personal digital traces is introduced in Chapter 3. The model, called *w5h*, uses the six dimensions (*who, when, where, what, why, how*) to unify features of each personal data object, regardless of its source.

- In Chapter 4 we introduce a frequency-based scoring methodology for searching personal digital traces. The static scoring function, named *w5h-f*, is based on our multidimensional data model and leverages entities interactions within and across dimensions in the data sets.

- In Chapter 5, we propose a learning-to-rank frequency-based scoring methodology, called *w5h-l2r*. Our scoring model relies on a representative feature set to represent query-matching object pairs built upon a novel frequency-based feature space that leverages entities interactions within and across dimensions in the dataset. A novel combination of known-item query generation techniques and an unsupervised ranking model to heuristically generate labeled training sets is also proposed.

- Chapter 6 introduces an implementation of our techniques, from data extraction, to entity recognition, classification and retrieval, that will be used as the basis of our experimental evaluation.

- An experimental evaluation of our proposed *w5h-f* and *w5h-l2r* scoring techniques is presented in Chapter 7. The scoring approaches are compared against two popular existing search tools, Solr [6] and Spotlight [1], and techniques, TFIDF [71], BM25 [70] and field-based BM25, on real data using both manually designed and synthetically generated search queries. The evaluation was conducted using personal digital data traces datasets and the Enron dataset.

- Concluding remarks and future research topics are presented in Chapter 8

Chapter 2
Literature Review

2.1 Personal Information Management

Jones and Teevan offer a very thorough investigation of Personal Information Management. In particular, they discuss how search in Personal Information Management differs from search in traditional Information Retrieval systems or web search engines: For these re-finding tasks, studies have shown that users would rather not search their data via keyword searches, but prefer to find their information by retracing their steps [80], navigating [12, 56], or orienteering in their data, which provides a context to their searches. Furthermore, the search behavior in Personal Management Systems is highly individualized [38]. In [50], Jones discuss the future of personal information management considering how personal information is migrating onto the Web and being accessed through mobile devices. In [48], the authors proceeds to discusses the extensive variety of technologies for managing information, including search for personal data. Our work is related to the wider field of Personal Information Management [49], in particular, search behavior over personal digital traces is likely to mimic that of searching data over personal devices. Unlike traditional information seeking, which focuses on discovering new information, the goal of search in Personal Information systems is to find information that has been created, received, or seen by the user.

Bell has pioneered the field of life-logging with the project MyLifeBits [36, 9] for which he has digitally captured all aspects of his life. While MyLifeBits started as an experiment, there is no denying that we are moving towards a world where all of our steps, actions, words and interactions will be recorded by personal devices (e.g., Google Glasses, cell phones GPS systems, FitBit and other Quantified Self sensors,...), or by public systems (e.g., traffic cameras, surveillance systems,...), and will generate a myriad of digital traces. *digi.me* [25] is a commercial tool that aims at extending Bell's vision to everyday users. The motivations behind *digi.me* are very close to ours; however *digi.me* currently only offers a keyword- or navigation-based access to the data; search results can be filtered by service, data type or/and date.

This vision of "Total Capture," where all data is kept for every user has its detractors. Sellen and Whittaker [73] argue that rather than storing a complete lifelog, Personal Information Management systems should focus on selectively identifying effective retrieval cues to jog users memories, and that life-logging systems should not replace human memory but rather support it. They suggest that Personal Information Systems should be designed with an understanding of which memory tasks are targeted: recollection, reminiscence, retrieval, reflection or remembering intentions. We design our system with a focus on retrieval through recollection (retracing digital steps) and reflection (learning from past experiences).

2.2 Context-aware Personal Data Model

The case for a unified data model for personal information was made in [54, 55, 87]. deskWeb [89] looks at the social network graph to expand the searched data set to include information available in the social network. Lifestreams [33] organizes

desktop content in time-oriented streams. Haystack [55] argues for a uniform semi-structured data model for user information. Stuff I've Seen [29] indexes all of the information the user has seen, regardless of its location or provenance, and uses the corresponding metadata to improve search results. MyLifeBits [36] aims at capturing all data generated by the user. Seetrieve [39] extends on this idea by only considering the parts of documents that were visible to the user to infer task-based (*"why"*) context to the file for later retrieval. Most notably, Personal Dataspaces [43, 28, 27, 26, 14] propose semantic integration of data sources to provide meaningful semantic associations that can be used to navigate and query user data (implicit context). Connections [77] uses system activity to make similar connections between files; [74] extends this approach to consider causality, using data flow, as contextual information. Most of these systems were developed before the advent of cloud-based storage, and assume that most data is available locally, or easily retrieved. In addition, while several do offer an integrated data model, their query models are typically keyword based, with sometimes one source of context (e.g. tasks for Seetrieve, time for Lifestreams) used to aid the search. In contrast we envision a retrieval process that follows the memory process and uses all types of contextual cues.

Contextual information has been considered in various computer science applications. Abowd *et al.* [3] and Dey [24] define context as any information that can be used to characterize *the situation of an entity* (person, place, object,...). Context-aware applications dynamically adapt to changes in the environment in which they are running: location, time, user profile, history. In depth surveys of context-aware models and systems are given in [15, 8]. Truong and Dustdar survey context-aware Web-Service systems in [61]. The value of contextual information in searching and browsing user behavior on web is also explored in [59, 58, 66].

Context-awareness has become increasingly popular with the wide adoption of mobile devices. While the types of context these systems consider overlap with ours, the overall approach is different from ours, for instance a contextually-aware Information Retrieval system will use the current context (e.g., user location and time of day) to adjust search results [75]. In contrast, we consider context as information that can be queried and used to guide the search.

Other file system related projects have tried to enhance the quality of search within the file system by leveraging the context in which information is accessed to find related information [20, 40] or by altering the model of the file system to a more object-oriented database system [16]. YouPivot [41] indexes all user activities based on time and uses the time-based context to guide searches. Social context (users' friends and communities) is leveraged in [76] for information discovery; similarly [23] uses temporal and location context to aid discovery in social media data. Our work integrates all these sources of contextual information and provides a unified complete model of context-aware personal data.

2.3 Personal Information Search

In this dissertation, we aim to provide comprehensive and intuitive scoring and search strategies for search of users' digital memories. Our search techniques are related to various work in top-k query processing techniques [32, 63, 60], which consider various dimensions as part of an object's score. Also related is the problem of identifying keyword query results in RDBMSs and ranking them based on some quality metric [5, 13, 47, 46], but these only focus on matching content keywords and have simple ranking techniques based on distance.

Several index structures have been proposed for text approximation [65],

querying text within a structure context [62, 53, 68, 44], and querying structure [37, 86]. Learning and query selectiveness based ranking techniques for desktop search are proposed in [21]; however, their ranking formula uses a simple linear function to aggregate weights for various file features (filename, size, date of creation, etc.).

Learning-to-rank approaches, as RankNet, LambdaRank, and LambdaMART [19], have proved to be very efficient to solve ranking problems. LambdaMART is the boosted tree version of LambdaRank, which is based on RankNet [18]. In [18], a probabilistic cost for training systems to learn ranking functions using pairs of training examples was used in a neural network model, RankNet, with the intention of minimizing the number of inversions (incorrect order among pairs of results) in ranking. LambdaRank improves RankNet by realizing that in order to train a model there is no need to use the costs themselves, only the gradients of the costs with respect to the model scores. LambdaMART combines MART(Multiple Additive Regression Trees) [35] and LambdaRank. While MART uses gradient boosted decision trees for prediction tasks, LambdaMART uses gradient boosted decision trees using a cost function derived from LambdaRank for solving a ranking task.

Email search is a type of personal search that has been well studied. [42] presents a learning-to-rank approach that improves the default ranked-by-time search by taking into consideration time recency and textual similarity to the query. [85] addresses the problem of learning-to-rank from click data in personal search. [88] explores how to effectively leverage situational contextual features (e.g. time of a search request and the location of the user while submitting the request) to improve personal search quality. In [10] the authors leverage user interaction data in a privacy preserving manner for personal search by aggregating

non-private query and document attributes across a large number of user interactions. In our scenario, each dataset is comprised by data from only one user, and so it is private by design, not being possible for us to leverage interactions from other users.

In [22] the authors use classic unsupervised IR models, such as BM25, as a weak supervision signal for training deep neural ranking models. In this context, weak supervision refers to a learning approach that creates its own training data by heuristically retrieving documents for a large query set. Three different neural network-based ranking models are presented, a point-wise ranking model and two pair-wise models. Combinations of neural models with different training objectives and input representations are compared against each other and against the baseline, BM25. The experiments showed that their best performing model significantly outperforms the BM25 model. In our work, we use a similar approach to retrieve matching objects to a given query in order to build our own training and evaluation sets.

There is a dearth of synthetic data sets and benchmarks to evaluate search over personal data. In [57], the authors describe methods for generating test collections for search experiments. In [7], known-item query generating techniques, as discussed in [30], are used to heuristically generate query sets. In this work, we use a similar approach to automatically generated larger sets of known-item queries to validate the search methods proposed.

Chapter 3
Data Model

Personal data is heterogeneous and distributed, the difficulty lies in integrating the data across sources and also, from the same source, unifying their schemas and linking entities into a unified data set. With that in mind, we propose a data model that relies on the context in which personal data traces are created, produced and gathered to integrate heterogeneous traces into a unified data model that will support accurate searches. The proposed model, called *w5h*, was derived from the following observations:

1. Personal digital traces are rich in contextual information, in the form of metadata, application data, or environment knowledge.

2. Personal digital traces can be represented following a combination of dimensions that naturally summarize various aspects of the data collection: *who, when, where, what, why* and *how*.

Work in Cognitive Psychology [84, 17, 72, 49] has shown that contextual cues are strong triggers for autobiographical memories. For instance, when an object is lost, e.g. keys, is common for a person to ask herself questions as: "When was the last time I remember using my keys?", "Who was there with me?", "What was I doing the last time I remember seeing my keys?". By retracing their steps and using the correct context, an individual is capable of finding the lost object. This suggests that a natural way to remember and learn from past events

is to include any pertinent contextual information when organizing and searching personal data. Abowd et al. [3] and Dey [24] define context as any information that can be used to characterize the situation of an entity (person, place, object,...). Context can be explicit, as the metadata information stored by the file system or application (e.g., timestamp, GPS location, tags, directory structure). It can also, be identified through application-based semantic information (e.g., email recipients, calendar meeting participants, check-in location) or it can be inferred, i.e., any information not directly connected to the data itself but that represents knowledge about the environment of the data collection. This could be related to the system environment, (e.g., which applications/documents were opened concurrently with a given document), social environment (e.g., which Facebook members had access to an event), or to the real world environment, (e.g., who was physically in the room or what the weather was when a given piece of data was collected).

Personal-information contextual data can be modeled following six dimensions that mirror the basic interrogative words: *what, who, when, where, why* and *how*. Answers to questions such as "when was an email sent", "who was involved in a conversation", "where a meeting took place", "what a file contains", "how the information was recorded", could help users to find data they remember having stored or accessed in the past, and also, it could support the process of inferring knowledge from their personal databases and their interactions with their data. Our *w5h* model uses these six dimensions as the unifying features of each personal digital trace object, regardless of its source. Using these natural questions as the main facets of data representation will also allow the combination of our data representation with a natural and intuitive query model for searching information in digital traces. Listed below are some examples of dimensional data that can

be extracted from a user's personal digital traces:

- **what**: content

  Messages, messages subjects, publications, description of events, description of users, list of interests of a user.

- **who**: with whom, from whom, to whom

  User names, senders, recipients, event owners, lists of friends, authors.

- **where**: physical or logical, in the real-world and in the system

  Hometown, location, event venue, file/folder path, URL.

- **when**: time and date, but also what was happening concurrently

  Birthday, file/message/event created-/modified-time, event start/end time.

- **why**: sequences of data/events that are causally connected.

- **how**: application, device, environment.

To illustrate how information on digital data traces can be associated with one of the six dimensions (*what, who, when, where, why* and *how*), we present 3 different examples: a Facebook post (Figure 3.1), a Gmail message (Figure 3.2) and a Google Calendar event (Figure 3.3). For all those data traces, each piece of information was identified as belonging to one of the six dimensions proposed. Even though multiple digital traces come from different sources and have their own data schema, they can be unified using the six dimensions proposed in our *w5h* model. For instance, the Facebook post in Figure 3.1 can be linked by our unified model to the Gmail message in Figure 3.2 and Google Calendar event in Figure 3.3, since they all have John Smith under the same dimension *who*.

Text messages are usually classified as *what*; however, implicit context derived from content could be classified differently. For instance, in Figure 3.1, the

```
{
message: March for Science in Seattle                          (WHAT)
from: John Smith                                               (WHO)
place: Seattle, Washington                                     (WHERE)
with_tags: Anna Smith                                          (WHO)
story: John Smith and Anna Smith in Seattle, Washington        (WHAT)
created_time: 2017-04-22T22:43:56+0000                         (WHEN)
data_type: Facebook post                                       (HOW)
}
```

Figure 3.1: Simplified example of a user Facebook post classified according to the *w5h* model.

```
{
from: John Smith                                               (WHO)
to: Anna Smith                                                 (WHO)
date: 2017-04-20T10:30:00+0000                                 (WHEN)
subject: March for Science                                     (WHAT)
body: Are you planning to join the March for Science this
weekend? If yes, we could go together.                         (WHAT)
data_type: Email                                               (HOW)
}
```

Figure 3.2: Simplified example of a user email (Gmail) classified according to the *w5h* model.

```
{
created: 2017-04-01T13:45:00+0000                  (WHEN)
summary: March for Science                         (WHAT)
organizer: John Smith                              (WHO)
start: 2017-04-22T22:43:56+0000                    (WHEN)
data_type: Google calendar event                   (HOW)
}
```

Figure 3.3: Simplified example of a user event (Google Calendar) classified according to the *w5h* model.

message "March for Science in Seattle" gives both *what* ("March for Science") and *where* ("Seattle"). Implicit context could be derived using techniques such as Named Entity Recognition (NER), which tries to identify identify and classify entities into categories such as persons (*who*), locations (*where*), times (*when*), etc.

