Finding Experts in a Corporate Environment

THESIS

submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE
TRACK INFORMATION ARCHITECTURE

by

Jasper Elbertus Godfried Oosterman
born in Amsterdam
Finding employees with a specific level of expertise, experts, is vital for knowledge sharing and information reuse. A company benefits when its employees have access to an efficient system for expert finding, because time spent searching is time lost. Many companies, however, do not have such a system. We propose a solution that can be used to automate the manual expert finding process of a company. Our solution builds upon existing Expert Finding techniques and requires only existing documents as input. We perform a case study at the company Tam Tam in which we implement and test our solution. We show that our solution performs at least as good as the current manual approach of Tam Tam. The performance of our solution in terms of MAP is 0.420. We motivate why our solution applied to another company would perform equally well. Finally, we investigate improvements on the used technique and propose an addition that in our case study increased the performance with 33%.
Preface

Before you lies the final product of my study performed at Delft University of Technology. A study I combined over the last eight years with social activities from study and student associations, part time jobs, friends and girlfriends. I have enjoyed all the things that Delft has to offer a student. And I enjoyed them again. And again. At some point I realized that there was nothing that stood between me and my graduation. At that point I started my Master’s Thesis. Now, October 2011, it is done.

I enjoyed both the practical side and the scientific side of my Thesis. I learned what scientific research and scientific writing is really about. I tried to be concise, sharp and to the point. If I succeeded is up to you, the reader.

I would like to thank my direct supervisors, Geert-Jan, who did not let me call him Professor or Mister Houben, and Bart Manuel from Tam Tam. For me you were an ideal combination. Geert-Jan helped and advised me from a scientific perspective and Bart did the same from a business perspective. Both amazed me with their sharp minds and even sharper comments and critique. Geert-Jan, thank you for your help in improving my Thesis and making it scientifically sound. Bart, thank you for your advice and critique and allowing me to use all data available within Tam Tam.

My thanks also goes out to Joost, Marieke, Michel and Ronald for proofreading my Thesis.

Finally I would like to thank the members of my graduation committee for taking the time and effort to read and criticize my work.

I wish all readers a pleasant reading and that you may someday think “I read that somewhere” and think back to my Thesis.

Jasper Elbertus Godfried Oosterman
Delft, the Netherlands
October 4, 2011
Preface

Contents

List of Figures

1 Introduction
  1.1 Background
    1.1.1 About Tam Tam
    1.1.2 About the Expert Finding research field
  1.2 Research objectives
    1.2.1 Approach
  1.3 Contributions
  1.4 Outline

2 Expert Finding Techniques
  2.1 Expert and expertise
  2.2 Related Work
    2.2.1 Knowledge Management
    2.2.2 Recommender Systems
  2.3 Expert Finding systems
  2.4 The Expert Finding research field
    2.4.1 Early research
    2.4.2 TREC Enterprise Track
    2.4.3 Current research
  2.5 Summary

3 Expert Finding at Tam Tam
  3.1 Examples of fulfilling information needs
  3.2 Documents infrastructure
    3.2.1 Storing documents
    3.2.2 Searching documents
    3.2.3 Recent developments
  3.3 Company structure
## List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>An example of associations between employees, documents</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>and topics.</td>
<td></td>
</tr>
<tr>
<td>6.1</td>
<td>The performance of the AGEwin document prior for various</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td>window sizes.</td>
<td></td>
</tr>
<tr>
<td>6.2</td>
<td>The performance of the AGEred document prior for various</td>
<td>43</td>
</tr>
<tr>
<td></td>
<td>$\delta$.</td>
<td></td>
</tr>
<tr>
<td>6.3</td>
<td>The performance of the AGEwinred document prior for various</td>
<td>44</td>
</tr>
<tr>
<td></td>
<td>$\delta$ and window sizes.</td>
<td></td>
</tr>
<tr>
<td>6.4</td>
<td>The performance of the AGEwinred document prior with $\delta$</td>
<td>44</td>
</tr>
<tr>
<td></td>
<td>= 100 for various window sizes.</td>
<td></td>
</tr>
<tr>
<td>6.5</td>
<td>The performance of the AGEwinredlin document prior for</td>
<td>45</td>
</tr>
<tr>
<td></td>
<td>various values of $\omega$.</td>
<td></td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

The greatest assets of (most) companies are their employees. Not only because no work would be done without employees but also because employees know things. Things about the company, about customers, about methods, about the infrastructure and more important, things about the products and topics they are working on. Senior employees at a company have gained experience and knowledge about the things they do. This expertise is very useful because they can share that expertise to prevent other employees from reinventing the wheel when solving problems. This saves time, which is good for the company. That is, if employees can find the colleague with the necessary expertise.

In small companies employees probably know all other employees and know (to some extent) what they are working on and what they have done in the past. In larger organizations this is not the case and finding an expert can take a lot of time. Time is also wasted by employees struggling to come up with a solution for the problem themselves. Having a system that indicates the right experts for a problem is a situation most companies desire but cannot achieve.

'Finding experts' is the process of determining (and then retrieving) who the experts are and who are not. We first noticed that there was a need for this process while exploring Delicious\(^1\). Here bookmarks are shared and the most interesting bookmarks are determined by the amount of persons sharing that bookmark. However, the contribution of each person’s bookmark, the person being domain expert or layman, is equal. The problem is how to determine that someone is an expert.

For Delicious the current approach is adequate although it could be improved by differentiating between experts and laymen. In corporate environments a waste of time equals a waste of money which enlarges the effect of the problem. Corporate environments can greatly benefit if they have an efficient way to find experts.

For our master’s thesis we decided to research this problem. In this document we present our research on expert finding in a corporate environment. Our research comprises of a case study on the company Tam Tam.

\(^1\)A social bookmarking site at [http://delicious.com](http://delicious.com)
1.1 Background

In this section we introduce the company Tam Tam and the Expert Finding research field. The current expert finding approach of Tam Tam is presented in detail in chapter 3. Additional information on Expert Finding and the research in this field is presented in chapter 2.

1.1.1 About Tam Tam

Tam Tam is an internet company that was founded in 1996 by Paul and Bart Manuel. Currently Tam Tam has offices in Rijswijk, Amsterdam and Utrecht and 113 FTE. They provide custom solutions to customers in various domains, including healthcare, energy, and government. Tam Tam serves customers by giving advice, creating designs, implementing solutions using state-of-the-art technology and by providing operational services. Tam Tam is perceived as one of the larger players in this market. In 2010 they ranked 10 on the Emerce 100 in the category full-service internet companies.2

Tam Tam currently does not have a system that enables employees to search for experts.

The author of this thesis worked as an all-round developer at Tam Tam since 2008.

1.1.2 About the Expert Finding research field

The Expert Finding (EF) research field, as a subfield of Information Retrieval, finds experts based on information in, and metadata from, documents. EF is about users, persons that use an EF system, and topics on which those users can be experts. Documents (primarily text documents) are used to link users to topics. In general an algorithm calculates a score for a user that indicates how well that user matches the topic. These scores are used by the EF system to determine which users are experts when a user searches for an expert on a topic. The research on EF is not about how knowledge can be stored or shared, or why or in what way documents should be stored. It is about retrieving information from documents and using that information to determine which users are experts.

1.2 Research objectives

Knowledge sharing between employees is important for a company. It reduces time spent searching for information or for solutions to problems. Finding these employees with the desired knowledge (the experts) is a difficult but necessary process for knowledge sharing. We believe that (especially larger) companies lack an efficient expert finding process and pose the following research question.

RQ: How can we automate the expert finding process of a corporation?

To answer this generic question research has to be done on the expert finding processes of a large number of different companies. We perform a case study on Tam Tam

2 http://www.emerce.nl/nieuws/emerce-100-2010
and reflect our findings for this case study on the generic research question. We pose the following research questions for this case study:

**RQ1**: Which Expert Finding techniques are available for corporate environments?
**RQ2**: What are the current processes for finding experts within Tam Tam?
**RQ3**: What is the goal of Tam Tam regarding Expert Finding?
**RQ4**: Which Expert Finding technique can be used to reach the goal of Tam Tam and how can this technique be implemented?

Having answered these research questions we are interested if we can improve the performance obtained after answering **RQ4** using properties specific for Tam Tam. We therefore pose the following research question:

**RQ5**: Can we improve on the performance obtained after answering **RQ4** using properties specific to the Tam Tam document collection?

### 1.2.1 Approach

Our approach for answering these research questions starts with a thorough analysis of the current situation. We analyse both the Expert Finding research field and the current expert finding approach of Tam Tam. The Expert Finding research field and the techniques usable for expert finding in corporations (**RQ1**) are presented in chapter 2. We analyse the current expert finding process at Tam Tam by observing common daily interaction between employees and by interviewing employees about expert finding. The combined results from the analysis and the interviews (**RQ2**) is presented in chapter 3.

The next step is to determine the problem of the current situation. We use the result from **RQ2** to find problems with the current process. We explore these problems and determine which problems need to be solved. The goal of Tam Tam concerning expert finding (**RQ3**) is to solve these problems and is presented in the second part of chapter 3.

To solve these problems we need to map a suited technique from the analysis for **RQ1** to the goal described in **RQ3**. We present this technique (**RQ4**) in chapter 4. We implement this technique within Tam Tam and perform an experiment that gives us the performance of the implemented technique. We present the experiment and the results from the experiment in chapter 5. We look in the literature for methods to improve the performance of the implemented technique. We adapt our implementation and perform the same experiment. The methods and the results from the second experiment (**RQ5**) are presented in chapter 6.

### 1.3 Contributions

We show for our case study that we are able to automate the manual expert finding process of a corporation. The performance of our implementation of the technique is similar to that described in the literature. We propose improvements on the technique and achieved an increase of the performance (MAP) of 33%.
1.4 Outline

Chapter 2 describes the research on expert finding and expert finding techniques. Next, in chapter 3 we describe the company Tam Tam and their problem concerning expert finding. In chapter 4 we describe our solution. Chapter 5 describes the setup and the results of our solution tested on Tam Tam. In chapter 6 we describe improvements for our solution and present the result for those improvements. We present our conclusion and future work in chapter 7.
In this chapter we describe techniques that can be used to find experts in a corporate environment.