The *w5h* data model is used both to unify heterogeneous digital trace data

from different sources, and to create links within the data, connecting relevant pieces of information together and also identifying possibly new connections within user data. Having defined the model, we still have to find a good mechanism to translate heterogeneous digital data traces into the proposed *w5h* contextual model. We will discuss 2 different solutions in Chapter 6. The first solution is a static version of our *w5h* classifier and requires human intervention. The second solution is a machine learning multi-class classifier that automatically maps the data from any data source into the *w5h* dimensions. This last version does not require human intervention.

In Chapter 4 and Chapter 5, we will introduce two frequency-based scoring models for personal data search.

Chapter 4

A Frequency-based Scoring Methodology for Personal Data Search

We leverage the *w5h* model presented in Chapter 3 to provide rich and accurate search capabilities over personal digital traces. Unlike Web search, where the focus is often on discovering new relevant information, search in personal data sets is typically focused on retrieving relevant information that the user knows exists in their data set. Besides, users have unique habits and interpretations of their own data. In this scenario, standard search techniques are not ideal as they do not leverage the additional knowledge the user is likely to have about the target object, or the connections between objects pertaining to a given user.

As pointed in [84], users tend to remember their actions using the six natural questions; thus, using them to guide search is a logical approach. We now evaluate the potential benefits of the *w5h* model for integrating and searching personal data. Specifically, we propose a search mechanism that supports queries containing conditions along each of the six interrogative dimensions. In this chapter, we will detail our first scoring model based on a novel frequency-based scoring methodology over the *w5h* data model, called *w5h-f*. This work was presented in [82].

4.1 Scoring Methodology

To illustrate our query and scoring methodology let us consider the following search scenario: the user is interested in message(s) from John Smith or/and Anna Smith about a 2016 bike ride. We consider each digital trace to be a distinct object that can be returned as the result to a query.

Definition 1 (Object in *w5h* Integrated Dataset)**.** *An object O in the dataset is a structure that has fields corresponding to the 6 dimensions mentioned earlier. Each of these dimensions contains 0 or more items (corresponding to text, entities identified by entity resolution, times, locations, etc). The fields of an object O are accessed using functions O.get("who"), O.get("what"), etc.*

Formal queries have the same structure as objects in the unified dataset. In the example above, the query has three filled dimensions: bike ride (**what**); John Smith, Anna Smith (**who**); 2016 (**when**).

Definition 2 (Query)**.** *A query Q over the dataset is represented as an object as defined in Definition 1.*

Given objects Q and O, O is considered as an answer to object Q treated as a query if it contains at least one of the dimensions specified in Q. In looking for (partially) matching objects to a given query, each dimension will be searched separately, and the results will be combined according to a scoring function, generating a rank-ordered list of candidates. The choice of scoring function can be application dependent. We propose our frequency-based scoring function, *w5h-f*, below.

4.2 Frequency-based Multi-dimensional Scoring: w5h-f

Because personal digital traces are byproducts of users' actions and events, they are not independent objects. Our intuition is that the correlation between traces (objects) can be leveraged to improve the accuracy of search results. For example, if the "bike ride" query from Section 4.1 returns several potential matches, one from Alice Jones, and one from Bob White, we may want to score the one from Alice higher if she communicates more frequently as a group with the user, Anna Smith, and John Smith, than Bob White.

Our *w5h-f* scoring scheme uses the correlation between users (or entities) and how they interact over time to rank an object. Because we are focusing on personal digital traces, all the data articulates around a user. By analyzing personal datasets, we observed a strong correlation between the user (owner of the data) and multiple users (*who* groups), through times (*who, when*), location (*who, where*) and data sources (*who, how*). Our scoring exploits those interactions and correlations by way of a frequency score. Frequencies can be computed for individual users or group of users. They can be associated with multiple times, multiple data sources, and also with a set of locations. For example, from a set of emails exchanged between a group of users, we can extract the frequency (number of interactions) with which those users communicated, and in which time period those interactions occurred. In short, frequency expresses the strength of relationships, based on users, time, location and data sources (*who, when, where, how*).

Algorithm 1 shows how frequencies are computed across multiple dimensions. Initially, a list of objects is retrieved for each data source. For each object, the algorithm extracts groups of users, times and locations. Then, the following frequencies are computed:

- Frequency of each individual user: number of objects that mention a user in the *who* dimension.
- Frequency of a group of users: number of objects mentioning a group of users. If {a,b,c} is the group mentioned, frequencies of subgroups of {a,b,c}, e.g. {a,b} and {b,c}, are not counted.
- Frequency of each individual user at specific times: number of objects that mention a user at matching times. Time is normalized, so variations are also considered. For instance, a query searching for June, will match objects with time June 2016 and June 2017.
- Frequency of a group of users at specific times: number of objects mentioning the group at a specific time.
- Frequency of a location: number of objects that mention a location.

Besides computing the frequencies per source, we also compute the total frequency of a user, group of users, times and locations by combining the individual results obtained for each data source. For simplicity, in Algorithm 1, every time a user or group of users has an interaction, the frequency is increased by one; however, in practice, the algorithm allows us to weigh differently distinct types of interactions. For example, likes or comments on a Facebook post could be weighed differently, giving more relevance to interactions coming from comments than likes. Different roles, e.g. From and To in an email, can also be weighed differently.

Definition 3 (Similarity Score)**.** *Given a query Q, an object O, and the frequencies above, we define:*

$$
\begin{aligned}
f\text{-}score(Q,O) &= f[g] \\
&+ \sum_{u \epsilon who} f[u] \\
&+ \sum_{u \epsilon who} f_s[u] \\
&+ \sum_{\substack{u \epsilon who \\ dt \epsilon when}} f[u][dt] \\
&+ \sum_{\substack{u \epsilon who \\ dt \epsilon when}} f_s[u][dt] \\
&+ \sum_{\substack{g \epsilon who \\ dt \epsilon when}} f[g][dt] \\
&+ \sum_{addr \epsilon where} f[addr] \\
&+ score_{when}(dt,O) \\
&+ score_{how}(s,O) \\
&+ score_{what}(O)
\end{aligned}
$$

where g is the group of users in the who *dimension of O, u is each user in g, dt is each time in the* when *dimension, s is a data source, addr is each location in the* where *dimension, $f[g]$ is the frequency of a group of users in the same object, $f[u]$ is the total frequency of each user across all data services, $f_s[u]$ is the frequency of each user in the data source s of the object, $score_{when}(dt,O) = 1$ when the date dt from query Q matches object O; otherwise, $score_{when}(dt,O) = 0$, $f[u][dt]$ is the total frequency of the user u in the time dt across all data sources, $f_s[u][dt]$ is the frequency of the user u in the time dt and data source s of the object, $f[g][dt]$ is the total frequency of the group of user g in the time dt, $f[addr]$ is the frequency of each location addr, and $score_{how}(s,O)$ is the score of an object O for a given source s: $score_{how}(s,O) = 1$ when the service s from query Q matches object O;*

otherwise, $score_{how}(s, O) = 0$. *Lastly,* $score_{what}(O)$ *is a text-based score for object* O*, using any chosen scoring function (e.g., TFIDF, BM25,...).*

The equation in Definition 3 assumes that a query Q has all 5 dimensions *what, who, when, where* and *how*; if a dimension does not exist in a query, the equation term corresponding to that dimension will be 0.

Algorithm 1 Frequency algorithm

```
procedure COMPUTE–FREQUENCY(SOURCE)
                          ▷ object(source) retrieves all objects from a given source
    for each O ∈ object(source) do
        group ← O.get('who')
        times ← O.get('when')
        locations ← O.get('where')
        for each time ∈ times do
                                        ▷ Frequency of a group of users given a time
            f[group][time] ← f[group][time] + 1
            for each user ∈ group do
                                                ▷ Frequency of user given a time
                f[user][time] ← f[user][time] + 1
            end for
        end for
        for each user ∈ group do
                                                              ▷ Frequency of a user
            f[user] ← f[user] + 1
        end for
                                                          ▷ Frequency of group of users
        f[group] ← f[group] + 1
        for each location in locations do
                                                                ▷ Frequency of location
            f[location] ← f[location] + 1
        end for
    end for
end procedure
```

Let us consider the query Q_0 (*what:* bike ride; *who:* John Smith, Anna Smith; *when:* 2016), and the object O_1 illustrated in Figure 4.1. According to the *w5h-f* methodology, the object O_1 will have the following score:

$$
\begin{aligned}
f\text{-}score(Q, O_1) &= f[g = \text{John S., Anna S.}] \\
&+ f[u = \text{John S.}] + f[u = \text{Anna S.}] \\
&+ f_s[u = \text{John S.}] + f_s[u = \text{Anna S.}] \\
&+ f[u = \text{John S.}][dt = 2016] \\
&+ f[u = \text{Anna S.}][dt = 2016] \\
&+ f_s[u = \text{John S.}][dt = 2016] \\
&+ f_s[u = \text{Anna S.}][dt = 2016] \\
&+ score_{when}(2016, O) \\
&+ f[g = \text{John S., Anna S.}][dt = 2016] \\
&+ score_{what}\text{“bike ride”}
\end{aligned}
$$

where $s =$ Facebook

```
Message: Bike ride around lake Washington          (WHAT)
From: John Smith                                   (WHO)
Place: Seattle, Washington                         (WHERE)
With_tags: Anna Smith                              (WHO)
Created_time: 2016-06-20T00:21:27Z                 (WHEN)
Data_type: Facebook Status                         (HOW)
```

Figure 4.1: Simplified example of a user Facebook status update about a bike ride around Lake Washington.

The *w5h-f* scoring model is focused around personal digital traces and as such we included specific group of correlations in our scoring. Other application scenarios could also benefit from our *w5h*, with other group and pairwise correlations highlighted in a dedicated frequency-based scoring. For instance, traces from weather sensors could have strong pairwise *(where,when)*, or *(where, how)* correlations. In Chapter 5, we extend our static scoring function by adopting

a more general group of correlations and a learning-to-rank approach using the state of the art LambdaMART algorithm.

In Chapter 7, we will use real user datasets to validate our *w5h-f* scoring model by comparing it against state of the art search approaches as Apple's Spotlight and Apache Solr, and techniques like TF-IDF and BM25.

Chapter 5

A Frequency-based Learning-To-Rank Approach for Personal Data Search

In Chapter 4 we have discussed how search in personal data differs from Web search by focusing on retrieving data that the user knows exists in their dataset. We also showed that by leveraging the additional knowledge the user is likely to have about the target object, and the existent connections between objects pertaining to a given user, we can considerably improve personal data search accuracy. As personal digital traces are very specific to each user and are constantly evolving over time, it is necessary to find a scoring model that can generalize well over user-specific datasets. Learning-to-rank approaches have proved to be very efficient to solve ranking problems. However, the existing models for ranking are trained on either explicit relevance judgments (crowdsourced or expert-labeled) or clickthrough logs. In our scenario (personal digital traces), none of these is available nor pursuable. Human-labeled training sets are not available, in addition, there is a dearth of synthetic personal datasets and benchmarks. To overcome those challenges, we propose a learning-to-rank approach that relies on a combination of known-item query generation techniques and an unsupervised ranking model (field-based BM25) to heuristically build training sets. Furthermore, in this chapter, we extend our set of frequency-based features taking into consideration the correlation between content (*what*), users (*who*), time (*when*), location

(*where*) and data source (*how*). We use a state-of-the-art learning-to-rank algorithm based on gradient boosted decision trees, LambdaMART [19], to learn a ranking model to map feature vectors to scores.

In this chapter, we make the following contributions:

- A representative feature set to represent query-matching object pairs built upon a novel frequency-based feature space that leverages entities interactions within and across dimensions in the dataset.
- A novel combination of known-item query generation techniques and an unsupervised ranking model to heuristically generate labeled training sets.
- A quantitative evaluation of the proposed search technique, as well as comparison with two popular search methodologies: *BM25* and *field-based BM25*. Our results show that moderately large personal datasets can benefit from state-of-the-art learning techniques when combined with a compact frequency-based feature set.

5.1 Scoring Methodology

The scoring methodology for the learning-to-rank approach is the same as the methodology presented for the *w5h-f* scoring model introduced in Chapter 4. In short, given objects Q (Definition 2) and O (Definition 1), O is considered as an answer to object Q treated as a query if it contains at least one of the dimensions specified in Q.

5.2 Frequency-based Features

The *w5h-f* scoring function introduced in Chapter 4 relies on the correlation between traces (objects) to improve the accuracy of search results. However, for the *w5h-f* scoring model only a specific group of correlations is considered. In this section, we expand our set of correlations by exploring all possible relationship between users (*who*), time (*when*), location (*where*), topics (*what*) and data sources (*how*). As we did before, we exploits those interactions and correlations by way of a frequency score. To keep it simple, every time an interaction occurs, the frequency score is increased by one. For each dimension and combination of dimensions we compute a score that will be used later as features to represent the input data in our learning-to-rank approach.

For the scoring model proposed in this chapter, called *w5h-l2r*, we use a set of 34 features to represent the input data. The feature set is comprised by 30 features resulting from all possible combinations between the dimensions *who, what, when, where* and *how* plus 4 extra features that model the correlation between group of users (*who groups*); group of users and time (*who groups, when*); group of users and data source (*who groups, how*); and finally, group of users, time and location (*who groups, when, where*). The feature vector is defined in Definition 4.