In section 2.1 we define the related terms *expert* and *expertise*. Related work is described in section 2.2. Next we introduce Expert Finding systems in 2.3. The research on Expert Finding is described in section 2.4. We summarize the possible techniques in section 2.5.

### 2.1 Expert and expertise

The term *expert* comes from the Latin term *expertus* which means experienced or proven. Synonyms of *expert*, provided by the Oxford English Dictionary (OED)\(^1\) and the Dutch Van Dale dictionary \(^2\), are *authority* and *specialist*. The following definitions are taken from the OED and Van Dale (translated from Dutch) for these three terms:

- One who is expert or has gained skill from experience.
- One whose special knowledge or skill causes him to be regarded as an authority; a specialist.
- Authority, specialist, in particular someone who passes judgment or gives advice in disputed matters.
- Person who because of profession or study in particular is authorized to judge on an issue.
- Someone who is dedicated to a specific part of an area of knowledge, operation or technology and who has mastered that part.
- In general use, one who specially or exclusively studies one subject or one particular branch of a subject.

We see a distinction in these definitions between *when* someone is an expert, *how* one becomes an expert and *what* the actions of an expert are. A person is an expert

\(^1\) [http://oed.com/](http://oed.com/)
\(^2\) [http://vandale.nl](http://vandale.nl)
2.2 Related Work

Employees in a company have (varying levels of) expertise. In most companies the real asset is the combined expertise of all the employees. Employees can use the expertise of other employees to solve problems or to provide information. There are different research fields that research experts and expertise. In this section we present two research fields that are related to the Expert Finding field. First we describe the field of Knowledge Management. Being a social science, the Knowledge Management focusses mainly on the people and their interaction. Next we describe the field of Recommender Systems. The focus in this field is more on systems instead of people.
2.2.1 Knowledge Management

The field of Knowledge Management (KM) distinguishes between individual memory and organizational memory. Individuals learn and gain expertise and organizations try to leverage that expertise into the organizational memory. KM is about how knowledge can be effectively elicited from persons, stored, retrieved and shared between persons from a sociological perspective. The focus is on reuse of the (stored) knowledge to the benefit of the company.

Although much research is done on the social part of knowledge sharing, researchers in that field recognize that advanced computer storage technology and sophisticated retrieval techniques (...) can be effective tools in enhancing organizational memory [1]. The implementation of KM processes and systems supporting those processes, however, is not trivial and company dependent. When these processes and systems are in place employees need to follow these processes (probably by filling out forms or creating documents) to ensure the quality of the system. KM systems are used for storing actual knowledge but also for retrieving that knowledge. A search for knowledge on a topic results in general in documents containing that knowledge but can in addition result in a list of experts on that topic.

2.2.2 Recommender Systems

Recommender Systems (RSs) are systems that provide recommendations to users of the system. Such systems aggregate ratings given on or created for entities to determine these recommendations [25]. For example, Delicious is a RS that recommends websites to users. Users give ratings to a website by creating an online bookmark to that website. Google also recommends websites to users, but ratings are not given by users but created by Google itself based on the structure of the internet. RSs are used for many purposes: A few examples are recommendations for music, books or websites. RSs do not store knowledge but only store the ratings.

We use the practice of the company GuruScan as illustration of a RS that recommends experts. Their approach is that knowledge can only exist inside people and not inside documents. Knowledge is gained through experience and knowledge sharing. Prior experience, knowledge, personal background and traits are involved in creating knowledge. Therefore knowledge cannot, in their opinion, be written down. Their goal is to provide a system that brings an expert and one problemowner together to share knowledge. Users of the system are requested to enter the topics of their expertise and to rate their level of expertise for that topic. Next to that the user enters other users with a high level of expertise on those topics. These users are notified with the request to rate themselves on that topic and to indicate other experts. An (undisclosed) algorithm then calculates expert ratings for users. Given a query the system returns a list of users with the highest expertise ratings.

---

3 http://guruscan.nl
4 This information comes from an interview on 23-02-2011 between Jasper Oosterman and Bart Verheijen, founder of GuruScan, at the office of GuruScan in Nijmegen.
2.3 Expert Finding systems

EF systems are software implementations that can be used by users for finding experts. Users can query or search on those systems. We now clarify the interaction between a user and an EF system. It starts with a user who has a problem which needs to be solved or who is in need of specific knowledge. The problem or knowledge is related to a topic. The need to solve the problem or the need for specific knowledge is called the information need of that user. The user transforms the information need into a query which is provided to the EF System.

**Definition 3** (Query). A query is something that a user conveys to the computer in an attempt to communicate an information need [21].

The format of a query should follow the standards for the system it is conveyed to. For example, a query for a (MSSQL/MySQL) database should follow SQL syntax while for an RDF system the query should follow SPARQL syntax. The common query syntax for EF systems is a combination of one or multiple query terms. A query term is a single word although in some system multiple words can be grouped by surrounding them with double quotation marks. Operators like AND and OR are not frequently used.

The result of a query is a list of users that the EF system determined to be experts. Based on the application the amount of experts can vary. Also some applications order the list of experts based on the level of expertise or (for probabilistic models) based on the confidence the system has that a person is an expert.

2.4 The Expert Finding research field

In this section we describe the research done in the field of Expert Finding (EF). We provide a historic overview of the field and describe the techniques that were described in published research. Most of the research describe a model that can be categorized as a candidate model, a document model or a combination of both.

1. The candidate model is based on creating a profile for each candidate expert (a user or employee) describing their expertise. Experts are found by finding profiles that best match a query.

2. The document model is based on retrieving documents which best describe the topic of a query. Experts are found by finding the candidate experts associated with those documents.

These models were formalized by Balog et al. and are extensively described in [2,3,5].

2.4.1 Early research

The research field of Expert Finding (EF) started around 1998 with the field study of McDonald and Ackerman [22]. They identified three behaviors for expert finding. The first, expertise identification, deals with the question: “Who are the experts on topic X?” The second, expertise selection, deals with the question: “Which of the people with the required expertise should I choose?”. McDonald and Ackerman also
identified *escalation* which deals with learning from mistakes made in the selection of identification [23]. Most research following this field study focussed on the expertise identification question.

One of the first expert finding systems is P@noptic Expert [13]. It uses a candidate model and creates a profile for each user. All documents (they used intranet pages) containing (the name of) a user are placed in that user’s profile. A document mentioning multiple users is put in multiple profiles. The assumption is that experts are likely to be mentioned in combination with the topics they are experts in. Queries are performed and processed on the profiles to retrieve experts. An undisclosed metric is used to rank the users to retrieve the top 10. No evaluation was published but preliminary results seemed promising.

Mockus et al. show that experts can be found using a software change management system [24]. They use a candidate model and create profiles of users consisting of experience atoms that are created when a user changes or creates a file. The amount of experience atoms created is relative to the size of the change. A file in the software change management system is directly linked to a project and an experience atom is thus also linked to a project. Experience atoms contain, next to the project, metadata such as tool or technology used and language. Because of that metadata the system is able to find employees who are experts on a project, but also on a specific language or technology. The higher the amount of (filtered) experience atoms of a user, the more likely it is that the user is an expert for that query. A qualitative study on two test groups of 50 developers showed that the calculated top contributors were also assessed as the most knowledgeable employees for those projects.

### 2.4.2 TREC Enterprise Track

The research in the field of Expert Finding intensified with the start of the Enterprise Track during the Text RETrieval Conference (TREC) in 2005 [12]. The goal of the Enterprise Track in 2005 was to let participants conduct experiments on enterprise data. A number of tasks were stated and a dataset⁵ was given to provide a standard platform for evaluation. One of the tasks was the Expert Search task: *Given a topical query, find a list of W3C people who are experts in that topic area*. A list of users and training topics were provided. Evaluation was done on the results for 50 queries. In 2006 the same dataset was used but with different queries. In 2007 a new dataset⁶ was created that was also used in 2008. The collection did not contain a list of users, which made that an implicit task to perform. The Expert Search task was to return email addresses of up to 100 users for 77 queries. The Enterprise Track stopped after 2008. We describe approaches that were evaluated on the datasets of the Enterprise Tracks.

In 2005 the best performing team of the Enterprise track, team THUENT, used a candidate model and performed the following steps in their algorithm [15]:

1. Create a user profile for each user containing all appearances of that person (name(s), email-address(es), phone number(s) and other unique values);

2. Extract data from documents for users according to their user profile and the document type;

---

⁵The TREC Enterprise Track 2005 and 2006 dataset consisted of a crawl of *.w3.org domain

⁶The TREC Enterprise Track 2007 and 2008 dataset consisted of a crawl of the *.csiro.au domain
3. Combine these description into an expertise profile.

Different document types were treated in a distinct way to retrieve the most relevant data from each document. A language model\(^7\) was implemented to rank users based on a query and their profile.

The second best team in 2005, team MSRA, implemented a document model which consisted of a relevance and co-occurrence model \(^10\). The relevance model represented whether or not a document was relevant to a query and the co-occurrence model represented whether or not a query was associated with a user, given a document. Another technique used by this team was clustering-based re-ranking which means they created clusters of contexts and persons to utilize relations between persons to improve the performance. This technique was used in combination with the relevance and co-occurrence model in their best performing run\(^8\) on the TREC 2005 dataset.

Macdonald and Ounis \(^{19}\) approached the problem as a voting problem and used data fusion techniques on the TREC 2005 dataset. They used a candidate model and analyzed three document weighting models combined with 12 candidate relevance models. The results were that the best performing models integrated the most highly ranked documents of the profile (strong votes), and the number of retrieved documents from the profile (number of votes). They concluded that document ranking is important for the performance of expertise finding and that using fields to represent the document structure always leads to performance improvements.

The best team in 2008 combined the candidate and document model \(^8\). They researched and tested both the candidate and the document model and various combinations of these models. In previous research the assumption was that if a query term and a person occurred in a document this indicated a relation between the term and the person. In this research they experimented with a window size such that the term and the person should occur near each other to indicate a relation. They also experimented with query expansion which is expanding the query with terms strongly related to the original query terms. A third technique they used was to make use of prior information about the documents and the users. They used the length of the URL of a document as an indication of the prior relevance of that document. The prior relevance of a user was determined by the function of that user in the company. They concluded that the query expansion and the priors had limited success. They also concluded that both models captured different experts and that combining both models resulted in substantial improvements.