Definition 4 (Feature Vector)**.** $\boldsymbol{x} = [x_1 \dots x_{34}]$ *is a feature vector comprised by 34 frequency-based features. Each feature* x_i *is computed by a frequency function* $f(S_i, Q, O)$*, where* $S_i \in \mathcal{S}$*.* $\mathcal{S}$ *represents all possible combinations between the 5 dimensions* who, what, when, where *and* how*, plus the 4 extra features that model the correlation between group of users.* Q *is a query (Definition 2) and* O *is an object in the user dataset (Definition 1).*

To illustrate our query and scoring methodology consider the following search

```
From: John Smith                                  (WHO)
To: Anna Smith                                    (WHO)
Date: 2018-09-04T10:30:00+0000                    (WHEN)
Subject: Lunch                                    (WHAT)
Body: Do you want to get something to eat?        (WHAT)
```

Figure 5.1: Simplified example of a user email message classified according to the 6 contextual dimensions model.

scenario: the user is interested in a message from 2018 (*when*), sent by John (*who*), about the topic "Lunch" (*what*). We can define query Q_1 as (*when*: 2018, *who*: John, *what*: Lunch). By definition, the object in Figure 5.1 (O_1) is a matching to the given query (Q_1) containing all dimension/item specified in the query – *when*:2018, *who*:John, and *what*:"Lunch". The query-object pair (Q_1, O_1) can be represented by a 34 frequency-based feature vector $\mathbf{x} = [x_1 \dots x_{34}]$ as introduced in Definition 4. Each feature x_i represents the frequency score for a set of dimensions S_i, query Q_i and object O_i:

$$x_1 = f((\text{what:Lunch}), Q_1, O_1)$$

$$x_2 = f((\text{who:John}), Q_1, O_1)$$

$$x_3 = f((\text{when:2018}), Q_1, O_1)$$

$$x_6 = f((\text{what:Lunch, who:John}), Q_1, O_1)$$

$$x_7 = f((\text{what:Lunch, when:2018}), Q_1, O_1)$$

$$x_9 = f((\text{what:Lunch, how:Gmail}), Q_1, O_1)$$

$$x_{10} = f((\text{who:John, when:2018}), Q_1, O_1)$$

$$x_{12} = f((\text{who:John, how:Gmail}), Q_1, O_1)$$

$$x_{16} = f((\text{what:Lunch,who:John, when:2018}), Q_1, O_1)$$

$$x_{18} = f((\text{what:Lunch,who:John, how:Gmail}), Q_1, O_1)$$

$$x_{20} = f((\text{what:Lunch,when:2018, how:Gmail}), Q_1, O_1)$$

$$x_{23} = f((\text{who:John, when:2018, how:Gmail}), Q_1, O_1)$$

$$x_{27} = f((\text{what:Lunch,who:John,when:2018,how:Gmail}), Q_1, O_1)$$

If a set of dimensions S_i is not present in query Q_i and object O_i, the frequency score $f(S_i, Q_i, O_i) = 0$.

To understand how frequencies ($f(S_i, Q_i, O_i)$) are computed, consider the following example: lets assume a dataset D containing 10 objects that mention John under the *who* dimension, being 4 of those 10 objects from Facebook and the remaining 6 from Gmail. Given object O_1 and query Q_1 from the previous example, we can say that the frequency of John (*(who:John)*) in dataset D for query Q_1 and matching object O_1 is $x_2 = f((\text{who:John}), Q_1, O_1)$, where $f((\text{who:John}), Q_1, O_1) = 10$. We can also say that the frequency of John in Gmail (*(who:John,how:Gmail)*) in dataset D for query Q_1 and matching object O_1 is $x_{12} = f((\text{who:John, how:Gmail}), Q_1, O_1)$, where $f((\text{who:John, how:Gmail}), Q_1, O_1) = 6$

5.3 Scoring the *What* Dimension

The *what* dimension in the six-dimension model is composed of content information comprising mostly of text. Based on that fact, we use two standard text approaches to link and score objects for the *what* dimension: field-based BM25 and topic modeling [79].

Field-based BM25. A field-based BM25 is a state-of-the-art TF-IDF type of ranking function that takes into consideration the document structure. In

our scenario, the fields in the field-based BM25 correspond to the 5 dimensions proposed, *what, who, when, where* and *how*. To compute the field-based BM25 score for the *what* dimension, we use a popular full-text search platform from the Apache Lucene project, Solr [6]. All data retrieved for a user is unified and parsed according with the six dimensions and then, exported to Solr. For each user query, we search Solr using the values from the *what* dimension, getting as a result a partial list of matching documents with its respective field-based BM25 score. Even though Solr contains the data for all 5 dimensions, we are only interested in using field-based BM25 to score the *what* dimension, since this dimension contains most of the content of an object. For the remaining dimensions, we use our frequency-based function as introduced in Section 5.2.

Topic Modeling. A "Topic" consists of a cluster of words that frequently occur together. Topic models use contextual cues to find connections between words with similar meanings and to distinguish between use of words with multiple meanings. Given a document, we would like to identify what possible topics have generated that data. In our case, topic modeling would be an important feature to connect different objects, including objects from different data sources. The association between topics (*what*), user (*who*), times (*when*), location (*where*) and source (*where*) could shed some light on finding objects that could be a better matching to the user query. To define topics for each object in the user dataset, we use a topic model package called MALLET and a text collection built from the content classified under the *what* dimension for each object in the user data set. The MALLET [64] topic model package includes a fast and scalable implementation of Gibbs sampling. The Gibbs Sampling algorithm considers each word token in the text collection in turn, and estimates the probability of assign the current word token to each object, conditioned on the topic assignments to all other word

tokens. For each object in the user dataset, MALLET computes the topic composition of documents. We use the most relevant topic for each document to cluster documents per topic. For each document in a topic, we extract the person/entity mentioned in *who* dimension, the times from *when*, location from *where* and source from *how*. With that information, we are able to build the correlation between person/entity, times, location and source for each topic (*what*). Also, we are able to estimate the frequency of those correlation/interactions using the frequency function presented in Section 5.2. Besides the topic composition of documents, MALLET also outputs the words in the corpus with their topic assignments and frequencies. We use this list of words per topic and the words specified in the user query (for the *what* dimension), to find the topic that are a more close representative of the user query. Then, we can use the topic that matches the query to find out a partial list of documents that are matching candidates to the query, based solely on the contents of the *what* dimension.

```
Message: Conference Center. View of the park.   (WHAT)
From: John Smith                                (WHO)
Place: Paris, France                            (WHERE)
With_tags: Anna Smith                           (WHO)
Created_time: 2018-09-05T11:00:00+0000          (WHEN)
Data_type: Facebook Status                      (HOW)
```

Figure 5.2: Simplified example of a user Facebook post classified according to the 6 contextual dimensions model.

To illustrate how topic modeling can support our search, consider T a topic composed by the following key words: hotel, lunch, street, trip, miles, view, lake, ride, restaurant and conference. Assuming that topic T is the most relevant topic for object O_1 in Figure 5.1, and object O_2 in Figure 5.2, we can say that objects O_1 and O_2 are correlated by their *what* dimension. By considering all objects

(documents) clustered under the same topic T, we can learn how strong person/entity (*who*), times (*when*), location (*where*) and source (*how*) are connected with relation to a topic (*what*). Again, this strength is measured by a way of a frequency score as presented in Section 5.2.

5.4 Learning-to-Rank Model

In the previous sections, we explained how query-document pairs are represented by a feature vector built upon our frequency-based feature space. To map the feature vector to a real-valued score we need to train a ranking model. Our choice of learning-to-ranking algorithm is the state-of-the-art LambdaMART [19]. LambdaMART uses gradient boosted decision trees, which incrementally builds regression trees trying to correct the leftover error from the previous trees. At the end, the prediction model is an ensemble of weaker prediction models that complement each other for robustness. During a training phase, we must define the best set of parameters that results in a robust and accurate model. For this cross-validation stage, we will consider the following parameters:

- tree: number of trees in the ensemble
- leaf: maximum number of leaves per tree
- mls: minimum number of samples each leaf has to contain
- shrinkage: learning rate
- metric: training metric to be optimized for

In the next section, we will discuss how training and evaluation sets can be built for personal data search, a scenario where publicly available personal training data does not exist.

5.5 Query Sets

The existent learning-to-rank models are trained on either explicit relevant judgments (crowdsourced or expert-labeled) or clickthrough logs. Due to privacy issues and the specialized and individualized nature of personal datasets, human-labeled training sets are non existent. To overcome this problem, we propose a novel heuristic to generate training data based on known-item query generation techniques and an unsupervised ranking model (field-based BM25). The proposed method is presented in Section 5.5.1.

5.5.1 Training Query Set

In a learning-to-rank algorithm, each pair of query-document(object) is represented by a vector of numerical features. In addition to the feature vector, pairs of query-documents could be augmented with some relevance information. Then, a model has to be trained to map the feature vector to a score. One of the challenges of using learning-to-rank for personal data search is to be able to build a training set without human intervention or any external information (e.g., expert labeling or click data). To this end, in this section we present a combination of heuristics that given a user dataset is able to simulate a human-labeled training set to tailor the learning model to each specific user dataset.

Considering the fact that personal data trace search is a known-item type of search, simulated queries can be automatically generated, using known-item query [30] generation techniques such as the ones presented in [7] and [57]. In this work, queries are created by randomly choosing a set of dimensions (*who, what, when, where, how*) and values/items (e.g. email's Subject, Facebook post's content) from a target object, as described in Algorithm 2. Each call to Algorithm 2

will result in a query-target object pair.

Algorithm 2 Known-item query generation algorithm.

```
1: procedure BUILD–QUERY(DATASET D)
2:                                                    ▷ Initialize query Q
3:     Q = ()
4:                          ▷ Randomly choose a target object O_i from the dataset D
5:     O_i = random(D)
6:                                                    ▷ Select d dimensions
7:     d = select_dimensions({what, who, when, where, how})
8:     for each d_i ∈ d do
9:           ▷ Randomly choose v values from target object O_i and dimension d_i
10:        v = select_values(d_i)
11:                                    ▷ Add dimension and values to the query Q
12:        Q(d_i) = v
13:    end for
14:    return Q
15: end procedure
```

By using the proposed known-item query generation technique, we are able to build a list of query-target object pairs. However, a learning-to-rank training set is composed not only by pairs of query-known document, but also by a list of matching documents per query. In [22], the authors use classic unsupervised information retrieval models, such as BM25, as a weak supervision signal for training deep neural ranking models. In a similar fashion, we adopt an unsupervised ranking model, field-based BM25, to retrieve matching objects to a given query. In Section 5.3, we explained how the data retrieved for a user is unified and parsed according with the six dimensions and then, the parsed data is exported to Solr where it can be searched using a field-based BM25 approach. Given a query generated by Algorithm 2, a call to Solr will retrieve a list of matching documents to this query — the list is ranked using field-based BM25. Now, for each query, we have a list of matching documents that includes the (generated) target object and its corresponding feature vector as described in Section 5.2. Since the target

object is known for this query, a relevance label of 1 is assigned to it; otherwise, the relevance label will be 0.

In both our scoring methodologies, presented in Chapter 4 and Chapter 5, queries are considered independently, without taking into consideration the possibility of search sessions composed by multiple queries that are somehow related. Search sessions would be an interesting extension to our work; however, it would require new scoring techniques that can take into consideration previous results while scoring new matching objects [4, 34]. Another point to be considered in the future are relaxation rules. Relaxation rules are important since the human memory is prone to mistakes, leading to a considerable number of inaccurate queries during the search process.

In Chapter 7 we evaluate the efficacy of the proposed learned ranking model by comparing its performance with two popular scoring methodologies: BM25 and field-based BM25.

Chapter 6
Search Implementation

In this chapter, we discuss our search implementation from data extraction to entity recognition, and classification, that will be used as the basis of our experimental evaluation.

6.1 Data Extraction

There is a dearth of synthetic data sets and benchmarks to evaluate search over personal data. This challenge has only been exacerbated by the recent explosion in the amount of personal digital traces, as well as the varied services that create, collect, and store them. A data extraction tool that accesses a variety of available services retrieving and storing users' data is a significant step towards the development of an individualized context-aware personal information search tool. This section describes the status of our personal information extraction tool and three different datasets retrieved using the tool. Besides the personal digital traces datasets, we present a public email dataset, called Enron.

6.1.1 Challenges

Creating a unified personal information search tool is not a trivial task. The first important step is the identification, retrieval, storage and modeling of all the data pertaining to a user. In this section, we will discuss the more relevant challenges

encountered in the process of retrieving and storing a user's digital life to create a personal database that is robust, reliable and secure.

Most of a user's personal data is fragmented across multiple sources. Even in the best scenario where a user has complete control of his own data stored only on personal devices, it is challenging to keep track of every single bit of data stored over time, and it is even harder to remember exactly in which device the data is stored. The fact that personal data may not even be controlled by the user, since it can be spread across multiple third-party services, adds an extra challenge to the process of identifying and retrieving data. Although some web services provide access to data through programmatic APIs, retrieving the data from the sources can be tricky. The access to the APIs varies for each service and they are constantly being updated. Many common services do not export such APIs and require access via web query forms or outdated screen scrapping methods to retrieve the data. The extraction tool that we are proposing identified and implemented access to a variety of data sources, retrieving the decentralized data and storing it in a single database.

The heterogeneity of data storage formats across different devices and services presents a second major challenge. One possibility for addressing this challenge is to pre-process the data before storing it. However, this task, besides being time consuming, is prone to mistakes that could lead to missing important data. Pre-processing the data also requires the extraction tool to include deep knowledge of each data format available; this is a difficult process, especially given the rapid rate of changes in the services sourcing the data. To avoid these problems, we store the data keeping their original format in a NoSQL database that is already optimized for semi-structured data. Our prototype uses MongoDB, a document-store system with a BSON encoding.

As the data is being retrieved, two new challenges arise: storage and privacy. In the last couple of years, the impressive growth in storage space while keeping costs low guarantees that tools as the one we are proposing can be implemented while imposing very little additional cost to the user. In our implementation, the personal data retrieved is stored in the user's own hard drive. Even though this approach has some limitation in the sense that the data is only available locally, by storing it in the user's hard drive we can guarantee some clear privacy and security benefits. A more flexible approach would be to make the data available to the user from different devices and locations, as is the case with personal clouds; however, this approach would require careful handling of private data and support for user permissions. It is important to clarify that when we retrieve data from a user, most of the time this data contains information about interactions between the user and several other people. However, the data is still private since it only contains information that was already shared with the user.

Nowadays, there is a growing concern with how personal information is retrieved, stored and used. Different countries have been creating and updating existing privacy laws and regulations to reflect the currently reality in which personal data are being created every day in a very large scale. For instance, the European Union (EU) have introduced the General Data Protection Regulation (GDPR) **??** that presents new requirements for companies that collect and process any type of personal information. Similarly, in the United Stated of America, the California state has introduced the California Consumer Privacy Act (CCPA or CACPA) **??**. Washington state is trying to pass the Washington Privacy Act (WPA) **??**, in which the key elements are very similar to that of the CCPA. With time, personal information management tools will have to change to accommodate the new privacy laws and regulations.

6.1.2 Extraction Tool

In the process of creating a Personal Information Database (PID), our extraction tool [83] identified and implemented access to a variety of data sources. The underlying personal data retrieved is stored in a flexible format that will allow us to perform data integration, search, and knowledge discovery. In this section, we focus in describing all steps involved in the personal information extraction aspect of our project.

The Personal Information Database (PID) is responsible for storing not only the data retrieved by the extraction tool but also all the extra information needed by our search tool as topic modeling data and frequency scores. Besides that, the PID will hold all the information about a user necessary to access the APIs implemented – information such as access tokens are essential to authorize the tool to access data stored by third party applications.

The Extraction Tool retrieves data from a wide range of sources: social data (Facebook, Twitter, LinkedIn, Google+), geolocation data (Foursquare), personal files (DropBox), Email (Gmail), Calendar (Google Calendar), Contacts (Google Contacts), web search/browser history (Firefox), and financial data (Mint). The data is accessed through individual system APIs. The data collected includes content, structure and explicit and implicit context.

Using a clean user-friendly interface, a user can authenticate and authorize the extraction tool to access their personal data for the range of services being offered. When a user registers a service using the extraction tool, a request is sent to the service API that, after the user has authorized it, will reply with an access token that is unique to the user and allows our tool to freely access the data. The access token is stored in the PID with all relevant information pertaining to that user. From now on, every time a user call the extraction tool to retrieve

his personal data, the tool will query the PID for the access token and then will access the service API retrieving all the user new data since the last time a call was placed. It is important to highlight that a user's first attempt to retrieve data results in the tool trying to retrieve as much past data as possible as allowed by each API.

All implementation was done using Python with the Django framework. Services are authenticated and authorized using Oauth2 and the data is accessed through APIs provided by each service.

6.1.3 Personal Digital Data Traces

In this section, we briefly discuss the services integrated by our extraction tool together with a description of the available data.

The variety of personal data available to be retrieved is enormous and new sources of data are constantly appearing; based on that, the extraction tool was built to easily integrate new services with their own data schema. As a starting point, our effort was channeled to selectively retrieve data from popular services. Table 6.1 briefly describes the services and data retrieved by the tool.

The extraction tool retrieves and stores the data in BSON format using MongoDB. The data is not pre-processed in any way, i.e., the tool dumps the data preserving the original schema defined by the service from which it was retrieved. The absence of a unique pre-defined schema makes the tool robust to the very frequent changes in source APIs and export formats. Figure 6.1 illustrates a piece of data retrieved from Facebook. From this small piece of data we can extract information such as: time, user name, data type (Facebook album), album name and time the album was created and modified.

Information such as time, location, text and people are frequently found in

Table 6.1: Services and data retrieved

Data Source	
Dropbox	files, folders
Facebook	feed, photo, album, checkin, event, friend, family, group, inbox, link, note, post, status, home, profile
Foursquare	badge, checkin, friend, photo, recent
Google Calendar	metadata, events
Google Contacts	contact, groups
Google+	people, activities, comments
Gmail	inbox, sent
Linkedin	profile
Twitter	favorite, mention, friend, follower, timeline, retweet, msg received, msg sent, tweet
Firefox, Chrome	browser history, search history
Mint	financial data


```
{
   "_id" : ObjectId("11111111111111111"),
   "_cls" : "FacebookData",
   "facebook_user" : ObjectId("11111111111111111"),
   "idr" : "album:1111111111@facebook/albums#11111111111",
   "time" : ISODate("2013-11-25T19:22:31.989Z"),
   "data_type" : "ALBUM",
   "data" : {
      "count" : 1,
      "from" : {
         "name" : "Daniela Vianna",
         "id" : "11111111111"
      },
      "name" : "Picasa Photos",
      "privacy" : "friends",
      "cover_photo" : "1134709742204",
      "updated_time" : "2009-07-25T00:58:40+0000",
      "link" : "https://www.facebook.com/album.php?fbid=1&id=1&aid=1",
      "created_time" : "2009-07-25T00:55:43+0000",
      "can_upload" : false,
      "type" : "app",
      "id" : "111111111111"
   },
   "neemiuser" : ObjectId("528bc41199c7a058a85ab681")
}
```


Figure 6.1: Data retrieved from the Facebook album of a user

data from different sources. Besides those well know entities, the richness of the data and the possibilities that it offers in terms of how they are related and

how they can be used to support a more robust search approach present a great stimulus to the study and development of a more personal context-aware search tool.

The extraction tool retrieves text-based data and the metadata of multimedia objects. Table 6.2 shows three datasets along with the number and size of objects retrieved from different sources over different periods of time.

	Dataset 1		Dataset 2		Dataset 3	
Data Source	**#Objs**	**Size**	**#Objs**	**Size**	**#Objs**	**Size**
Facebook	1493	9Mb	2384	19Mb	3875	28Mb
Gmail	1136	107Mb	10926	1Gb	28318	3Gb
Dropbox	-	-	573	32Mb	573	32Mb
Foursquare	-	-	55	59Kb	55	59Kb
Twitter	-	-	2062	10Mb	3929	22Mb
Google Calendar	2	9Kb	209	389Kb	330	620Kb
Google+	1	1Kb	102	343Kb	110	367Kb
Google Contacts	157	158Kb	427	430Kb	525	629Kb
Firefox	-	-	-	-	412	415Kb
Mint	-	-	-	-	181921	63Mb
Total	2789	116Mb	16738	1.4Gb	219,993	3.6Gb

Table 6.2: Personal data sets

6.1.4 Enron

In Section 6.1.2 we have presented our Extraction Tool as a mean to retrieve personal data from a variety of data sources. Three different datasest were retrieved using the tool and detailed in Section 6.1.3. We have called this type of dataset as pertaining to the Personal Digital Data Traces application, since they are composed by a variety of digital data traces from different data sources. Another type of dataset studied in this dissertation, is an email dataset which contains a total of about 0.5M emails from 158 employees of the Enron Corporation. This data was originally made public, and posted to the web, by the Federal Energy Regulatory Commission during its investigation. The Enron dataset can be downloaded from

`https://www.cs.cmu.edu/~./enron/`. A python parser was created to translate each email into a JSON object that is then stored in MongoDB creating a Personal Information Database (PID) for the Enron dataset.

6.2 Classification

As discussed in Chapter 3, contextual information should be represented in a unified model that will allow to both represent the data in its context, and to search and navigate seamlessly through this contextual data. To this end, we proposed a six dimension model called, *w5h*. Having defined the *w5h* model, it is still necessary to find an effective solution to map the context information from each digital data trace into the six dimensions. In this section, we will introduce two different solutions: a static version of our *w5h* classifier, that requires human intervention, and a machine learning multi-class classifier that automatically maps the data from any data source into the *w5h* dimensions. This last version does not require human intervention.

6.2.1 Static w5h Classifier

Digital traces have their own structures but most are retrieved in a semi-structured data format (typically JSON through APIs), or are extracted along with some metadata. As a first step towards data classification, we implemented parsers to represent the raw data from each source in the *w5h* model, thus unifying the data downloaded into a single data collection. The identification of data according to the six dimensions is done by analyzing the data available to be retrieved for each data source implemented and then building a dictionary of words/labels for each *w5h* dimension. Much of the classification is intuitive, for instance, the words

From and *To* should be classified under the *who* dimension, while words *Subject* and *Body* should be classified as *what*. Text messages are classified as *what*, even though some specific information derived from content could be classified differently (e.g., "I went to the market today" gives both *when* ("today"), *where* ("market") and *who* ("I")). Note that the *how* and *why* dimensions are more ambiguous. For now, we consider *how* as the type of information recorded, e.g., a Facebook comment. The *why* dimension is not explored in this dissertation; it is derived from inference and can be used to connect events [51, 52].

Figure 6.2 shows an example of a Facebook comment classified according to the *w5h* model using the static set of parsers. By using the dictionary of words/labels built based on the data available to be retrieved for each service (e.g. Facebook, Enron, Gmail, etc.), the parser is able to map a Facebook "message" to the *what* dimension, the "from.id" and "from.name" to the *who* dimension, and the "created_time" to the *when* dimension.

The disadvantage of static parsers to map the raw data into the *w5h* model is that every time a new source of data is included in the data retrieval phase or an existing service changes the format of their data, the dictionary of words/labels and the set of parsers have to be updated. To avoid this problem, we proposed in Section 6.2.2 a machine learning multi-class classifier to translate raw data retrieved from third-party sources into the 6 dimension model without the need of human intervention.

6.2.2 Machine Learning w5h Multi-class Classifier

The problem of automatically mapping the personal data retrieved into the 6 dimensions model is the well know problem of multi-class classification, which means that there are multiple classes to be predicted, but each instance can be

```
{
_id: ObjectId()
source: “facebook”
feed_id: “xxxxxxxxx”
data: {
                what: [
                  {
                                key: “message”
                                value: “Personal Information Search and Discovery”
              }
                ]
                who: [
                  {
                                key: from.id
                                value: ”xxxxxxxx”
                  },
                  {
                                key: “from.name”
                                value: “John Smith”
                  }
                ]
                when: [
                  {
                                key: “created_time”
                                value: “2013-08-28 12:34:59+00:00”
                  }
                ]
                how: [
                  {
                                key: “data_source”
                                value: “facebook.status.comments”
                  }
                ]
                where: [ ]
                why: [ ]
}
data_type: “status.comments”
}
```

Figure 6.2: Example of a user Facebook comment parsed according to the *w5h* model.

assigned to only one class. In this section, we introduce our *w5h* multi-calss classifier, a simple deep learning classifier that translates personal data traces into the dimensional model proposed in this chapter.

A simple view of deep learning is that of chain of multiple layers of processing units in which the output of one layer work as an input for the next layer. The basic idea is that giving a learning model, the model's weight are constantly being adjusted in response to the error it produces. This cycle continues until the error

Arch.	Description
C1	LSTM, Dense(softmax)
C2	LSTM, Dropout(0.3), Dense(softmax)
C3	LSTM, Dropout(0.5), Dense(softmax)
C4	LSTM, Dense(relu), Dense(softmax)
C5	LSTM, Dropout(0.3), Dense(relu), Droupout(0.3), Dense(softmax)
C6	LSTM, Dropout(0.5), Dense(relu), Dropout(0.5), Dense(softmax)

Table 6.3: Machine learning multi-class classifier architectures.

cannot be reduced any longer. In our scenario, the input data to the *w5h* classifier is a set of sentences and labels. Labels are the *w5h* dimensions and sentences are each individual information in the user dataset. For instance, in Figure 3.1, Figure 3.2 and Figure 3.3, each line corresponds to a sentence/label pair. To be fed into a deep learning architecture, sentences are transformed in embedding vectors by a Word2vec algorithm[1] . Labels are reshaped into one-hot encoded binary matrices. Architectures were built combining LSTM (Long Short-Term Memory) [45] and Dense layers. Dropout [78], a regularization technique, was used in some architectures to reduce the complexity of the model with the goal to prevent overfitting. The simplification is done by randomly setting some of the dimensions of the input vector to zero. Table 6.3 describes each architecture evaluated. The x in Dropout(x) refers to the percentage of units (neurons) randomly deactivated in a layer.

Dense layers are classic fully connected neural network layers, i.e., each input node is connected to each output node. LSTM (Long Short-Term Memory) is a special kind of RNN (Recurrent Neural Network) capable of learning long-term dependencies. In our case, even though the network isn't recursive, the LSTM unit helps by adding another layer without causing explosion in the parameter space (weighted to be fitted). It is important to keep in mind, that in this work

[1] `https://radimrehurek.com/gensim/models/word2vec.html`

Architecture	Avg. Accuracy	Std.
C1	99.96%	+- 0.01%
C2	99.95%	+- 0.02%
C3	99.89%	+- 0.10%
C4	99.46%	+- 0.99%
C5	99.72%	+- 0.31%
C6	89.00%	+- 12.68%

Table 6.4: Average classification accuracy and standard deviation for each classification model.

we are exploring moderately large datasets, so adding extra layers would require more data to fit the parameters of the model.

The designing and configuring of deep learning models require a great amount of decisions that can be empirically evaluated. We adopted a 5-fold cross validation process to estimate the performance of models. We use categorical cross-entropy as the training criterion (loss function); Adam optimization algorithm as the optimization algorithm for our models; and, Accuracy as our evaluation metric. Table 6.4 shows the average classification accuracy and standard deviation for each architecture. The evaluation was conducted using the dataset *Dataset 2* described in Table 6.2. Architecture $C1$, with accuracy over 99.9%, is the most accurate with the lowest variation.

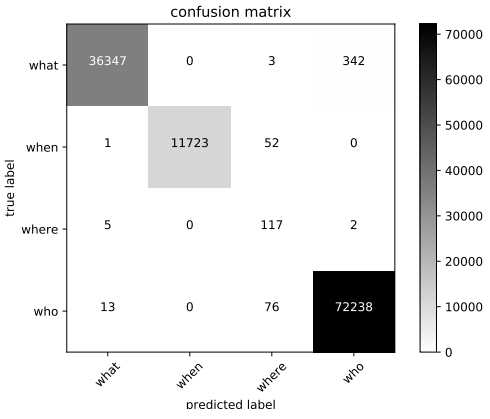


Figure 6.3: Confusion matrix with predictions for dataset *Dataset 1*. The model was trained using dataset *Dataset 2*.

The confusion matrix in Figure 6.3 shows the accuracy of the model for dataset *Dataset 1* (Table 6.2), using the training data from *Dataset 2* (Table 6.2), with the true labels represented in the y-axis and predicted labels in the x-axis. All correct predictions are located in the diagonal of the table. The results indicate that a machine learning classifier can accurately translate dynamic and heterogeneous set of personal data into the *w5h* model.

Our implementation uses the classifier to translate raw data into the *w5h* model and does not require user intervention.

6.3 Retrieval

The design of multidimensional indexing structures is the next step towards building a search tool. To support queries and scoring techniques for *w5h* search, we index our personal data using MongoDB single field indexes. Single field indexes are user-defined indexes on any single field or sub-field of a document. When the data is mapped by our classifier (Section 6.2), it is stored as a JSON document with fields for every *w5h* dimension. Then, a MongoDB single field index is created for each of the 5 dimensions: *who, when, where, why* and *how*. The exception is the *what* dimension. The *what* dimension in the *w5h* model is composed basically by content information comprising most of text (e.g., the body of an email, the body of a Facebook comment, or the text content of a Twitter tweet), so *w5h-f* uses a text-based score function to score this dimension and a MongoDB single field index is not needed.