### 2.4.3 Current research

Balog et al. adapted the candidate model and the document model to include the structure of a document. Structures documents were split up in different parts (title, chapters or even paragraphs) and each part was assigned a weight \(^4\). Macdonald et al. researched the document–topic association and what effect increased accuracy had on the performance of EF systems \(^{20}\). Relationships between users and hierarchical relations were also researched by \(^{5}\)\(^{17}\)\(^{26}\) with promising results.

---

\(^7\)For more information about language models the reader is advised to read \(^{21}\).

\(^8\)In this context a run is considered the execution of an algorithm on a dataset.
We have seen that the interest in EF diminished after the Enterprise Tracks stopped. The Expert Search task was replaced by the Entity Search task within TREC. This task focussed more on the meaning of information in documents. Balog, one of the main researchers in EF, used the knowledge from EF and applied it to Entity Search [6].

2.5 Summary

Efficiently finding experts in an organization is vital. The research field of Expert Finding (EF) searches for ways to automatically find experts based on the information need of a user. Documents contain information about certain topics. Each document is also linked to employees, either because the employee is the author of that document or because the employee is mentioned in that document. Figure 2.1 shows an example of how documents are used to link employees to topics. Employee E1 is associated with three documents and all these documents are associated to topic T1. Document D4 is associated to two topics and two employees (E2 and E3) are associated to that document. We described different techniques that can determine the weight of these associations and that determine, for a query, which employee is an expert on the topic of the query.

Two main models are used in the described techniques: The candidate model and the document model. The candidate model is based on the creation of a profile for each user. A profile of a user consists of documents that are related to the user. Queries are performed on the profiles of users. The document model is based on finding documents related to the query and then finding users related to the relevant documents. Both models are used to estimate the probability that a person is an expert given a specific query.

Which EF technique is the best depends on the company and the application. In the next chapter we describe the company of our case study, Tam Tam, their current expert finding process and problems with that process. We then choose which EF technique can solve that problem.
Chapter 3

Expert Finding at Tam Tam

In the previous chapters we defined expertise and how it is related to experts. We have described techniques that can be used to automatically determine the expert for an information need. In this chapter we describe our case study Tam Tam and the approach taken by employees of Tam Tam to fulfill their information needs.

We start in section 3.1 with examples that illustrate how information needs can be fulfilled within Tam Tam. We show that these information needs can be fulfilled by information from both employees and documents. Next, in section 3.2, we describe the infrastructure that is used to store and to search for documents. We describe the company structure in 3.3 to illustrate how employees search for people. In section 3.4 we formalize the complete search process how employees fulfill their information need. We state the changes taking place at Tam Tam and how these changes influence the search process in section 3.5. We describe problems with the search process and identify that the main problem is related to Expert Finding. We conclude the chapter in section 3.6 and state the goal of Tam Tam regarding Expert Finding.

3.1 Examples of fulfilling information needs

We describe three examples to illustrate how employees fulfill their information need. These example illustrate the need for finding documents and for finding employees.

Example 1. Tam Tam has an issue tracking system in which employees are assigned calls (bugs, issues, problems, request etc.) that need to be solved. There are many different Content Management Systems (CMS) that are used by customers. It frequently occurs that an employee is assigned a call, related to the CMS of a customer, that he or she has not worked on before. There is some basic information (server location, credentials, etc.) available but that is in most situations not enough. The employee that is assigned the call thus searches for someone with the expertise on that customer for a quick start. We have seen two frequently used approaches: 1) ask the call-manager for a reference to someone who worked on that customer before or 2) find the employee(s) that built the system.

Example 2. An employee is assigned the task to update the Content Management System (CMS) for a specific customer. Most of the time the provided upgrade manual does not specify the possible side effects of the update. An employee first has to update
a test version of the CMS to determine possible side. The update provides the employee with experience on the update process of that CMS which that employee documents and stores it in a logical place. When another employee performs the update on the live CMS that document needs to be found. The employee searches on the most likely place, the intranet, for the name of the CMS and the word update to retrieve the document.

Example 3. Sometimes expertise is required that is not linked to a specific CMS, technology, function or division. The effect is that the list of employees possible having the expertise cannot be narrowed down. A company-wide request (for example an email to all employees) can be done to find employees that possess that expertise. One of these requests was the following email: I am searching for examples of unique webshops. Does someone have an idea? The sender got seven replies and sent a second email summarizing the results.

Examples 1 and 2 show that to fulfill their information needs employees search for documents as well as for employees. Example 3 shows that company-wide emails are sent to all employees to find employees who are able to provide specific information. A search action can be split up in two parts: Performing the search and retrieving the information. The search is to get the name and location of an employee or document. Information can then be retrieved by eliciting the information from the employee or by reading the document.

In the next section we describe in detail how employees store and search for documents. As stated before Tam Tam does not have an automated system in which employees can search for experts. We therefore describe in section 3.3 the structure of the company to show the possible approaches employees can take to search for people.

3.2 Documents infrastructure

In this section we describe the infrastructure of Tam Tam that is used to store documents and the possibilities to search for these documents. We distinguish between personal documents and documents related to projects. This is a general overview and is not only focussed on document that can be used for Expert Finding.

3.2.1 Storing documents

Tam Tam has two large portals. One for employees, the Employee Portal, and one for customers called the Customer Portal. Both portals are Microsoft SharePoint 2007 implementations. The portals in general consist of three elements; normal text pages, applications and item libraries. The text pages are HTML pages (mostly simple formatted text), the applications are .NET applications with custom functionality and item libraries contain a number of items with the same item type. Item types can be customized by employees who define the different fields of that type but there also exists a generic document library where any document (Microsoft Office documents, text documents, archives, executables, etc.) can be stored. Any page, application or library can be placed anywhere on the portal and nested structures are also possible.

Documents related to projects can be found on different locations. Documents related to a project can be stored in a document library on the Customer Portal. Doc-
documents related to the customer in general are stored in a document library on the Employee Portal. Besides the portals there is a network share, called the Project Share, that also is used to store documents related to projects. Dropbox is also used as a means to share documents with customers. To store (versioned) code files for projects employees use SourceGear Vault, Subversion and GIT.

Personal documents can also be stored in a number of places. First, each employee has his or her own laptop on which documents can be stored. Each employee also has a personal network folder on the company network. Some employees also use Microsoft SkyDrive or other cloud storage to store their personal documents. Tam Tam has its own Microsoft Exchange server and there is a very high limit to the size of mailboxes. Mailboxes are not only used to archive email, but also to archive documents.

### 3.2.2 Searching documents

The standard portal search is able to search in all text pages, all libraries (of any type) and (almost) within any document type. Custom Search Providers can also be added (for example Twitter search if that seems useful) to increase the amount of search locations. On both portals the search only finds items within that specific portal.

There is no special document search for the Project Share but since it is a network share the operating system sees it as a normal folder. All operating systems are able to search (some more effective than others) through folders but special search software, for example Agent Ransack, is also used. This also holds for Dropbox. Versioning systems in general do not have specialized search methods. Mostly they are able to search for file names within a specific repository.

Personal document are private by nature and other employees cannot search through these documents. Numerous methods exist for an employee to search through personal document including specialized software like Xobni.

### 3.2.3 Recent developments

There are initiatives taken to index documents from various sources to increase the effectiveness of a document search. A Microsoft SharePoint 2010 environment, called Findr, was implemented to index both portals and to use the improved search methods included in the 2010 version. Another project, called DevSearch, was started to index all code files and to provide a means to search through these files.

These developments are the first steps in making documents within Tam Tam accessible. In the future it is likely that there will be only one interface that enables employees to search for documents. To find documents that match a query, documents have to be indexed.

The indexing of documents in the current systems is very basic. The title and contents of textual documents are indexed and documents are returned for a query only if the title or contents contain the words in the query. This kind of indexing and search is likely to be adequate if employees already know which document they are looking for.

---

1. Microsoft Exchange is a application by Microsoft that has, among others, email and agenda functionality.
looking for. However, if specific information is needed but the existence of documents containing that information is not known, this will not be adequate. Adequate means that if such information exists it is found in an efficient way.

### 3.3 Company structure

The structure described in this section is based on the structure of Tam Tam at the end of 2010. Much has changed and these changes are described in section 3.5.

Tam Tam is situated in Rijswijk on the first and second floor of an office building. Each floor has, besides some meeting rooms, an open office space. Employees work at desks that are grouped together with three to five other employees. All desks are supposed to be flexible workplaces (dutch: flexplek) but employees usually work at the same desk. Each employee belongs to a unit and each unit has a unit manager. Board members of Tam Tam work among regular employees.

Most employees work in project teams. These teams can be formed from employees in different units, have usually fewer than 8 members and have a project manager.

The hierarchy of Tam Tam is very flat. Employees report to their project manager and to their unit manager. Unit managers report to the board. This flat hierarchy in combination with the open office space create a open atmosphere in which the bar to contact other employees is very low.

### 3.4 Search process in Tam Tam

We observed the daily routine of Tam Tam employees and observed how employees searched for information. We interviewed four employees with different functions (developer, interaction designer, process advisor and partner) to elicit how they searched for information and how they shared knowledge. From these observations, interviews and own experience (see section 1.1.1) we have identified different actions that employees take when they have a specific information need. The following list summarizes (in order) these actions. The list only consists of actions that are taken to reuse existing expertise, either in documents or from employees.

1. The employee searches his or her own memory for the expertise.

2. The employee knows the (possible) location of a document that contains the expertise.

3. The employee performs a global document search.

4. The employee knows which person has or persons (possibly) have the expertise.

5. The employee asks people nearby where the expertise can be found.

6. The employee asks an information broker (manager, senior, information mediator or expertise concierges) where the expertise can be found.

7. The employee performs a global people search.
Global document search is included in this list but one remark has to be made. The employees we have spoken all indicated that they never searched for a document unless they were certain that the document existed. Even in that case the employees rather asked other employees than search for the document themselves. They commented that the available search systems were very slow and that the results were not satisfying.