When a query is submitted, each dimension is individually matched against the user's dataset using the above pre-computed indexes. Each separate search returns a list of objects that partially match the query for a given dimension, which are then scored using one of the *w5h* scoring methodologies.

6.4 Entity Resolution

In a personal dataset, the same person may appear in different services using variations of their names and email addresses. A very similar situation happens with location addresses. The format of addresses varies considerably between different services, and even within the same service. Most of the time users enter the address by hand, resulting in addresses with different spellings and incomplete information. In this section we describe the process we adopted to create entities for the *who* and *where* dimensions. The identification of common entities in different data sources allows for more accurate searching.

Our scoring technique relies on frequency scoring of the same entity across objects. To make this possible, we need to identify separate instances of the same entity in data traces coming from the same sources, and across sources. For instance, the same person may appear in different services using variations of their names and email addresses. A very similar situation happens with location addresses. The format of addresses varies considerably between different services, and even within the same service. Most of the time users enter the address by hand, resulting in addresses with different spellings and incomplete information. In this section we describe the process we adopted to create entities for the *who* and *where* dimensions. The identification of common entities in different data sources allows for more accurate searching.

Entity Resolution for the *who* dimension. Almost 100% of the personal data retrieved has information associated with the *who* dimension. Our goal is to identify unique entities (person) that may be referred to differently (e.g. different email addresses). The first step to solve the ER problem for the *who* dimension is to process the entire user dataset, and extract all information classified under *who*; for example, names and email addresses. We use the Stanford Entity Resolution

Framework (SERF), a generic open-source infrastructure for Entity Resolution (ER) [2], to identify entities. SERF uses the *swoosh* algorithm [11], proved to be optimal in the number of record comparisons in worst-case scenarios. Our decision to use SERF to build *who* and *where* entities was based on the fact that SERF is open source and addresses the entity resolution problem in an efficient and simple way. Using SERF person entities are identified and grouped in final entities that are stored in MongoDB in a separate collection.

Entity Resolution for the *where* dimension. Entity Resolution for the *where* dimension is the problem of identifying records in a database that refer to the same underlying location, and grouping them into a unique entity. As usual, ER is challenging since the same *where* location can be represented in multiple, ambiguous and error-prone ways. To disambiguate and match location data, we used Google Geocoding, Google Places API and SERF. We start by using Google Maps to disambiguate places that appear under different names and to augment the existing data. However, there are a number of challenges to be faced. In most scenarios, given an ambiguous location (e.g. *Student Center*), the Google Maps API outputs a set of results instead of a unique address, making it difficult to identify which one of the listed addresses is the target place. To overcome this issue, we rank all addresses returned by a Google Maps search using a *tf* (term frequency) function computed based on the user's dataset. For example, consider a set of results returned by the API search; the set of addresses includes an address in France; if the user's dataset does not have any data related to France, the address in France will be associated with a low *tf*. Similarly, when Google Maps API does not return any result for a given search, we augment the location search by using information from other related digital traces. We then use SERF for deduplication and record linkage for all the locations that have

the same geocoded address information or geographical coordinates (longitude, latitude).

Chapter 7
Evaluation

In this chapter we evaluate the efficacy of the *w5h-f* (Chapter 4) and *w5h-l2r* (Chapter 5) scoring approaches by comparing its performance with two popular existing search tools, Solr [6] (using different scoring methodologies: TFIDF, BM25, and field-based BM25), and Spotlight [1].

We start by discussing the accuracy of the search approach *w5h-f* for a set of search scenarios manually designed to be representative of possible user queries. Then, we explore the accuracy of the *w5h-f* approach using a much larger set of synthetically generated queries. Personal digital data traces datasets were used to validate the *w5h-f* approach. Finally, we use two different types of datasets, a personal digital data traces dataset and the Enron dataset, to validate the learning method (*w5h-l2r*) by comparing its performance with *BM25* and *field-based BM25* scoring methodologies.

7.1 Evaluation of the Frequency-based Scoring Approach: w5h-f

7.1.1 Dataset

The evaluation is performed using a real dataset collected by our extraction tool (Section 6.1.2) for one user (Dataset 2). Data is retrieved from current popular

services and sources of digital traces and stored in its original format in MongoDB, with the data from each service stored in its own collection. The dataset is presented in Table 7.1 along with the number and size of objects retrieved from different sources over different periods of time. This dataset will be used to evaluate the *w5h-f* scoring approach proposed in Chapter 4.

	Dataset 2	
Data Source	**#Objs**	**Size**
Facebook	2384	19Mb
Gmail	10926	1Gb
Dropbox	573	32Mb
Foursquare	55	59Kb
Twitter	2062	10Mb
Google Calendar	209	389Kb
Google+	102	343Kb
Google Contacts	427	430Kb
Total	16738	1.4Gb

Table 7.1: Personal dataset

Digital traces are mapped into the 6 dimensions using one of the classifiers introduced in Section 6.2. After classification, the data that was initially stored in individual collections will now be unified in one single MongoDB collection, named parsed collection. Table 7.2 shows the number of objects in the parsed collection for Dataset 2. During the classification an object can be separated in multiple objects. For instance, a Facebook post may contain multiple comments. In this case, the Facebook post and each individual comment will result in one individual object in the parsed collection. This explains the number of objects in the parsed collection (36381 objects) being greater than the total number of objects in the dataset before classification (16738).

	Dataset 2
	#Objs
Parsed Collection	36381

Table 7.2: Number of objects in the parsed collection for Dataset 2

Our scoring technique relies on frequency scoring of the same entity across objects. Table 7.3 presents the number of *who* and *where* entities for Dataset 2

	Dataset 2
	#Entities
Who	31849
Where	647

Table 7.3: Number of *who* and *where* entities for Dataset 2

7.1.2 Evaluation Techniques

Solr. Solr [6] is a popular open source full-text search platform from the Apache Lucene project. To integrate Solr and MongoDB we used a generic connection system called MongoConnector [67]. Essentially, this connection allows Solr to extract the content of each object in the MongoDB collection as text for indexing and subsequent searches. For the experiments in this chapter, for each unique dataset in MongoDB we create two different data collections in Solr. The first collection integrates all data (raw data) retrieved by the extraction tool, from each different data source, in an unified collection. This approach allows user to search for information across the entire set of retrieved digital traces, which is already a significant step forward from the current state-of-affair. However, this approach does not solve the problems caused by the heterogeneity of the original data — e.g., different data formats affecting search accuracy — and also does not take advantage of contextual information attached to the data. TFIDF and BM25 are the scoring methods used to evaluate Solr search over the integrated raw collection. The second data collection created on Solr contains the data classified according to the *w5h* model, addressing the heterogeneity of the data and taking into consideration the contextual information attached to the data. Then, we run BM25 Solr field-based search over the integrated parsed data.

Spotlight. We also compare our search approach to Spotlight, the desktop search platform in Apple's OS X. Spotlight allows users to search for files based on metadata [1]. As with Solr, this approach also works using the integrated raw (original) data. However, since Spotlight is a desktop search, we wrote an extraction program that stored each object in the evaluation dataset as an individual file in a machine running OS X Yosemite version 10.10.5. Besides storing the objects as files, the extraction program also parses the data to extract metadata that can be added to the files. The type of metadata extracted are authors (MDAuthors: from, creator, actor...), creation date (MDCreationDate: created-time, created-at...), content change date (MDChangeDate: modified, updated-time...), content creator (MDCreator: data source) and path of a file (MDFroms: dropbox file path). It is important to mention that Spotlight only ranks one item that it views as most relevant to a query. All other matching items are returned without ranking, typically organized by type of documents (e.g., email, pdf, etc.).

w5h-f Our proposed approach relies on the six memory cues (what, who, when, where, why and how) to guide search. The *w5h-f* approach uses the data parsed according to the *w5h* model. The correlation between users/entities and how they interact over time through different services, including the frequency users communicate, is used to rank objects, as described in Chapter 4. *w5h-f* uses entity resolution, as described in Section 6.4, to disambiguate/link entities from different sources (e.g. Facebook, Gmail, Twitter...) in the data set.

7.1.3 Case Studies

We begin our evaluation by studying four manually created search scenarios designed to be representative of realistic user searches targeting different personal

Search Approach	Query Description	Rank
Scenario 1 - target: a Google+ post about SIGIR 2013 posted by Ashley in 2013		
Spotlight	MDContent: SIGIR, MDAuthors: Ashley, MDCreationDate: 2013	2 - 14
(Solr) TFIDF	SIGIR, Ashley, 2013	11
(Solr) BM25	SIGIR, Ashley, 2013	12
(Solr) Field-based BM25	who:Ashley, what:SIGIR, when:2013	8
w5h-f	who:Ashley, what:SIGIR, when:2013	5
Scenario 2 - target: an email sent to Anna discussing ER solutions		
Spotlight	MDContent: Anna, MDContent: ER	2-5712
(Solr) TFIDF	Anna, ER	17
(Solr) BM25	Anna, ER	5
(Solr) Field-based BM25	who: Anna, what: ER	5
w5h-f	who: Anna, what: ER	1
Scenario 3 - target: a photo of a cat posted on Facebook by Katie in March 2012		
Spotlight	MDContent:photo, MDContent:cat, MDAuthors:Katie, MDCreationDate:2012-03	2-2964
(Solr) TFIDF	photo, cat, Katie, 2012-03	5468
(Solr) BM25	photo, cat, Katie, 2012-03	9106
(Solr) Field-based BM25	what:photo, what:cat, who:Katie, when:2012-03	65
w5h-f	what:photo, what:cat, who:Katie, when:2012-03	13
Scenario 4 - target: a Facebook photo of Anna taken in Campos		
Spotlight	MDContent:Photo, MDContent: Anna, MDContent: Campos	2-3169
(Solr) TFIDF	Photo, Anna, Campos	17
(Solr) BM25	Photo, Anna, Campos	43
(Solr) Field-based BM25	what: Photo, who: Anna, where: Campos	1
w5h-f	what: Photo, who: Anna, where: Campos	1

Table 7.4: Representative search scenarios targeting information stored in a user's personal dataset.

digital traces from the dataset *Dataset 2* described in Table 7.1. For each scenario, we compose one query for each of *Spotlight*, *Solr (TFIDF)*, *Solr (BM25)*, *Solr (Field-based BM25)* and *w5h-f* using the same information. Query conditions are derived from information in the target objects, and all conditions are classified accurately along the dimensions within *Spotlight*, *field-based Solr* and *w5h-f*.

Table 7.4 describes the search scenarios, the corresponding queries, and the rank of the target object as returned by each search method. Note that the target objects are always found, since the queries are accurate, and all three search tools currently return all matching objects. When Spotlight does not return the target item as the 1st ranked result, we report the ranking as the range from 2 to the total number of returned items.

The results show that *w5h-f* achieves the best accuracy by always ranking the target object higher than or equal to *Spotlight* and *Solr*. The differences can be significant (e.g., scenarios 1, 2, and 3), demonstrating that using memory cues to guide search can lead to improved search accuracy. We next discuss each of the search scenarios in more detail to show how differentiating between the dimensions, and using frequency information, helps to improve search accuracy.

In scenario 1, the user is searching for a data item containing information about the 2013 SIGIR Conference. The information was sent or posted by Ashley. In this scenario, identifying Ashley as *who* and 2013 as *when* allows *w5h-f* to rank the target object higher than all instances of *Solr*. When compared with *Solr* field-based BM25, using the same parsed data as *w5h-f*, the fact that *w5h-f* scoring function takes into consideration the frequency that Ashley communicated with the user during the year of 2013 using Google+, allows *w5h-f* to rank the target object higher than *Solr*. *Spotlight* was unable to leverage the same distinctions as

w5h-f since the target object was not ranked number 1. Thus, *Spotlight* returned the target object as an unranked item among 13 other items.

Scenario 2 searches for a message about ER solutions. The message was sent by (or mentions) the friend Anna. Again, *Spotlight* was not able to rank the target object and returned the item among 5711 other items. *Solr* BM25 and *Solr* field-based BM25, that represent state-of-the-art TFIDF-like scoring functions, scored the target object higher than *Solr* TFIDF. The fact that *Solr* BM25 runs over the raw data and *Solr* field-based BM25 runs over parsed data shows that the classification of the data into the 6 dimensions alone is not the reason why the *w5h-f* approach scores the target object higher than all other approaches. The *w5h-f* approach goes beyond the simple use of a person's name, as it can also rely on the entity resolution algorithm to identify common users across different data sources and group them in unique entities. Using an entity instead of a name allows us to eliminate unwanted results. For example, if we search for the name Anna, documents from Anna Smith and Anna Doe will be returned. However, searching for the entity Anna, where Anna is the entity id for Anna Smith, only the objects from the desired Anna (Smith) will be returned. The impact of entity resolution in this scenario is relevant considering that Anna is a very common name in the user data set. Using memory cues (dimensions), entity resolution, and a frequency-based scoring function, *w5h-f* was able to rank the target object 1st.

Scenario 3 targets a photo of a cat sent or taken by Katie in March 2012. In this case, the classification of photo and cat as *what* and Katie as *who* allows *w5h-f* and *Solr* field-based BM25 to rank the target object much higher than *Solr* BM25, *Solr* TFIDF and *Spotlight*. Entity resolution in the *who* dimension and the scoring function based on frequency help *w5h-f* to rank the target object in

the top 20.

Scenario 4 looks for a picture of Anna taken at a place called Campos. The good performance achieved by the *w5h* and *Solr* field-based BM25 approach is explained by the fact that those approaches were able to classify Anna under the dimension *who* and Campos under dimension *where*. Since Campos is a very common family name in the user database, the keyword search approaches ended up returning lots of documents matching Campos as location and also as a name.

7.1.4 Simulated Known-Item Queries

We now study a larger set of automatically generated known-item queries: search of personal data is usually focused on retrieving information that users know exists in their own data set. Because personal data trace search is a known-item type of search, simulated queries can be automatically generated using heuristics as the one described in Algorithm 2. Each call to Algorithm 2 will result in a query-target object pair.