The first three actions can be taken by the searching employee without interrupting other employees. Using action 4 the employees with the desired expertise are contacted directly. With action 5 and 6 the searching employee first contacts employees for a referral and then contacts the referred employees. The interviewed employees indicated that most of the time within three referrals the expert was found. For action 7 it is unknown how many employees are interrupted.

In the interviews, besides questions about how employees search, we asked whether searches were successful and how much time the search took. We differentiated between the amount of time needed to perform the search and to retrieving the information. This is an important distinction because, for example for action 7 using a company-wide email, the search takes only a few minutes while retrieving the information (waiting for answers) can take up to a day. The interviewed employees indicated that finding employees with specific expertise (the experts) was in almost all cases successful. For some searches multiple referrals were made, and it also happened that an expert was unavailable at the time of the search, but in the end the expert could always be contacted. One exception is the global people search because it regularly happens that there are no responses for a company-wide email. The following list contains the time estimations for each search action provided by the interviewed employees and our own experience.

1. Determining if one has the knowledge itself or posseses information about the existence of a document that contains the knowledge is a matter of seconds.

2. Finding a document in local document storage or in a repository (with known location) takes up to 5 minutes.

3. A global document search can take up to 5 minutes. This depends on the search terms used, the process of refining the search terms and the number of queries that have to be performed before retrieving the desires result. After 5 minutes of unsuccessful searching employees resort to other actions.

4. Contacting a small group of candidates by email or chat or to walk up to them takes about 5 to 10 minutes. We noticed that Tam Tam employees always have their email and chat program open and respond quickly to personal messages.

5. Asking employees that work nearby takes a few minutes depending if those employees are directly available or want to finish a piece of work. Locating the references employee(s) takes (as in action 4) about 5 to 10 minutes.

6. To find and speak to an information broker takes around 10 to 15 minutes. These employees have in general more scheduled meetings and a busier agenda.

7. A global people search takes around 5 minutes to perform after which answers can be received. After one day it is not expected to receive any more anwers.
3.4 Search process in Tam Tam

After a document or an employee is found the information needs to be retrieved. The time necessary depends on the information need, the employee or document and is highly variable.

3.4.1 Generic nature of the search process

We described the search process, the effectiveness of the process and the time needed for the actions in the process for our case study Tam Tam. In this section we motivate that the Tam Tam search process is generic and applicable to other companies. The time estimates, however, are Tam Ta specific. We note that the search process where employees themselfes search for experts is a social process. The social motivation of the actions in the process, like why employees choose a specific action, is researched in the field of Knowledge Management and are outside the scope of this thesis.

In the search process the following assumptions have been made.

- **Assumption 1:** Documents created for knowledge sharing are digitally accessible for other employees.

- **Assumption 2:** There is at least one information broker and that person can be contacted.

- **Assumption 3:** Employees rather search locally than globally.

Assumption 1 is supported by the trend that companies in all sectors are digitizing. Almost every company has a website and some kind of intranet, portal or other corporate network. All these networks have in common (even the simplest Local Area Network) that documents can be shared. Exceptions are companies that deal with classified information that cannot be shared among unclassified employees.

Assumption 2 states that for each employee another employee should knows his or her expertise. Only that way an employee can be found using a chain of referrals. We believe that in each company it holds that an employee works in a team, has someone to report to or performs the task of information broker.

We motivate assumption 3 by the findings in the field study of McDonald [22]. He states that the first rule-of-thumb in organizations is to keep it local and to try relative peers or immediate supervisors. If that fails the next step is to cross departemental boundaries. If that also doesn’t lead to results the last resort is an expertise concierge. We argue that, although an expertise concierge is one of the later steps, the last resort is a global people search. Another difference is that McDonald explicitly mentions the departemental boundary where the process of Tam Tam does not. This can be explained by the fact that Tam Tam does not have departments and that the different units are not closed silos.

Although the focus within Tam Tam is more on searching experts that on searching documents we believe the different actions in their search process are representative for other similar (midsized) companies.

The time estimates for the actions, however, are not generic. The time estimated to find documents depends on the infrastructure of a company and how their documents are indexed. The time needed to find employees depends on the hierarchical structure of the company, on the office space (open, cubicles or seperate rooms) and the amount and location of offices and departments.
3.5 Changes in Tam Tam

The description of the structure of Tam Tam in section 3.3 was about the situation of Tam Tam at the end of 2010. There have been a lot of changes that affected the search process. Tam Tam has been growing (in both turnover and number of employees) since its start in 1996 and have moved past the point where all employees could fit into one office. Currently (2011) there are three offices of Tam Tam: the main office in Rijswijk and offices in Utrecht and Amsterdam. Where in the old situation all employees knew each other this will not be the case in the new situation. In the new situation employees lunch at different locations and time. Also a number of social gatherings, like the friday afternoon drinks, are now held simultaneously on multiple locations. A board member of Tam Tam described this as the crossing of the 100 FTE boundary. Problems that might occur because of this crossing are similar for most companies that cross this boundary.

One consequence of employees being more dispersed is that expertise is more dispersed. Another good consequence is that there will be more expertise and that more documents will likely be produced. We identify that this causes a problem with document search and a problem with expert search. Both problems are related to scalability.

The opinion of the interviewed employees was that the current systems to search for documents are slow and the results not satisfying. These systems will not perform better if the number of documents increases. Another aspect is that project teams are more of less free to use the infrastructure they prefer. This way the cloud storage application Dropbox was introduced at Tam Tam. Although it is a continuation of a trend and not directly related to current situation, more employees and thus more project teams increases the chance that new storage locations are introduced. There is, however, no internal project that has the goal to create a system that is able to search within all storage locations nor one that (tries to) define a company-wide standard for document storage. The problem described here will likely increase the time needed for search action 3. In the case of Tam Tam, however, it will probably lead to a lower usage of global document search, because employees will resort to other actions to fulfill their information need.

The efficiency of expert search is likely to decrease. We believe that the number of other employees of which an employee knows the expertise is limited. With more employees the length of the chain of referrals will generally increase and possibly cross office locations. This puts more load on information brokers who then have to know the expertise of employees located in different offices. The effectiveness of expert search will likely not decrease based on the third assumption in the search process. This is only true if employees are willing to invest more time in finding the experts. The problem described here affects search action 5, 6 and 7. The times needed to perform such action are likely to increase. We believe this is the case for most companies that expand to geographically dispersed locations.
3.6 Conclusion

In this chapter we described the search process of Tam Tam. Information needs of employees can be fulfilled by either documents or employees. The search process thus consists of both document search and expert search.

Within Tam Tam there are many document storage locations and there are no guidelines that state which should be used. Not all files are indexed and there is no single system that can search through all of the different storage locations. An increased number of employees will likely increase the amount of documents created. This will, with the current systems, decrease the efficiency and effectiveness of document search.

Tam Tam also doesn’t have a system that enables employees to search for experts. Employees mostly search for experts via referrals from other employees. The increasing number of employees and the expansion to multiple locations complicate the search for expert employees. The length of the chain of referrals to fulfill an information need will in general increase. Information brokers will more often be needed, especially when the expert search crosses the boundaries of an office location. This decreases the efficiency.

We believe these problems scale with the size of the company and, for a growing company, need to be solved. In this thesis we only focus on the expert search problem. A solution to the problem should meet the following requirements.

1. The solution should be scalable. The efficiency and effectiveness should not decrease when the company grows.

2. The performance of the solution should be equal or better than the performance of the current approach.

The goal of Tam Tam is to create an efficient and effective expert search process that scales with the growth of the company.

In the next chapter we describe how we can reach this goal using a technique from chapter [2].
Chapter 4

A solution for expert finding

This chapter describes our solution for the expert finding problem. The Candidate Model is described in detail in section 4.1 and we state why this model can be applied in our case study. We describe the choices a corporation has to make to implement this solution in section 4.2. Next, we describe how documents should be preprocessed to be used in our solution. We conclude the chapter by describing the approach we take investigate how we can improve this solution.

4.1 Determining expertise

Expertise is a subjective concept and is not tangible. Due to this nature the expertise of a person cannot be calculated. We are however able to estimate the expertise of a person. In chapter 2 we concluded that there are two models, the document model and the candidate model, that are currently used. Both model the probability of a candidate expert $ca$ being an expert given the topic of query $q$. This is equivalent to determining $p(ca|q)$. We assume that an employee with the highest probability to be expert on a topic also has the highest level of expertise on that topic. Under this assumption both models can be mapped to the problem statement of Tam Tam.

Both models were previously tested on the TREC Enterprise Track datasets and performed almost equally well. The candidate model seems more intuitive than the document model. The complexity of the software implementation also seems less than that of the document model. We choose the candidate model as model for our solution for Tam Tam.

4.1.1 The candidate model in detail

In this section we describe the candidate model by Krisztian Balog [2]. The probability $p(ca|q)$ is difficult to determine accurately since the query $q$ is generally very short. It is common to apply Bayes’ Theorem to this probability to get the probability $p(q|ca)$ which can be determined more accurately.

$$p(ca|q) = \frac{p(q|ca) \cdot p(ca)}{p(q)},$$

(4.1)

where $p(q|ca)$ is the probability of a query given a candidate and $p(ca)$ and $p(q)$ are priors beliefs that $ca$ and $q$ are relevant, respectively. The probability $p(ca)$ is the
4.1 Determining expertise

A solution for expert finding

likelihood that $ca$ is relevant for any query and is assumed uniform. The probability $p(q)$ is, given a query, equal for each candidate. These terms are thus removed which gives equation [4.2]

$$p(ca|q) \propto p(q|ca)$$

(4.2)

The probability $p(q|ca)$ can be seen as the likelihood that candidate $ca$ produces (writes about) the topic of query $q$. To model this probability we use the information from corporate documents.

The assumption is that documents related to a candidate are representative for the level of expertise of that candidate. For any topic the level of expertise of a candidate is a combination of the relevance of documents (for that topic) that are related to that candidate. We thus need strength of the association of a document and a query, and of a document and a candidate.