For this set of experiments, we built a query set using *Dataset 2* (Table 7.1). The set comprises 5 different groups of queries, each containing 1500 queries for 250 different scenarios. Each scenario is automatically created by randomly choosing a target object from the user dataset. We then choose d dimensions, from which we randomly select v random values. We adapted the queries to each of our evaluation methods: *Solr* TFIDF, *Solr* BM25, *Solr* field-based BM25, and *w5h-f*. Table 7.5 shows the parameters (d, v) for the 5 query groups.

Including pertinent contextual information when searching personal data can significantly improve accuracy. Tables 7.7 and 7.6 show the MRR (Mean Reciprocal Rank), NDCG@10 (Normalized Discounted Cumulative Gain through position 10) and NDCG@20 (through position 20) of each approach, *Solr*

Parameter	Group 1	Group 2	Group 3	Group 4	Group 5
#scenarios	250	250	250	250	250
dimensions (d)	what	what, who	what, who when	what, who, when, how	what, who, when, how
#values (v)	1	1	1	1	2(who,what), 1(when,how)

Table 7.5: Parameters used to generate five groups of queries.

TFIDF, *Solr* BM25, *Solr* field-based BM25, and *w5h-f*, for Group $1-5$ of queries. If the target object has the same ranking as other matching objects, we report the median value of the range. Observe that all search implementations that use the data parsed according to the *w5h* model, *Solr* field-based BM25 and *w5h-f*, outperform the keyword-based approaches, *Solr* TFIDF and *Solr* BM25. These results show how valuable it is to use context (*w5h-f* and *Solr* field-based BM25) to find matching documents.

Methods	MRR	NDCG@10	NDCG@20
Solr TF.IDF	0.2920	0.3384	0.3673
Solr BM25	0.4742	0.5192	0.5352
Solr Field-based BM25	0.4979	0.5428	0.5619
w5h-f (no entity)	0.5632	0.5993	0.6136
w5h-f	0.6119	0.6414	0.6546

Table 7.6: MRR, NDCG@10, NDCG@20 for Group 2 of queries.

The use of a more elaborated approach to search text data can positively impact the final results obtained by the *w5h* approaches. As previously mentioned, the *what* dimension in the *w5h* model is composed basically by content information comprising most of the text. *w5h-f* uses *Solr* field-based BM25 to score the *what* dimension. The impact of the text search using *Solr* field-based BM25 versus *Solr* TFIDF and *Solr* BM25, can be seen in Table 7.7 (a), which presents MRR, NDCG@10 and NDCG@20 for Group 1 of queries (queries have only the *what* dimension). We can observe that *Solr* field-based

Methods	MRR	NDCG@10	NDCG@20
Solr TF.IDF	0.1959	0.2304	0.2513
Solr BM25	0.2127	0.2481	0.2702
Solr Field-based BM25	0.2383	0.2712	0.2996
w5h-f	0.2383	0.2712	0.2996

(a) Group 1

Methods	MRR	NDCG@10	NDCG@20
Solr TF.IDF	0.3580	0.4036	0.4234
Solr BM25	0.5267	0.5619	0.5777
Solr Field-based BM25	0.6117	0.6582	0.6772
w5h-f	0.7072	0.7488	0.7628

(b) Group 3

Methods	MRR	NDCG@10	NDCG@20
Solr TF.IDF	0.3328	0.3925	0.4179
Solr BM25	0.5357	0.5888	0.6036
Solr Field-based BM25	0.6327	0.6765	0.6951
w5h-f	0.7539	0.7931	0.8013

(c) Group 4

Methods	MRR	NDCG@10	NDCG@20
Solr TF.IDF	0.3772	0.4270	0.4569
Solr BM25	0.5345	0.5924	0.6152
Solr Field-based BM25	0.5769	0.6363	0.6510
w5h-f	0.6514	0.7014	0.7124

(d) Group 5

Table 7.7: MRR, NDCG@10, NDCG@20 for groups 1,3,4,and 5 (Group 2 is in Table 7.6). Compared against w5h-f all the results are statistically significant (Wilcoxon signed-rank test).

BM25 and *w5h-f* use a more efficient approach to search and score text data than *Solr* TFIDF and *Solr* BM25. Note that since Group 1 has only one textual dimension in the query, the *w5h-f* is equivalent to the underlying text-based scoring approach for the *what* dimension; field-based BM25 in our implementation. The results show that the adoption of a field-based text search for the *what* dimension leads to better results.

Being able to disambiguate/link people from different sources of data

can significantly improve the accuracy of search. To analyze the importance of the entity resolution phase presented in Section 6.4, we created a group of queries (Group 2) composed by values from the *who* and *what* dimensions. The results, for the dataset *Dataset 2*, are illustrated in Table 7.6, with *w5h-f* approach being superior when using entity resolution, compared with an implementation of *w5h-f* that does not use entity resolution.

Including frequency information as part of the scoring results in significant improvements. Tables 7.6 and 7.7 show that *w5h-f*, which uses our proposed frequency scoring (Chapter 4), consistently outperforms *Solr* field-based BM25, which also relies on the *w5h* model but does not consider frequency. This shows that taking into consideration the correlation between dimensions while scoring an object improves the search accuracy.

Our evaluation shows that using tailored frequency-based multidimensional scoring approaches yields significant improvements in search accuracy over personal digital traces where the desired search outcome is a specific known object.

7.2 Evaluation of the Frequency-based Learning-to-Rank Approach: w5h-l2r

7.2.1 Case Studies 1: Personal Digital Data Traces

Data Set.

We perform our evaluation using a real dataset collected by our extraction tool (Section 6.1.2) containing approximately two hundred thousand objects. Table 7.8 shows the composition of our real user dataset (Dataset 3), including the number and size of objects retrieved from different sources over different periods of

	Dataset 3	
Data Source	**#Objs**	**Size**
Facebook	3875	28Mb
Gmail	28318	3Gb
Dropbox	573	32Mb
Foursquare	55	59Kb
Twitter	3929	22Mb
Google Calendar	330	620Kb
Google+	110	367Kb
Google Contacts	525	629Kb
Bank	412	415Kb
Firefox	181921	63Mb
Total	219,993	3.6Gb

Table 7.8: Personal dataset.

time. The dataset was automatically classified according with the 6 contextual dimensions: *what, who, when, where, why* and *how*. We used this unified dataset to evaluate the frequency-based learning-to-rank approach proposed.

Training and Evaluation Query Sets.

We train and evaluate our model using heuristically generated samples. As detailed in Section 5.5.1, each query is automatically created by randomly choosing a target object from the evaluation dataset. We then choose d dimensions, from which we randomly select v random values. For this set of experiments, we built a training set comprised by 19000 queries over our personal dataset (Table 7.8). To built the query sets, we use $v = 1$ and 4 different values for parameter d: {*what, who*}, {*what, who, when*}, {*what, who, when, how*}, and {*what, who, how*}. The evaluation set was built in a similar fashion. Approximately 6000 queries were heuristically generated using the same combination of parameters as the training set. Since less than 2% of objects in the user dataset have location, the dimension *where* was not included in the query sets.

Evaluation Techniques and Metrics.

We evaluate the efficacy of the proposed approach by comparing it with two popular scoring methodologies: *BM25* and *field-based BM25*.

BM25 is a state-of-the-art type of TF-IDF function that ranks a list of matching documents based on the query content that appears in each document. To be able to use BM25 with the retrieved dataset (Section 7.2.1), the heterogeneous and decentralized digital traces have to be integrated in one unified collection. It is done by exporting the data retrieved to a unified data collection in Solr [6], a popular open source full-text search platform from the Apache Lucene project. This approach allows user to search for information across the entire set of retrieved digital traces, which is already a significant step forward from the current state, where users have to search each data source individually.

Field-based BM25 is a version of BM25 that takes into consideration the structure of a document. In our scenario, the fields in the field-based BM25 correspond to the five dimensions proposed: *what, who, when, where, how*. Before being exported to Solr, the retrieved dataset (Section 7.2.1) is unified and parsed according with the *w5h* model. It allows for the dataset to be searched using field-based BM25 with each field corresponding to a respective dimension. Note that by using the five dimensions, **we are giving the field-based BM25 approach the advantage of using our multidimensional data model to unify and organize the user data**.

The scoring model proposed is evaluated using 4 standard evaluation metrics: Mean Reciprocal Rank (MRR) of the top-ranked 50 documents, success (precision) of the top 1 retrieved document (success@1), success of the top 3 retrieved document (success@3), and success of the top 10 retrieved document

(success@10). Wilcoxon signed-rank test with $p\_value < 0.05$ is used to determine statistically significant differences.

Ranking Model.

To train and evaluate our model, we use the LambdaMART implementation provided by the RankLib library [69]. RankLib is a library of learning to rank algorithms that is part of The Lemur Project [81].

The first step in our evaluation was to define the best set of parameters that would give us a more robust and accurate model. With that in mind, we used a 5-fold cross validation process to estimate the performance of different models using LambdaMART. The parameters evaluated are: number of trees (tree); number of leaves for each tree (leaf); minimum leaf support (mls), minimum number of samples each leaf has to contain; and, training/evaluation metric (metric). In our evaluation we considered the following parameters:

- tree: 50, 100, 250 and 500
- leaf: 10, 15, 35 and 45
- mls: 10, 20 and 50
- shrinkage: 0.01, 0.03, 0.1, 0.3, 0.5, 1.0
- metric: MRR (Mean Reciprocal Rank)

After the validation process, we selected the model that shows the best performance on the training set, also taking into account the spread between training and testing metrics. The model selected, that we will call *w5h-l2r*, has the following parameters: number of trees = 50; number of leaves = 15; minimum leaf support = 10; shrinkage = 0.1.

Results.

In Table 7.9 we compare the ranking performance of the baseline (*BM25*), *field-based BM25* and learned ranking model (*w5h-l2r*) with respect to the entire evaluation set composed by approximately 6000 queries heuristically generated. The results show that both search models using the data parsed according to the multidimensional data model, *field-based BM25* and *w5h-l2r*, outperform the keyword-based approach, *BM25*, for MRR, success@1, success@3 and success@10. It shows that traditional keyword-based search methods are not appropriate in a setting where users may remember valuable contextual cues to guide the search. Observe that the learned ranking model, *w5h-l2r*, outperforms the *field-based BM25* approach for all 4 evaluation metrics, **showing that moderately large datasets can also benefit from learning-to-rank techniques when paired with a representative feature set built from our novel frequency-based feature space**.

Method	MRR	success@1	success@3	success@10
BM25	0.3629	0.2701	0.4104	0.5350
Field-based BM25	0.5082	0.4252	0.5502	0.6690
w5h-l2r	0.5184	0.4406	0.5601	0.6900

Table 7.9: MRR, success@1, success@3, success@10 for all 6000 queries (groups 1 to 4). Compared against the baseline (*BM25*), the results are statistically significant (Wilcoxon signed-rank test).

We now conduct a more thorough evaluation by dividing the evaluation set in four different groups by the dimensions in each query as described in Table 7.10.

Table 7.11a-d, show the MRR, success@1, success@3 and success@10 of each search approach, *BM25* (baseline), *field-based BM25*, and *w5h-l2r*, for Group 1 to 4 of queries. For all 4 groups, the search approaches that use the data classified according with our multidimensional data model are considerably more

Groups	Dimensions
Group 1	what, who
Group 2	what, who, when
Group 3	what, who, when, how
Group 4	what, who, how

Table 7.10: Dimensions used to generate four groups of queries.

Method	MRR	success@1	success@3	success@10
BM25	0.3435	0.2450	0.3939	0.5290
Field-based BM25	0.4407	0.3605	0.4809	0.6000
w5h-l2r	0.4432	0.3730	0.4851	0.6430

(a) Group 1

Method	MRR	success@1	success@3	success@10
BM25	0.3759	0.2875	0.4224	0.5370
Field-based BM25	0.5760	0.4850	0.6261	0.7420
w5h-l2r	0.5970	0.5084	0.6465	0.7660

(b) Group 2

Method	MRR	success@1	success@3	success@10
BM25	0.4213	0.3223	0.4614	0.5870
Field-based BM25	0.6168	0.5215	0.6417	0.7810
w5h-l2r	0.6331	0.5272	0.6618	0.7940

(c) Group 3

Method	MRR	success@1	success@3	success@10
BM25	0.3484	0.2591	0.3888	0.5200
Field-based BM25	0.4621	0.3863	0.4974	0.6150
w5h-l2r	0.4632	0.3964	0.4902	0.6000

(d) Group 4

Table 7.11: MRR, success@1, success@3, success@10 for groups 1,2,3, and 4

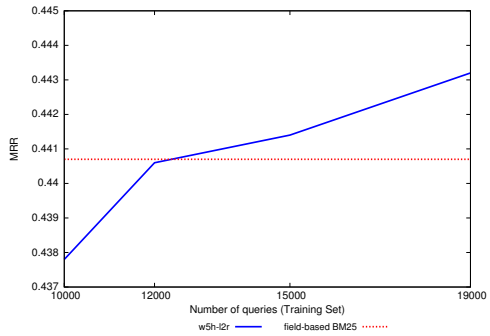
accurate than the keyword-based approach, *BM25*, confirming the importance of including contextual information to improve search accuracy when searching personal data. When compared against each other, *field-based BM25* and *w5h-l2r*, the learned ranking model outperforms the *field-based BM25* model for all four groups; however, the improvements were more relevant for Group 2 (*what, who,*

when) and Group 3 (*what, who, when, how*), showing that for this dataset, using the proposed learning model and training data, the *when* dimension and all related features played an important role in scoring query-document pairs. The results for *w5h-l2r* when compared with *field-based BM25* are statistically significant (Wilcoxon signed-rank test, $p\_value < 0.05$) for Groups 2 and 3, evaluation metric MRR and success@k. For Group 1 the results are not statistically significant for MRR and success@3. For Group 4, the results are not statistically significant for MRR, success@1 and success@3.

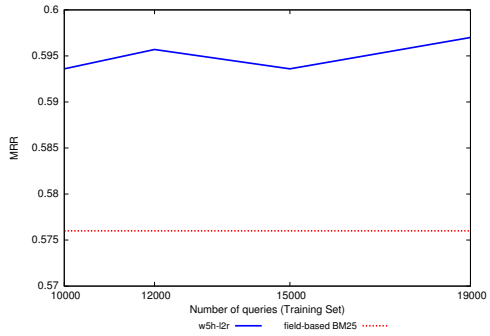
Figure 7.12 presents the performance (MRR) of the learning model, *w5h-l2r*, for Group 1 to 4 of queries as the number of training samples increases. We observe that the performance of the learned ranking model (*w5h-l2r*) clearly improves as the size of the training set increases just modestly, **showing the validity of our training set generation techniques**.

The importance of a feature in a gradient boosted decision tree model such as LambdaMART can be conveyed by the number of times such feature appears in the internal (non-leaf) nodes of the decision trees that form the tree ensemble. Since our model has 50 trees, each having 15 leaves (and 14 internal nodes), there are 700 branches overall. In Table 7.13 we present the feature frequency distributions for the trained *w5h-l2r* model. The most frequent feature in our model is the *what* dimension, that represents the content of an object and is scored using *field-based BM25*. Then, features (*who,when*), (*who,how*), and (*who*) appear next, all of them related to **the *who* dimension, which is expected since personal digital traces are byproduct of actions and events of users (*who*), and are typically focused on user interactions**.

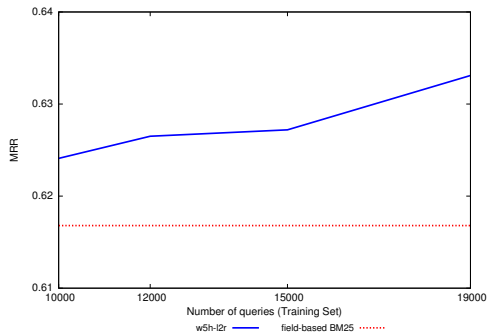
The results presented in this section indicates that personal data search can improve greatly by taking into consideration the knowledge the user has about



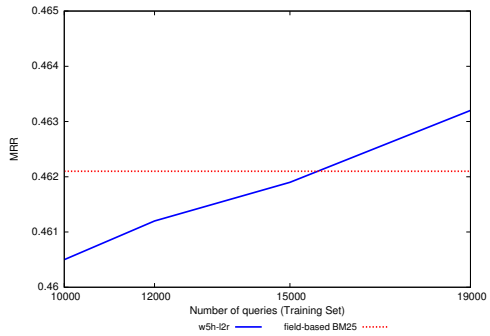
(a) Group 1



(b) Group 2



(c) Group 3



(d) Group 4

Table 7.12: Performance (MRR) of the learning model, *w5h-l2r*, for groups 1, 2, 3 and 4 of queries as the number of training samples increases.

Feature	Frequency
what	217
who, when	91
who, how	80
who	75
what, how	72
what, who, how	51
what, when	40
what, who	28
who, when, how	23
when	16
what, who, when, how	4
what, when, how	2
how	1

Table 7.13: Feature frequencies for the *w5h-l2r* model.

the object being searched. The multidimensional data model, based on the 6 contextual dimensions, proved to be an intuitive and efficient model to unify and link heterogeneous personal digital traces. The advantage of using a learning approach to re-rank search results can be seeing by the improvement presented by the *w5h-l2r* approach when compared against both methods, *BM25* and *field-based BM25*. Including a compact feature space based on frequency information resulted in significant improvements.

7.2.2 Case of Studies 2: Enron

Data Set.

To verify the validity of our frequency-based learning-to-rank approach over other domains, we have implemented it over an email dataset: the Enron [31] dataset (Section 6.1.4). The Enron email dataset contains a total of about 0.5M emails from 158 employees of the Enron Corporation, obtained by the Federal Energy Regulatory Commission after the company collapsed into bankruptcy resulting in

a federal investigation.

Training and Evaluation Query Sets.

To train and validate our model we use heuristically generated samples as we did in Section 7.2.1. Each query is automatically created by randomly choosing a target object from the evaluation data set. We then choose d dimensions, from which we randomly select v random values (Section 5.5.1). For this set of experiments, the training set is comprised by 48000 queries over the Enron dataset and the evaluation set is comprised by 2000 queries. Training and evaluation query sets were built using $v = 1$ and 4 different values for parameter d: {*what, who*}, {*what, who, when*}, {*what, who, when, how*}, and {*what, who, how*}.

Evaluation Techniques and Metrics.

To evaluate the efficacy of the proposed approach for the Enron dataset, we use the same metrics adopted to validate the Personal Digital Data Traces dataset (Section 7.2.1). The proposed approach is compared against *BM25* and *field-based BM25* and 4 standard evaluation metrics are used: Mean Reciprocal Rank (MRR) of the top-ranked 50 documents, success (precision) of the top 1 retrieved document (success@1), success of the top 3 retrieved document (success@3), and success of the top 10 retrieved document (success@10). Wilcoxon signed-rank test with $p\_value < 0.05$ is used to determine statistically significant differences.

Ranking Model.

To train and evaluate our model, we use the LambdaMART implementation provided by the RankLib library [69]. The model's parameters were defined using a 5-fold cross-validation process. For number of trees (tree), number of leaves

for each tree (leaf), minimum leaf support (mls), and training/evaluation metric (metric) we considered the same values as the ones used with the Personal Digital Data Traces dataset (Section 7.2.1):

- tree: 50, 100, 250 and 500
- leaf: 10, 15, 35 and 45
- mls: 10, 20 and 50
- shrinkage: 0.01, 0.03, 0.1, 0.3, 0.5, 1.0
- metric: MRR (Mean Reciprocal Rank)

We selected the model that shows the best performance on the training set, also taking into account the spread between training and testing metrics. The model selected, that we will call *w5h-l2r*, has the following parameters: number of trees = 50; number of leaves = 15; minimum leaf support = 20; shrinkage = = 0.3.

Results.

As with the Personal Digital Data Traces dataset, for the Enron dataset the evaluation set was divided in four different groups by the dimensions in each query as described in Table 7.10.

Table 7.14a-d, show the MRR, success@1, success@3 and success@10 of each search approach, *BM25* (baseline), *field-based BM25*, and *w5h-l2r*, for Group 1 to 4 of queries. For Group 1 (Table 7.14a), the search approach *w5h-l2r* is slight better than *BM25* (baseline) and *field-based BM25* for MRR and success@1. For Group 2 (Table 7.14b) and Group 3 (Table 7.14c), the search approaches that use

Method	MRR	success@1	success@3	success@10
BM25	0.2591	0.1320	0.1120	0.0502
Field-based BM25	0.2549	0.1260	0.1053	0.0510
w5h-l2r	0.2688	0.1560	0.1020	0.0504

(a) Group 1: what, who

Method	MRR	success@1	success@3	success@10
BM25	0.2346	0.1220	0.0980	0.0450
Field-based BM25	0.4139	0.2360	0.1767	0.0732
w5h-l2r	0.4218	0.2495	0.1790	0.0727

(b) Group 2: what, who, when

Method	MRR	success@1	success@3	success@10
BM25	0.2422	0.1328	0.0979	0.0449
Field-based BM25	0.4090	0.2314	0.1791	0.0736
w5h-l2r	0.4213	0.2575	0.1764	0.0744

(c) Group 3: what, who, when, how

Method	MRR	success@1	success@3	success@10
BM25	0.2442	0.1060	0.1087	0.0478
Field-based BM25	0.2585	0.1140	0.1133	0.0492
w5h-l2r	0.2477	0.1220	0.1020	0.0492

(d) Group 4: what, who, how

Table 7.14: MRR, success@1, success@3, success@10 for groups 1,2,3, and 4

the data classified according with our multidimensional data model are considerably more accurate than the keyword-based approach, *BM25*, and the results are statistically significant (Wilcoxon signed-rank test, $p\_value < 0.05$). For those groups, the learned ranking model, *w5h-l2r*, outperforms the *field-based BM25* model for all metrics, the exceptions being success@10 for Group 2 and success@3 for Group 3. For Group 4, all approaches had a similar performance. For most scenarios in this group of queries, the features based on the *how* dimension are not contributing to differentiate Enron results.

Table 7.15 shows the feature frequency distributions for the learned ranking model *w5h-l2r*. The two most frequent features in our model are based on the *what*

Feature	Frequency
what	273
what, how	118
who, when	97
who	77
what, who	56
what, when	25
what, when, who	19
when	18
what, when, who, how	15
how	2

Table 7.15: Feature frequencies for the *w5h-l2r* model and Enron dataset.

dimension, that represents the content of an object and is scored using field-based BM25. Then, features related to the *who* dimension appear next, representing the frequency of users (*who*) and the interactions between user/time (*who, when*) and user/topic (*who, what*).

The results discussed in this section show that even though the data and scoring model were proposed with the Personal Digital Data Traces dataset in mind, it can be extended to different domains with promising results.

Chapter 8
Concluding remarks

In this dissertation, we tackled the problem of searching personal digital data traces. We discussed the characteristics of personal data – usually small, heterogeneous, distributed across different sources –, and introduced a set of tools and techniques that allow users to easily access and search their own data on their own devices. As a first step, we presented a multi-dimensional data model, the *w5h* model, based on the six natural questions: *what, when, where, who, why and how*. The data model represents and integrates data across sources unifying their schemas and linking entities into a unified data set.

Based on the *w5h* data model, we designed two frequency-based scoring strategies for search: *w5h-f* and *w5h-l2r*. The scoring approaches leverage the correlation between users (*who*), time (*when*), location (*where*), data topics (*what*), and provenance (*how*) to improve search over personal data. The *w5h-f* scoring approach is a static function focused around personal digital traces and as such we included specific groups of correlations in our scoring. Other application scenarios could also benefit from our *w5h*, with other group and pairwise correlations highlighted in a dedicated frequency-based scoring. The second scoring model, *w5h-l2r*, is a learning-to-rank approach that expands the set of correlations from *w5h-f* to a set of 34 features to represent the input data (documents) for a query. The state of the art LambdaMART algorithm is used to map feature vectors to scores. Learning-to-rank approaches rely on human-labeled or clickthrough-based

training sets which are not available in our scenario; to overcome the lack of a publicly available training set, we proposed a combination of known-item query generation techniques and an unsupervised ranking model (*field-based BM25*) to generate query sets.

Tools for data extraction, classification, entity resolution, and topic modeling were implemented to validate the data model and scoring approaches proposed. Two different types of data were used for the evaluation: a publicly available email collection and personal digital data traces collections from real users.

Experiments over personal datasets composed by data from a variety of data sources showed that our *w5h-f* approach significantly improved search accuracy when compared with traditional search methods such as Apple's Spotlight and Apache's Solr, and techniques like TF-IDF, BM25, and field-based BM25. The results showed that search on personal data can be improved when the algorithms consider the context in which personal data traces are created, produced and gathered, and that including frequency information as part of the scoring function results in significant improvements. Also, being able to disambiguate/link people (entity resolution) from different sources of data can significantly improve the accuracy of search.

To evaluate our learning-to-rank approach, *w5h-l2r*, we used the Enron email collection and a personal digital data trace collection. The *w5h-l2r* performance was compared with *BM25* and *field-based BM25* scoring methodologies. The results showed that moderately large personal datasets can benefit from state-of-the-art learning techniques when combined with a compact frequency-based feature set.

In summary, we have showed that using contextual information to model and

integrate personal data can lead to more accurate scoring and searching methodologies. By introducing a compact feature set, we made it possible for moderately large datasets to take advantage of modern learning-to-rank techniques that are usually employed with very large datasets. Also, we have designed a known-item query generation approach that allowed us to generate query sets on the fly for a type of application (personal search) in which datasets and training sets are non existent; such approach is private by design, allowing for the end-to-end machine learning pipeline to run on the device of the user. In the future, the same techniques could be applied to different sets of applications, as we have done with the Enron dataset.

All the methodologies and techniques presented in this work to retrieve, integrate and search personal data pave the way to new research directions such as: preserving privacy while mining and searching personal data; integrating external information (e.g. weather) to improve personal search accuracy; using Natural Language Processing to improve query understanding considering the 6 dimensional data model proposed; applying query relaxation to personal data search with the intention to account for users fuzzy memories.

References

[1] *Spotlight.* `https://developer.apple.com/library/content/documentation/Carbon/Conceptual/MetadataIntro/MetadataIntro.html`.

[2] *Stanford entity resolution framework.* `http://infolab.stanford.edu/serf/`.

[3] G. D. Abowd, A. K. Dey, P. J. Brown, N. Davies, M. Smith, and P. Steggles, *Towards a better understanding of context and context-awareness*, in Proceedings of the 1st International Symposium on Handheld and Ubiquitous Computing, HUC '99, London, UK, UK, 1999, Springer-Verlag, pp. 304–307.

[4] E. Agichtein, E. Brill, S. Dumais, E. Brill, and S. Dumais, *Improving web search ranking by incorporating user behavior*, in Proceedings of SIGIR 2006, August 2006.

[5] S. Agrawal, S. Chaudhuri, and G. Das, *DBXplorer: A system for keyword-based search over relational databases.*, in Proceedings of the 2002 International Conference on Data Engineering (ICDE'02), 2002.

[6] *Apache solr.* `http://lucene.apache.org/solr/`.

[7] L. Azzopardi, M. de Rijke, and K. Balog, *Building simulated queries for known-item topics: An analysis using six european languages*, in Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '07, New York, NY, USA, 2007, ACM, pp. 455–462.

[8] M. Baldauf, S. Dustdar, and F. Rosenberg, *A survey on context-aware systems*, Int. J. Ad Hoc Ubiquitous Comput., 2 (2007), pp. 263–277.

[9] G. Bell and J. Gemmell, *Total Recall: How the E-Memory Revolution Will Change Everything*, Penguin, 2009.

[10] M. Bendersky, X. Wang, D. Metzler, and M. Najork, *Learning from user interactions in personal search via attribute parameterization*, in

Proceedings of the 10th ACM International Conference on Web Search and Data Mining (WSDM), 2017, pp. 791–800.

[11] O. Benjelloun, H. Garcia-Molina, D. Menestrina, Q. Su, S. E. Whang, and J. Widom, *Swoosh: a generic approach to entity resolution*, The VLDB Journal, 18 (2009), pp. 255–276.