The document - query association

The association between a document and a query is modeled as $p(d|q)$. We again apply Bayes’ Theorem, leave out $p(q)$, and get

$$p(d|q) \propto p(q|d) \cdot p(d),$$

(4.3)

where $p(q|d)$ is the likelihood that $d$ produces $q$ and $p(d)$ the prior probability that $d$ is relevant. For now we ignore this prior probability but we explain document priors in [6.1]. We thus assume $p(d|q) \propto p(q|d)$.

A query can consist of multiple query terms (see section [2.4]). The unigram model is a simple but effective model to model the relation between the query and the query terms. The unigram model assumes independence of the query terms and is given by

$$p(q|d) = \prod_{t \in q} p(t|d),$$

(4.4)

where $p(t|d)$ is the probability that $t$ is produced by $d$. This probability can be seen as the likelihood that we retrieve $t$ when we randomly choose a term from $d$. It follows that we estimate $p(t|d)$ as

$$p(t|d) = \frac{tf_{t,d}}{length(d)},$$

(4.5)

where $tf_{t,d}$ is the raw term frequency of term $t$ in document $d$ and $length(d)$ the number of words in $d$. We use the number of words and not the byte length or character count because these are advantageous for longer terms.

The document - candidate association

The association between a document and a candidate, modeled as $p(d|ca)$, determines the likelihood that the content of a document $d$ is related to the candidate $ca$. Most research assumes that documents have no metadata and determine the association based on the occurrence of the candidate in the document. An occurrence of a candidate is the occurrence of any property (name, email, etc.) that identifies a candidate.

It is in a corporate environment likely that some metadata of documents, especially author information, is available. Metadata can be retrieved from documents or from
the CMS that contains the document. For example, the document libraries on the portals of Tam Tam store metadata (date, username etc.) when documents are created, edited or deleted. In this thesis we assume author information is available and use the following boolean model.

\[
p(d|ca) = \begin{cases} 
1 & \text{if } ca \text{ is (one of) the author(s) of } d, \\
0 & \text{otherwise.} 
\end{cases} \quad (4.6)
\]

### Combining associations to estimate expertise

We have described both the document-query and the document-candidate association. We combine them to determine \( p(t|ca) \)

\[
p(t|ca) = \sum_{d \in D} p(t|d) \cdot p(d|ca). \quad (4.7)
\]

To model \( p(q|ca) \) we combine equation 4.4 and 4.7 into

\[
p(q|ca) = \prod_{t \in q} \sum_{d \in D} p(t|d) \cdot p(d|ca). \quad (4.8)
\]

In this model a candidate can be assigned a probability of zero if a term from the query does not occur in the documents associated with that candidate. This is an unwanted situation and it is one of the reasons it is common to use a smoothing method. A smoothing method adds probability mass so that a non-zero probability is assigned even if a document does not contain a query term. In most smoothing methods there is a smoothing parameter that controls the amount of smoothing.

Balog proposed a Jelinek-Mercer smoothing [16] method with a smoothing parameter \( \lambda_{ca} \) based on the amount of candidate information available in the collection. The smoothing parameter \( \lambda_{ca} \) is given by

\[
\lambda_{ca} = \frac{\beta_{ca}}{\beta_{ca} + \sum_{d \in D_{ca}} \text{length}(d)}, \quad (4.9)
\]

where \( D_{ca} \) is the set of documents of which \( ca \) is the author and \( \text{length}(d) \) is the number of words in document \( d \) and \( \beta_{ca} \) defined as

\[
\beta_{ca} = \frac{\sum_{ca} |D_{ca}| \cdot \langle d \rangle}{|ca|}, \quad (4.10)
\]

where \( \langle d \rangle \) is the average document length and \( |ca| \) the number of candidates. When much is known about a candidate \( \lambda \) is small and more weight is assigned to the documents of the candidate. Otherwise more weight is given to the so-called background probability. This background probability is the probability that the topic occurs in a random document. We use this smoothing method in combination with equation 4.8 to get the following, more accurate, estimate

\[
p(q|ca) = \prod_{t \in q} \left\{ (1 - \lambda_{ca}) \left( \sum_{d \in D} p(t|d) \cdot p(d|ca) \right) + \lambda \cdot p(t) \right\}, \quad (4.11)
\]
where \( p(t) \) is the probability that \( t \) occurs given the document collection. The probability \( p(t) \) is the likelihood that we retrieve \( t \) when we pick a random word in the document collection and is estimated by

\[
p(t) = \frac{\sum_{d \in D} tf_id}{\sum_{d \in D} \text{length}(d)},
\]

(4.12)

where \( D \) is the collection of documents.

Equation 4.11 is the model for expert finding that we use and implement in this thesis.

### 4.1.2 Selecting experts

The model in equation 4.11 can be used to assign a value between 0 and 1 to each employee. This value represents the likelihood that an employee is an expert on a topic. We use this likelihood to order the retrieved employees. An employee with the highest value is most likely an expert on the topic. To select experts, we sort the employees in descending order based on the likelihood and return the top \( n \) as results.

Another option is to return all employees of which the likelihood is above a certain threshold. For evaluation purposes we choose to return all employees for a query. For an operational system we only return the top 5 employees for a query. We do not use a threshold because this creates unwanted behaviour. Thresholds filter out employees that are not likely to be experts. However, we want to return the most knowledgeable employees for a query, even if those employee are not “real” experts.

### 4.2 Company specific choices

In the previous section we described the model we will use to find experts. This is a generic model that can be applied to any organization. In this section we describe the choices an organization has to make before implementing our solution.

First a choice has to be made which persons should be included in the system and are candidate experts. Is the system used for all kinds of expertise (Marketing, managing, programming etc.) or only for experts with technical expertise? Should very busy persons, for example the board of the company, be included?

Next, a choice has to be made regarding documents. Which documents, or documents from which storage location, need to be included. Our solution works on words in documents, but metadata (which contains words) of non-textual documents can also be used. Should non-textual documents be included? Should documents not inside the corporation but related to candidates, like web logs, be included? In other words a choice has to be made which (type of) documents are included in the document collection of the organization. We explain in the next section the preprocessing that needs to be done on the chosen documents.

### 4.3 Preprocessing documents

There are many different document formats that each has its own extension. Plain text files (.txt) can be read directly but there are proprietary formats (like .pdf and .docx)
that cannot. Special software is needed to read documents with those types and to display the special style, layout or images that is also included in those documents. However, we only need the text (or textual metadata) from these document. We describe so-called filters that solve that problem and show how we format the documents for optimization.

4.3 Filters

Although (most) proprietary format are used to provide readers with an enhanced experience, vendors of these format acknowledge the need for plain text. Most vendors provide so-called filters for their format that allows programs to index the content. Adobe, vendor of the .pdf format, created the Adobe PDF iFilter¹ that allows indexing of documents with that format. This means that using the filter we can retrieve all text in the document in plain text format. Another vendor is Microsoft that created the Microsoft Office Filter Packs² that enables us to retrieve the contents of any Microsoft Office document in plain text.

Using these filters we are able to retrieve the plain text representation of (almost) any document.

4.3.2 Optimization of documents

Having each document in plain text does not make each document usable. Document written in markup languages (like HTML and XML) use markup tags that structure the information in a document. Some text between tags is also not used for human readers like the style tag that contains style information. The following steps remove the tags and the text not meant for human reader.

1. Remove all CDATA blocks (between <![CDATA[ and ]]>),
2. remove all script blocks (between <script> and </script>),
3. remove all style blocks (between <style> and </style>),
4. remove all HTML markup (all tags starting with < and ending with >),
5. decode all HTML entities such that &euml; becomes é,

We further reduce the size of the documents, without losing information by performing the following steps.

1. replace the new line symbols \r and \n by spaces,
2. replace tab characters with spaces,
3. collapse subsequent space characters.

4.4 Improving our solution

The solution described in the previous part of this chapter describes our standard approach to the problem. We use the Candidate Model that was described in Expert Finding research. The Candidate Model was tested on the TREC Enterprise Track datasets and had a good performance. We believe that this model will also perform good in our case study. We perform an experiment on our case study to measure the performance. We describe this experiment and the results in chapter 5.

We are, however, also interested in how we can increase the performance of this solution. Research on the Candidate Model showed that document priors can be used to improve the performance of this model and are easy to include. What document priors exactly are and what they add to our solution is described in chapter 6. We perform the same experiment again, but in addition use the document priors. We call this our extended solution. Chapter 6 also described the results of that experiment.
Chapter 5

Experiment

In the previous sections we described the manual expert finding process of Tam Tam, the problem of Tam Tam with that process and our solution to solve that problem. In this chapter we describe the setup for our experiments and the results for our experiment using the standard solution. This setup is the same for experiments using our standard solution (this chapter) and the extended solution (chapter 6).

In section 5.1 we state the choices (see section 4.2) made for Tam Tam. Based on these choices we describe in section 5.2 the setup of the experiments. Section 5.3 describes the performance metrics we use to measure the performance of our solution. The results for the experiment is presented in section 5.4. We present our observations of these results in section 5.5.

5.1 Tam Tam specific choices

Tam Tam is a full-service internet company. There is a wide range of functions within Tam Tam like programmers, advisors and designers. An expert on, for example a product, can be both the programmer who developed the product as the manager who was responsible for the product. The system should be able to search for any expert. All employees of Tam Tam are therefore considered candidate experts.

As we described in section 3.2 Tam Tam has a number of different document storage locations. We chose to use the document libraries on both portals and the Project Share network folder. These locations both contain documents related to projects. We only look at documents for which we can get a textual representation. We chose to use documents with the following extensions: .doc, .docx, .txt, .rtf, .ppt, .pptx, .odf, .odt, .odp, .htm, .html and .one.

Tam Tam also has an issue tracking system. Employees who solve these issues gain experience. Information in this system is thus very likely to be useful and is therefore included.

Research showed that email is also a good source for information about expertise [7,9]. We therefore also include emails from employees. We explain the privacy with email in the next section.

Most Tam Tam employees are also very active on the internet. We chose to include two types of online platforms, weblogs (blogs) and Twitter, on which employees of Tam Tam share their knowledge.
5.2 Setup of the experiment

In this section we describe the setup of our experiments. To perform and measure the effectiveness of the experiment in a standard way, we need a test collection consisting of three things [21]:

1. A collection of documents
2. A test suite of information needs (queries)
3. A set of relevance judgments

In section 5.2.1 we describe the Tam Tam collection of documents. We present the queries we used for the experiment in section 5.2.2. In section 5.2.3 we present our set of relevance judgments. Finally, in section 5.2.4 we describe the software implementation of our solution.

5.2.1 Tam Tam Document collection

In our solution we assumed that each document has an author. Since the system only needs find expert in the employees of Tam Tam we only included documents for which we 1) could find an author and 2) that author was a current employee (a candidate expert) of Tam Tam. To link documents to candidates we created a list of current employees.

We describe for each of the document sources how we gathered these documents and how we matched these documents against the list of candidate experts.

Table 5.1 summarized the properties of the Tam Tam document collection specified by document type.

<table>
<thead>
<tr>
<th>Document type</th>
<th># of documents</th>
<th>average words/document</th>
<th># of candidates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Issue</td>
<td>3694</td>
<td>1012</td>
<td>53</td>
</tr>
<tr>
<td>Issue note</td>
<td>4855</td>
<td>353</td>
<td>43</td>
</tr>
<tr>
<td>Issue comment</td>
<td>24348</td>
<td>293</td>
<td>85</td>
</tr>
<tr>
<td>Tweet</td>
<td>10245</td>
<td>84</td>
<td>71</td>
</tr>
<tr>
<td>SharePoint document</td>
<td>4419</td>
<td>9054</td>
<td>78</td>
</tr>
<tr>
<td>Blog post</td>
<td>1033</td>
<td>946</td>
<td>52</td>
</tr>
<tr>
<td>Email</td>
<td>58805</td>
<td>1396</td>
<td>114</td>
</tr>
<tr>
<td>Project document</td>
<td>19036</td>
<td>3409</td>
<td>64</td>
</tr>
</tbody>
</table>

Table 5.1: Properties of documents types in the document collection. The number (#) of candidates is the number of candidates who were the author of at least one document of that type.

Candidate experts

Tam Tam has a database table containing all the current employees. We gathered the records from all active employees and created a collection of 120 candidate experts. We gathered the following unique field for each employee; Tam Tam and Twitter ID,
login & display name and email addresses. Display names are a combination of first
and last name and are unique in this collection.

**Documents from the issuetracker**

Tam Tam hosts an issuetracker (see example[1]) which customers and employees use.
Within an issue employees can communicate through comments and notes. The notes
are not visible for the customer. We assume for simplicity that issues, comments and
notes are independent entities. We thus split issues in smaller parts and define three
document types: issues, issue comments and issue notes. The issuetracker used a
database that contained a table for each of these types. We retrieved the information
using SQL queries.

The issues, issue notes and issue comments all used references to a separate table
which contained the issue tracker users. These users were linked to the (unique) lo-
gin name which had the format CUSTOMER\<login> or TAMTAM\<login>, where
TAMTAM is the domain all employees were in. We matched the <login> part of each
user with the login name of the candidates. Issues had four different user references
(owner, creator, assigned and developer). The problem with the creator field is that
most issue are created by the employee managing the system. The owner field was
only added during a recent update previously created issues did not have a values. We
therefore linked an issue to the developer of that issue.

We retrieved 3694 issue documents, 4855 issue note documents and 24348 issue
comment documents.

**Twitter**

Twitter[1] is a microblogging service that allows persons to post messages of maximum
140 characters that are made publicly available. To retrieve the tweets from candidates
we needed their Twitter usernames. Using scripts, manual labor and a global email
requesting Twitter user names we gathered 77 Twitter usernames associated with 76
candidates. We used the C# library TweetSharp[2] to retrieve the last 200 tweets from
those accounts.

We only retrieved tweets from candidates and linked each tweet during retrieval to
the candidate. We retrieved 10245 tweets linked to 72 candidates. For four candidates
we were unable to receive tweets, since their accounts were set to private.

**Documents on SharePoint environment**

We retrieved the documents by querying Findr (see section[3.2.3]) using the build-in
Web services of SharePoint[3]. We first used the Search Web service to retrieve the
paths of all documents. We retrieved the binary data of all documents that had the
required extension.

The SharePoint environments are only accessible for users that are logged in. All
actions, including uploading a file, are marked with the login name of the logged

---

in user. We matched these login name with the login names of our candidates. We retrieved 4419 documents from both SharePoint environments.

**Blogs**

Employees of Tam Tam have the opportunity to create a blog on the SharePoint portal. Some employees have, next to or instead of that blog, other publicly available blogs. Each blog on the portal has the same URL that only differs in the login name of the user. Using the login names of the candidates we crawled all possible blog locations to see if that candidate created a blog. We used the build-in List Web service to query the portal for all posts from existing blogs. We used an existing list of public blogs created by candidates. We used the RSS feed of those blogs to retrieve the title and description of each blog post.

We only retrieved blog posts from blogs created by candidates. We linked each blog post to the candidate that created the blog. We gathered in total 1072 blog posts originating from 100 blogs belonging to 61 candidates.

**Email**

Tam Tam used a Microsoft Outlook Exchange Server to process incoming and outgoing email. We wanted to parse the emails currently in the email box of candidates. Since email can contain very sensitive information we did not want to extract emails without prior permission from the candidates in question. We let the board of Tam Tam recruit volunteers with the promise that their emails would never become public and only be processed automatically. Twenty four candidates volunteered and gave permission. For those candidates we extracted the emails in all folders including drafts, trash and all user created folders. The chat history for Office Communicator 2007 (a client for voice calls and chat) is by default saved into a folder in the email box. Saved chats on the default location were therefore also gathered and treated as an email.

We used the Exchange Web Services to recursively crawl all folders and files (emails) from the root of a candidates email folder. Based on the unique Message-ID we retrieved from the 24 mailboxes 89775 email messages. We did not gather the attachments of those messages.

We determined the author of an email based on header information. Emails have, in the header data, a name and an emailaddress for the sender and for each receiver. We first determined all emailaddress - candidate combinations based on the name field of all emails. We did this by matching the name field with a combination of first and last name and the display name of the candidates. Next we matched, for each email message, the emailaddress of the sender to emailaddresses associated with candidates. We only kept the emailmessages for which there was a match with a candidate. This resulted in 58805 email messages.

**Project Share**

We used the Windows Shell library (Shell32.dll) to crawl the folders and to gather metadata from the documents. On the share we found 718025 documents of which

---


5. The Message-ID is a value in the header of an email that is unique for each sent message.
80131 documents had the required extension.

We used the “Owner” and “Authors” fields of the metadata for author information. We used the “Owner” field if it was not empty and the “Authors” field otherwise. We encountered person references in the following ways; domain login names, login names without domain and full names which consisted of the the first and last name. We matched the login with the login names of the candidates and the full names with the combination of first and last name and with the display name of the candidates. All documents either had zero or one match with a candidate.

5.2.2 Queries

The second part of the test collection is the list of queries that represent the information need of users. We first determined which aspects of Tam Tam a candidate can have expertise on.

We found the following aspects covered a large part of Tam Tam:

- Customers of Tam Tam
- Domains that Tam Tam serves
- Technology used in Tam Tam
- Content Management Systems (CMS) used in Tam Tam

We explicitly include CMS because a majority of the solutions for customers are based on a CMS. To be representative our experiment should cover a large part of these aspects. Together with a senior employee of Tam Tam we created eight keyword queries for the four aspects:

- customer, ICM and “Royal Haskoning”
- domain, Zorg (dutch for Healthcare) and Overheid (dutch for Government)
- technology, Android and “Mobile Development”
- cms, Smartsite and SharePoint

5.2.3 Relevance judgments

For these 8 queries we created relevance judgements. A relevance judgement states the truth about the relevance of a employee for a query. In this case an employee is relevant if he or she is an expert on the topic of the query. If an employee is an expert, however, is very subjective and (partly) depends on the opinion of others. We therefore determine the relevance of a candidate based on the subjective opinions of a large number of employees.

We therefore created a questionnaire for all employees of Tam Tam. We asked for each query which employees, in their opinion, were experts. Employees could choose and select up to five different candidates for each query. Our intent was that employees also ordered their selection, but we became aware this was not understood by some of the employees. We therefore only counted whether a candidate was selected or not.
The questionnaire was sent to all 120 employees. A total of 60 employees responded who entered information for 445 queries. Not all employees answered all 8 queries. The 60 employees selected 1070 candidates in total. We counted the number of times an candidate was selected for each query. This number does not prove that a candidate is an expert on the topic of the query. It merely states how many employees (of a total of 60) believed that the candidate was an expert. We assume that employees selected candidates for a query because they though (or knew) that these candidates had a certain level of expertise. Given this assumption, if a large number of employees selected a candidate, then, based on our definition of expert, that candidate is an expert.

In our case study we defined 10 as the minimum number of employees who should select a candidate. We thus filter out all candidates that got less then 10 votes. Each of the remaining candidates is relevant (an expert) with the normalized number of votes as the level of relevance.

Table 5.2 lists the amount of experts for each query.

<table>
<thead>
<tr>
<th>Query</th>
<th># of experts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Android</td>
<td>2</td>
</tr>
<tr>
<td>ICM</td>
<td>4</td>
</tr>
<tr>
<td>Mobile Development</td>
<td>2</td>
</tr>
<tr>
<td>Overheid</td>
<td>2</td>
</tr>
<tr>
<td>Royal Haskoning</td>
<td>4</td>
</tr>
<tr>
<td>Sharepoint</td>
<td>6</td>
</tr>
<tr>
<td>Smartsite</td>
<td>5</td>
</tr>
<tr>
<td>Zorg</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 5.2: Number of experts per query

5.2.4 Software implementation

The first step in the software implementation was to preprocess all the documents (see section 4.3) to get the optimized textual representation of each document. All documents that were gathered were stored in binary format in an Microsoft SQL Server 2008 database together with the employee ID of the author. We created a C# program that retrieved the binary data from the database, used the IFilter library\(^6\) to get the textual representation of the documents, and stored the results back in the database.

Using SQL queries we determined the total number of words and the number of occurrences of each keyword query in each document. We copied these these numbers, in combination with the employee ID and the document type, into a Microsoft Excel document. This created a file of 20 MB but Excel was able to handle such amount of data.