[12] O. Bergman, R. Beyth-Marom, R. Nachmias, N. Gradovitch, and S. Whittaker, *Improved search engines and navigation preference in personal information management*, ACM Trans. Inf. Syst., 26 (2008), pp. 20:1–20:24.

[13] G. Bhalotia, A. Hulgeri, C. Nakhe, S. Chakrabarti, and S. Sudarshan, *Keyword searching and browsing in databases using BANKS.*, in Proceedings of the 2002 International Conference on Data Engineering (ICDE'02), 2002.

[14] L. Blunschi, J. peter Dittrich, O. R. Girard, S. Kirakos, K. Marcos, and A. V. Salles, *A dataspace odyssey: The imemex personal dataspace management system*, in In CIDR, 2007.

[15] C. Bolchini, C. A. Curino, E. Quintarelli, F. A. Schreiber, and L. Tanca, *A data-oriented survey of context models*, SIGMOD Rec., 36 (2007), pp. 19–26.

[16] C. M. Bowman, C. Dharap, M. Baruah, B. Camargo, and S. Potti, *A File System for Information Management*, in Proceedings of the Intl. Conference on Intelligent Information Management Systems (ISMM), 1994.

[17] W. Brewer, *Memory for randomly sampled autobiographical events*, Cambridge University Press, 1988, pp. 21 – 90.

[18] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender, *Learning to rank using gradient descent*, in Proceedings of the 22Nd International Conference on Machine Learning, ICML '05, ACM, 2005, pp. 89–96.

[19] C. J. Burges, *From ranknet to lambdarank to lambdamart: An overview*, tech. rep., June 2010.

[20] J. Chen, H. Guo, W. Wu, and C. Xie, *Search Your Memory! – An Associative Memory Based Desktop Search System*, in Proceedings of the 2009 ACM International Conference on Management of Data (SIGMOD'09)", 2009.

[21] S. Cohen, C. Domshlak, and N. Zwerdling, *On Ranking Techniques for Desktop Search*, ACM Transactions on Information Systems (TOIS), 26 (2008).

[22] M. Dehghani, H. Zamani, A. Severyn, J. Kamps, and W. B. Croft, *Neural ranking models with weak supervision*, in Proceedings of The 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2017.

[23] L. R. A. Derczynski, B. Yang, and C. S. Jensen, *Towards context-aware search and analysis on social media data*, in Proceedings of the 16th International Conference on Extending Database Technology, EDBT '13, New York, NY, USA, 2013, ACM, pp. 137–142.

[24] A. K. Dey, *Understanding and using context*, Personal Ubiquitous Comput., 5, pp. 4–7.

[25] *digi.me*. `https://www.digi.me`.

[26] J.-P. Dittrich and M. A. V. Salles, *iDM: A unified and versatile data model for personal dataspace management*, in Proceedings of the 32nd International Conference on Very Large Data Bases (VLDB'06), 2006.

[27] X. Dong and A. Halevy, *A platform for personal information management and integration*, in Proceedings of the Second Biennial Conference on Innovative Data Systems Research (CIDR'05), 2005.

[28] X. Dong, A. Halevy, E. Nemes, S. B. Sigurdsson, and P. Domingos, *Semex: Toward on-the-fly personal information integration*, in In Workshop on Information Integration on the Web (IIWEB, 2004.

[29] S. Dumais, E. Cutrell, J. J. Cadiz, G. Jancke, R. Sarin, and D. C. Robbins, *Stuff ive seen: A system for personal information retrieval and re-use*, in Proceedings of the 26th International ACM SIGIR Conference (SIGIR'03), 2003.

[30] D. Elsweiler and I. Ruthven, *Towards task-based personal information management evaluations*, in Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '07, New York, NY, USA, 2007, ACM, pp. 23–30.

[31] *Enron email dataset*. `http://www.cs.cmu.edu/~enron/`.

[32] R. Fagin, A. Lotem, and M. Naor, *Optimal aggregation algorithms for middleware*, Journal of Computer and System Sciences, 66 (2003).

[33] S. Fertig, E. Freeman, and D. Gelernter, *Lifestreams: An alternative to the desktop metaphor*, in Conference Companion on Human Factors in Computing Systems, CHI'96, 1996.

[34] S. Fox, K. Karnawat, M. Mydland, S. Dumais, and T. White, *Evaluating implicit measures to improve web search*, ACM Transactions on Information Systems, 23 (2005).

[35] J. H. Friedman, *Greedy function approximation: A gradient boosting machine*, Annals of Statistics, 29 (2000), pp. 1189–1232.

[36] J. Gemmell, G. Bell, and R. Lueder, *Mylifebits: a personal database for everything*, Communications of the ACM, 49 (2006), pp. 88–95.

[37] R. Goldman and J. Widom, *Dataguides: Enabling query formulation and optimization in semistructured databases*, in Proceedings of the 23rd International Conference on Very Large Databases (VLDB'97), 1997.

[38] J. Gwizdka and M. Chignell, *Personal Information Management*, in Jones and Teevan [49], 2007, ch. Individual Differences.

[39] K. Gyllstrom and C. A. N. Soules, *Seeing is retrieving: building information context from what the user sees*, in IUI, 2008, pp. 189–198.

[40] K. A. Gyllstrom, C. Soules, and A. Veitch, *Confluence: Enhancing Contextual Desktop Search*, in Proceedings of the 30th International ACM SIGIR Conference (SIGIR'07), 2007.

[41] J. Hailpern, N. Jitkoff, A. Warr, K. Karahalios, R. Sesek, and N. Shkrob, *Youpivot: improving recall with contextual search*, in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '11, New York, NY, USA, 2011, ACM, pp. 1521–1530.

[42] G. Halawi and A. Raviv, *Rank by time or by relevance?: Revisiting email search*, in Proceedings of CIKM 2015: 24th ACM CONFERENCE ON INFORMATION AND KNOWLEDGE MANAGEMENT, 2015.

[43] A. Halevy, M. Franklin, and D. Maier, *Principles of dataspace systems*, Communications of the ACM, (2006).

[44] H. He, H. Wang, J. Yang, and P. S. Yu, *BLINKS: ranked keyword searches on graphs*, in Proceedings of the 2007 ACM International Conference on Management of Data (SIGMOD'07), 2007.

[45] S. Hochreiter and J. Schmidhuber, *Long short-term memory*, Neural Comput., 9, pp. 1735–1780.

[46] V. Hristidis, L. Gravano, and Y. Papakonstantinou, *Efficient IR-style keyword search over relational databases.*, in Proceedings of the 29th International Conference on Very Large Databases (VLDB'03), 2003.

[47] V. Hristidis and Y. Papakonstantinou, *Discover: Keyword search in relational databases.*, in Proceedings of the 28th International Conference on Very Large Databases (VLDB'02), 2002.

[48] W. Jones, *Transforming technologies to manage our information : the future of personal information management. Part 2*, Synthesis lectures on information concepts, retrieval, and services ; no. 28, Morgan & Claypool Publishers, 2014.

[49] W. Jones and J. Teevan, eds., *Personal Information Management*, University of Washington Press, 2007.

[50] W. P. Jones, *The future of personal information management*, Synthesis lectures on information concepts, retrieval, and services, no. 21, Morgan & Claypool Publishers, 2012.

[51] V. Kalokyri, A. Borgida, A. Marian, and D. Vianna, *Integration and exploration of connected personal digital traces*, in Proceedings of the ExploreDB'17, Chicago, IL, USA, May 19, 2017, 2017, pp. 3:1–3:6.

[52] V. Kalokyri, A. Borgida, A. Marian, and D. Vianna, *Semantic modeling and inference with episodic organization for managing personal digital traces*, in Proceedings of the 16th International Conference on Ontologies, DataBases, and Applications of Semantics (ODBASE'17), Springer, 2017, pp. 273–280.

[53] J. Kamps, M. Marx, M. de Rijke, and B. Sigurbjornsson, *Structured queries in xml retrieval*, in Proceedings of the 14th ACM international Conference on Information and Knowledge Management (CIKM'05), 2005.

[54] D. R. Karger, *Personal Information Management*, in Jones and Teevan [49], 2007, ch. Unify Everything: It's All the same to Me.

[55] D. R. Karger, K. Bakshi, D. Huynh, D. Quan, and V. Sinha, *Haystack: A general-purpose information management tool for end users based on semistructured data*, in CIDR, 2005, pp. 13–26.

[56] C. S. Khoo, B. Luyt, C. Ee, J. Osman, H.-H. Lim, and S. Yong, *How Users Organize Electronic Files on Their Workstations in the Office Environment: A Preliminary Study of Personal Information Organization Behaviour*, Information Research, (2007).

[57] J. Kim and W. B. Croft, *Retrieval experiments using pseudo-desktop collections*, in Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM '09, New York, NY, USA, 2009, ACM, pp. 1297–1306.

[58] J. Kiseleva, *Using contextual information to understand searching and browsing behavior*, in Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '15, New York, NY, USA, 2015, ACM, pp. 1059–1059.

[59] J. Kiseleva and M. Pechenizkiy, *Context mining and integration into predictive web analytics*, 01 2013, pp. 383–387.

[60] C. Li, M. A. Soliman, K. C.-C. Chang, and I. F. Ilyas, *Ranksql: Supporting ranking queries in relational database management systems.*, in Proc. of the 31st International Conference on Very Large Databases (VLDB'05), 2005.

[61] H. linh Truong and S. Dustdar, *A survey on context-aware web service systems*, 2009.

[62] S. Liu, Q. Zou, and W. W. Chu, *Configurable indexing and ranking for xml information retrieval*, in Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'04), 2004.

[63] A. Marian, N. Bruno, and L. Gravano, *Evaluating top-k queries over web-accessible databases*, ACM Transactions on Database Systems, 29 (2004).

[64] A. K. McCallum, *Mallet: A machine learning for language toolkit.* http://mallet.cs.umass.edu, 2002.

[65] A. Moffat and J. Zobel, *Self-indexing inverted files for fast text retrieval*, ACM Transactions on Information Systems (TOIS), 14 (1996).

[66] N. U. Mohammed, T. H. Duong, and G. S. Jo, *Contextual information search based on ontological user profile*, in Computational Collective Intelligence. Technologies and Applications, J.-S. Pan, S.-M. Chen, and N. T. Nguyen, eds., Berlin, Heidelberg, 2010, Springer Berlin Heidelberg, pp. 490–500.

[67] *Mongo connector.* `https://github.com/10gen-labs/mongo-connector.git`.

[68] N. Polyzotis and M. Garofalakis, *Xcluster synopses for structured XML content*, in Proceedings of the 2006 International Conference on Data Engineering (ICDE'06), 2006.

[69] *Ranklib*. `http://www.lemurproject.org/ranklib.php`.

[70] S. E. Robertson and S. Walker, *Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval*, in In Proceedings of the 17th annual international ACM SIGIR conference, 1996, pp. 232–241.

[71] G. Salton and C. Buckley, *Term-weighting approaches in automatic text retrieval*, Inf. Process. Manage., 24 (1988), pp. 513–523.

[72] D. Schacter, *The seven sins of memory: How the mind forgets and remembers.*, Houghton Mifflin, 2001.

[73] A. J. Sellen and S. Whittaker, *Beyond total capture: a constructive critique of lifelogging*, Commun. ACM, 53 (2010).

[74] S. Shah, C. A. N. Soules, G. R. Ganger, and B. D. Noble, *Using provenance to aid in personal file search*, in 2007 USENIX Annual Technical Conference on Proceedings of the USENIX Annual Technical Conference, ATC'07, Berkeley, CA, USA, 2007, USENIX Association, pp. 13:1–13:14.

[75] 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, SIGIR '05, 2005.

[76] M. Smith, V. Barash, L. Getoor, and H. W. Lauw, *Leveraging social context for searching social media*, in Proceedings of the 2008 ACM workshop on Search in social media, SSM '08, 2008.

[77] C. A. N. Soules and G. R. Ganger, *Connections: using context to enhance file search*, in Proceedings of the twentieth ACM symposium on Operating systems principles, SOSP '05, New York, NY, USA, 2005, ACM, pp. 119–132.

[78] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, *Dropout: a simple way to prevent neural networks from overfitting.*, Journal of Machine Learning Research, 15 (2014), pp. 1929–1958.

[79] M. Steyvers and T. Griffiths, *Latent Semantic Analysis: A Road to Meaning*, Laurence Erlbaum, 2007, ch. Probabilistic topic models.

[80] J. Teevan, C. Alvarado, M. S. Ackerman, and D. R. Karger, *The perfect search engine is not enough: a study of orienteering behavior in directed search*, in CHI, 2004, pp. 415–422.

[81] *The lemur project*. `http://www.lemurproject.org`.

[82] D. Vianna, V. Kalokyri, A. Borgida, A. Marian, and T. Nguyen, *Searching heterogeneous personal digital traces*, in ASIST'19: Proceedings of the 82nd ASIS&T Annual Meeting, Melbourne, AU, 2019.

[83] D. Vianna, A.-M. Yong, C. Xia, A. Marian, and T. Nguyen, *A tool for personal data extraction*, in Proceedings of the 10th International Workshop on Information Integration on the Web (IIWeb), 2014, pp. 80–83.

[84] W. A. Wagenaar, *My memory: A study of autobiographical memory over six years*, Cognitive Psychology, 18 (1986), pp. 225 – 252.

[85] X. Wang, M. Bendersky, D. Metzler, and M. Najork, *Learning to rank with selection bias in personal search*, in Proc. of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2016, pp. 115–124.

[86] D. Xin, C. Chen, and J. Han, *Towards robust indexing for ranked queries*, in Proceedings of the 32nd International Conference on Very Large Databases (VLDB'06), 2006.

[87] Z. Xu, M. Karlsson, C. Tang, and C. Karamanolis, *Towards a Semantic-Aware File Store*, in Proceedings of the Workshop on Hot Topics in Operating Systems (HotOS'03), 2003.

[88] H. Zamani, M. Bendersky, M. Zhang, and X. Wang, *Situational context for ranking in personal search*, in WWW, 2017.

[89] S. Zerr, E. Demidova, and S. Chernov, *deskweb2.0: Combining desktop and social search*, in Proc. of Desktop Search Workshop, In conjunction with the 33rd Annual International ACM SIGIR 2010, 23 July 2010, Geneva, Switzerland, 2010.