We created Excel formulas that calculated \(\lambda_{ca}, p(t|d), p(d|ca)\) and \(p(t)\) as described in chapter 4. We combined these values in Excel according equation 4.11 and let Excel calculate the probability \(p(q|ca)\) for each employee and query.

To measure the performance of our solution we created Excel formulas for the P@5, P@10, AP and nDCG metrics. The next section discusses these metrics.

---

5.3 Performance metrics

The metrics in this section are used to measure the performance of retrieval systems. The result of executing a query on a retrieval system is called the result set. In our experiment the result set consists of an ordered list of employees. The employees are ordered by their \( p(q|ca) \) value. All employees that, for a query, are considered experts are relevant and have a level of relevance. Other employees are considered non-relevant for that query. We selected these metrics because we wanted to answer the following question:

- How does our solution perform compared to the current manual approach of Tam Tam?
- How does our solution perform compared to other Expert Finding techniques?
- How does our extended solution perform compared to our standard solution?

We were unable to calculate the performance of the current manual approach. However, employees indicated that in most cases up to three referrals were enough to find the expert. This means for our system that within the top 3 employees returned for a query at least one should be an expert. We answer the first question by calculating the percentage of queries for which this is true.

Most research on Expert Finding use precision, the variants of precision (P@5, P@10 and MAP) and recall. We will use (some of) these metrics to compare to other research. However, these metrics only take the relevance of candidates, and not the level of relevance, into account. The Discounted Cumulative Gain does use the level of relevance and sees increased adaption in the field [21]. We use this metric to compare our own experiments.

5.3.1 Precision and Recall

Precision and recall are two standard information retrieval metrics. Precision is the fraction of relevant employees in the result set. Recall is the fraction of relevant employees that are retrieved from the total number of relevant employees.

In our experiment, each employee is assigned a probability and is included in the result set. The recall score is therefore always 1, because all employees are returned, and the precision score always very low, because all non-relevant employees are also returned. Recall and precision are therefore for our experiment not suited as performance indicators.

\( P@n \) is the precision at a cutoff value of \( n \), thus using only the top \( n \) employees. In some cases the maximum number of relevant employees is lower then the cutoff value. We adapt the \( P@n \) metric such that in those cases the \( P@n \) value can still be 1. Let \( R(q,i) \) be a function to determine whether the employee at rank \( i \) is relevant for query \( q \) and let \( k(q) \) be the total number of relevant employees for query \( q \). Then,

\[
P@n(q) = \begin{cases} \frac{1}{n} \cdot \sum_{i=1}^{n} R(q,i) & \text{if } k(q) \geq n, \\ \frac{1}{k(q)} \cdot \sum_{i=1}^{k} R(q,i) & \text{otherwise}, \end{cases}
\]
where \( n \) is the amount of retrieved employees. For example if only 4 experts exist for a query, and these experts are all retrieved within the top 10, the value for \( P@10 \) is 1.

The AP is determined by calculating the precision at the rank of each relevant employee. Let \( c \) be the total number of relevant employees, \( R(q, i) \) a function to determine whether the employee at rank \( i \) is relevant for query \( q \) and \( P@i \) the precision at rank \( i \). Then,

\[
AP(q) = \frac{\sum_{i=1}^{n} P@i \cdot R(q, i)}{c},
\]

(5.2)

where \( n \) is the amount of retrieved employees.

Both \( P@n \) and AP are used to determine a performance score for one query. The performance of a set of queries is calculated by taking the mean of the \( P@n \) or the AP scores. The mean of the AP scores is called the Mean Average Precision (MAP). Both metrics are used to state the performance of a system, based on a set of test queries, in a single number.

### 5.3.2 Normalized Discounted Cumulative Gain

The Discounted Cumulative Gain (DCG) takes the level of relevance into account. A high DCG is achieved when all relevant employees rank high and are ordered according their level of relevance. Let \( R(q, i) \) be the level of relevance for employee \( j \) on query \( q \). Then,

\[
DCG(q) = \sum_{i=1}^{k} \frac{2^{R(q, i)} - 1}{\log_2(1 + i)}.
\]

(5.3)

Because in general the size of the result set and the number of relevant employees can differ per query, the DCG score is normalized such that different queries can be compared. The normalized DCG (nDCG) is determined by dividing the DCG by the DCG of a perfect ordering. The perfect ordering is the result set sorted by the level of relevance.

The performance of a set of queries is calculated by the mean of the nDCG values of each query. We call this mean the Mean normalized Discounted Cumulative Gain (MnDCG).

### 5.4 Results: standard solution

We executed our queries on the Tam Tam document collection. Table 5.3 states the result per query. For clarity we also present in table 5.4 the top 10 results for each query where we indicated on which positions we retrieved experts. We also show the performance using only a specific document type. This gives an indication how useful certain documents are for the expert finding process. We present these results in table 5.5.

Table 5.3 shows that there are significant differences between the performance of the queries. The query with the highest AP is ICM and the query with the highest nDCG is Royal Haskoning. The query with the worst performance for both metrics by far is Zorg. The Candidate Model was previously tested on the TREC Enterprise Track 2007 and 2008 dataset. The MAP for the 2007 dataset was 0.484 and for the 2008 dataset 0.394. The MAP of our experiment (0.420) is similar to those results.
Table 5.3: Performance by query on the Tam Tam document collection. The average of the AP and nDCG columns are the MAP and MnDCG, respectively.

Table 5.4 shows that for 6 out of 8 queries we retrieve at least one expert in the top 3. For the query Overheid the first expert is found at rank 6 and for Zorg at rank 7.

Table 5.5 shows the difference in usefulness of the documents of a certain type. We see that the performance for Tweet and Email are substantial higher than the performance of the other document types.

Table 5.4: Top 10 per query for the standard model. X denotes a relevant employee and Max R is the maximum number of relevant employees for that query.

5.5 Observations

We observed three surprising results related to the P@n metric, the nDCG metric and the queries Overheid and Zorg.

The P@10 score is on average higher than the P@5 score. However, as can be seen in table 5.4, this does not mean that there are more experts found at rank 6 to 10 than in rank 1 to 5. This is contradictory and is likely caused by our adaption to the P@n metric. Would we have not adapted the metric the results would also not be correct. For example if all experts for the query Android (2 experts) would be retrieved in the
5.5 Observations

<table>
<thead>
<tr>
<th>Type</th>
<th>MAP</th>
<th>MnDCG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tweet</td>
<td>0.389</td>
<td>0.567</td>
</tr>
<tr>
<td>Blog</td>
<td>0.221</td>
<td>0.420</td>
</tr>
<tr>
<td>Comment</td>
<td>0.120</td>
<td>0.288</td>
</tr>
<tr>
<td>Note</td>
<td>0.136</td>
<td>0.299</td>
</tr>
<tr>
<td>Issue</td>
<td>0.146</td>
<td>0.329</td>
</tr>
<tr>
<td>Sharepoint</td>
<td>0.209</td>
<td>0.394</td>
</tr>
<tr>
<td>Project</td>
<td>0.125</td>
<td>0.336</td>
</tr>
<tr>
<td>Email</td>
<td><strong>0.383</strong></td>
<td><strong>0.555</strong></td>
</tr>
</tbody>
</table>

Table 5.5: The MAP and MnDCG per document type.

top 5 the P@5 score would be 0.4 instead of 1. P@n metrics are clearly unsuited as performance indicator when the number of relevant candidates is smaller than n. We therefore do not compare these scores to other research. To reduce confusion, we also exclude the P@n scores in the presentation of other results in this thesis.

For each query the nDCG scores are higher than the AP. Because different formulas are used it is logically that the values are not equal. However, the difference between the nDCG and AP, expressed in percentages, also differs per query. An analysis of the results show that for the queries with a large difference between the AP and the nDCG, the found experts are, based on the level of relevance, in the correct order.

For the queries Overheid and Zorg we did not find experts within the top 3 retrieved candidates. One assumption is that is caused by a lack of data. Table 5.6 lists the total number of occurrences of each query in the document collection. This table shows that the occurrences of a query in the document collection and the performance of a query are not correlated. A possible explanation for Zorg is that the word has two meanings. It is both a domain (healthcare) and the first-person singular simple present of the dutch verb zorgen (to take care). A manual evaluation of the first 50 twitter documents containing the query revealed that 48 were of the first and only 2 of the second form. The same evaluation for email documents revealed that only 14 were of the first form and 36 of the second form. Both meanings of the query zorg are thus used in the Tam Tam document collection. The query Overheid does not have this problem. We did not find problems with the use of that term. This indicates that the term is also frequently used by non-experts. Considering the query Zorg was ambiguous we conclude that for 6 out of 7 queries we were able to find an expert within the top 3 retrieved candidates.

This indicates that Tweets and Emails are more useful for the expert finding process.
<table>
<thead>
<tr>
<th>Query</th>
<th># occurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Android</td>
<td>839</td>
</tr>
<tr>
<td>ICM</td>
<td>3581</td>
</tr>
<tr>
<td>Mobile Development</td>
<td>5977</td>
</tr>
<tr>
<td>Overheid</td>
<td>1765</td>
</tr>
<tr>
<td>Royal Haskoning</td>
<td>5717</td>
</tr>
<tr>
<td>Sharepoint</td>
<td>22313</td>
</tr>
<tr>
<td>Smartsite</td>
<td>5728</td>
</tr>
<tr>
<td>Zorg</td>
<td>5450</td>
</tr>
</tbody>
</table>

Table 5.6: Number of occurrences in the Tam Tam document collection.
Chapter 6

Improving the solution

In this chapter we describe the prior probability of documents, called document priors, and how they can be applied to our solution. We propose two types of document priors. One document prior is based on the type of documents and one is based on the age of documents. We perform the same experiment as described in the previous chapter but with the inclusion of the document priors.

In section 6.1 we introduce document priors. The result of the experiment is described in section 6.2 followed by the observations in section 6.3.

6.1 Document priors

Document priors can be used to increase the accuracy of the model given that information about the relevance of documents is available. Document priors are denoted in the model by \( p(d) \), the probability that document \( d \) is relevant, independant from the query. Included in the model, document priors can change the relevance of a document based on some property. For example Kraaij demonstrated a simple document prior based on the assumption that longer documents are more relevant because they contain more information [18]. The probability \( p(d) \) was thus lower for documents with less words than for longer documents.

Balog showed that not all document types in the TREC Enterprise Tracks are of equal importance [2]. Table 5.5 shows that is also true for the Tam Tam document collection. We believe that by assigning a higher relevance to document with a document type that is shown to be more useful can increase the performance of our solution. In section 6.1.1 we describe the document prior based on document types.

The aspect of time was surprisingly not mentioned in the research on documents priors and expert finding. We believe, however, that the age of documents is an indication for the relevance of documents. In section 6.1.2 we describe various document priors based on the age of documents.

We use equation (4.3) to include document priors into our model. The following equation shows our model we use in our extended solution.

\[
p(q|ca) = \prod_{t \in q} \left\{ (1 - \lambda_{ca}) \left( \sum_{d \in D} p(t|d) \cdot p(d|ca) \cdot p(d) \right) + \lambda \cdot p(t) \right\}.
\]  

\[(6.1)\]
6.1 Document priors

6.1.1 Document types

Our first document prior is based on the type of the document. We relate the type of the document to the origin of the document. This means that all Twitter messages have the same type, but also that all documents from the Project Share have the same type. We calculate the relevance of a type by determining the performance of documents of that type based on two performance metrics; The Mean Average Precision (MAP) and the Mean normalized Discounted Cumulative Gain (MnDCG).

We estimate the prior probability that a document is relevant based on the type in the two following ways.

\[
p_{\text{typeMAP}}(d) = \frac{\text{MAP}(dt_d)}{\sum_{dt \in DT} \text{MAP}(dt)},
\]

\[
p_{\text{typeMnDCG}}(d) = \frac{\text{MnDCG}(dt_d)}{\sum_{dt \in DT} \text{MnDCG}(dt)},
\]

where \(\text{MAP}(dt_d)\) and \(\text{MnDCG}(dt_d)\) are the MAP and MnDCG calculated using only documents of the type \(dt\) of \(d\), and \(DT\) is the collection of all document types.

We will refer to these models as DTmap for equation 6.2 and DTdcg for equation 6.3.

6.1.2 Age of documents

Our second document prior involves the age of documents. We believe that newer documents are more relevant than older documents. We propose four different functions that calculate the document prior based on the age of documents. We define the age of a document to be the difference between its creation date and the creation date of the most recent document in the collection.

The first approach uses a window (an interval) to determine whether documents are relevant or not.

\[
p_{\text{ageWindow}}(d) = \begin{cases} 1 & \text{if } a(d) < \omega, \\ 0 & \text{otherwise}, \end{cases}
\]

where \(a(d)\) is the age of the document in days and \(\omega\) is the window size. Documents older than the window size are considered not relevant.

The second approach uses all documents but reduces the relevance according a monotonically decreasing function bases on the age of a document.

\[
p_{\text{ageReduced}}(d) = \frac{\delta}{\delta + a(d)},
\]

where \(\delta\) is a parameter that reduces the curve of the function. Higher values for \(\delta\) cause a smaller reduction of relevance in older documents.

The third approach uses a combination of these two models. Only documents within the window size are relevant and the relevance of older document within the window size is reduced.

\[
p_{\text{ageWinRed}}(d) = \begin{cases} \frac{\delta}{\delta + a(d)} & \text{if } a(d) < \omega, \\ 0 & \text{otherwise}, \end{cases}
\]

40
where $\beta$ is a parameter to smooth the curve and $\omega$ is the window size.

The fourth approach combines a window and a linear decreasing function to reduce the probability.

$$p_{\text{AGEwinredlin}}(d) = \begin{cases} 1 - \frac{a(d)}{\omega} & \text{if } a(d) < \omega, \\ 0 & \text{otherwise,} \end{cases} \quad (6.7)$$

where $\omega$ is the window size.

We will refer to the models as AGEwin for equation 6.4, AGRed for equation 6.5, AGEwinred for equation 6.6 and AGEwinredlin for 6.7.

## 6.2 Results: extended solution

In this section we present the results for the model extended with document priors. We first show the results for the document prior based on document type and second the document prior based on the document age.

### 6.2.1 Results with document priors based on document type

We determine the performance (MAP and MnDCG) of our model using only documents with a specific document type. We normalize the performance scores in table 5.5 and use these scores as input for the performance calculation for the complete document collection. For the DTmap document prior the MAP is 0.446 and a MnDCG of 0.591. For the DTdcg prior the MAP is 0.426 and the MnDCG is 0.570. Using DTmap the MAP increased with 6% and the MnDCG with almost 5%. Using DTdcg the MAP only increases 1.5% and the MnDCG only 1%.

### 6.2.2 Results with document priors based on age

For the models based on the age of documents we do not have the parameters needed to calculate the performance. We experimented with different values for the parameters and present the results for the best performing parameters.

**Results for AGEwin**

We experiment with different window sizes ranging from 25 to 4511 (all documents) for the AGEwin document prior. The MAP of the different window sizes are shown on a logarithmic scale in figure 6.1. Figure 6.1 shows that the performance is not a linear or polynomial function of the window size. The performance is maximal for a window size of 150 and drops significantly for higher values. The performance per document type for window size 150 is shown in table 6.1.

The MAP using AGEwin is increased by 31% and the MnDCG 17%. This is with a window size of 150 but not all window sizes increase the performance. A window size of 300 actually reduces the performance with 6%.
6.2 Results: extended solution

<table>
<thead>
<tr>
<th>Query</th>
<th>P@5</th>
<th>P@10</th>
<th>AP</th>
<th>nDCG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Android</td>
<td>1,000</td>
<td>1,000</td>
<td>0,583</td>
<td>0,597</td>
</tr>
<tr>
<td>ICM</td>
<td>0,500</td>
<td>0,750</td>
<td>0,594</td>
<td>0,771</td>
</tr>
<tr>
<td>Mobile Development</td>
<td>1,000</td>
<td>1,000</td>
<td>1,000</td>
<td>1,000</td>
</tr>
<tr>
<td>Overheid</td>
<td>0,000</td>
<td>0,000</td>
<td>0,088</td>
<td>0,290</td>
</tr>
<tr>
<td>Royal Haskoning</td>
<td>0,750</td>
<td>1,000</td>
<td>0,813</td>
<td>0,932</td>
</tr>
<tr>
<td>Sharepoint</td>
<td>0,200</td>
<td>0,833</td>
<td>0,544</td>
<td>0,568</td>
</tr>
<tr>
<td>Smartsite</td>
<td>0,400</td>
<td>0,600</td>
<td>0,520</td>
<td>0,542</td>
</tr>
<tr>
<td>Zorg</td>
<td>0,250</td>
<td>0,500</td>
<td>0,263</td>
<td>0,581</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td>0,513</td>
<td>0,710</td>
<td><strong>0,551</strong></td>
<td><strong>0,660</strong></td>
</tr>
</tbody>
</table>

Table 6.1: Performance by query of the AGEwin document prior with a window length \(\omega\) of 150. The average of the AP and nDCG columns are the MAP and MnDCG, respectively.

![Figure 6.1: The performance of the AGEwin document prior for various window sizes.](image)

**Results for AGEred**

We experimented with values ranging from 1 to 1000 for \(\delta\). The effect on the performance of various \(\delta\) is plotted on a logarithmic scale in figure 6.2. The performance increases up to \(\delta = 7\) after which the performance on average decreases.

The performance for the \(\delta = 7\) is shown in table 6.2. For \(\delta = 7\) the MAP increases 16% and the MnDCG increases 11% compared to our standard model. The performance using AGEred is in all cases higher than our standard model.

**Results for AGEwinred**

For the AGEwinred prior we experimented with values for the two parameters \(\delta\) and \(\omega\). Figure 6.3 shows the effect of various values of \(\delta\) for a number of window sizes. Figure 6.3 shows that the performance increases slowly converges and reaches it maximum at \(\delta = 100\). This is the case for all the window sizes we tested.

Figure 6.4 shows the performance for \(\delta = 100\) and various window sizes. We see the same irregular shape as with the AGEwin model. The performance is, for \(\delta = 100\),
### 6.2 Results: extended solution

Table 6.2: Performance by query of the AGEred document prior with $\delta = 7$. The average of the AP and nDCG columns are the MAP and MnDCG, respectively.

<table>
<thead>
<tr>
<th>Query</th>
<th>P@5</th>
<th>P@10</th>
<th>AP</th>
<th>nDCG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Android</td>
<td>1,000</td>
<td>1,000</td>
<td>0.450</td>
<td>0.627</td>
</tr>
<tr>
<td>ICM</td>
<td>0.500</td>
<td>0.500</td>
<td>0.554</td>
<td>0.695</td>
</tr>
<tr>
<td>Mobile Development</td>
<td>1,000</td>
<td>1,000</td>
<td>0.833</td>
<td>0.967</td>
</tr>
<tr>
<td>Overheid</td>
<td>0.000</td>
<td>0.000</td>
<td>0.110</td>
<td>0.308</td>
</tr>
<tr>
<td>Royal Haskoning</td>
<td>0.750</td>
<td>1,000</td>
<td>0.813</td>
<td>0.932</td>
</tr>
<tr>
<td>Sharepoint</td>
<td>0.200</td>
<td>0.833</td>
<td>0.542</td>
<td>0.556</td>
</tr>
<tr>
<td>Smartsite</td>
<td>0.400</td>
<td>0.600</td>
<td>0.441</td>
<td>0.527</td>
</tr>
<tr>
<td>Zorg</td>
<td>0.250</td>
<td>0.500</td>
<td>0.152</td>
<td>0.412</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td>0.512</td>
<td>0.679</td>
<td><strong>0.487</strong></td>
<td><strong>0.628</strong></td>
</tr>
</tbody>
</table>

Figure 6.2: The performance of the AGEred document prior for various $\delta$.

for all window sizes above our model without document priors. The best performing combination of parameters is $(\delta = 100, \omega = 150)$. Table 6.3 shows the performance per query for this combination.

The MAP compared to the standard model increases with almost 33% and the MnDCG with 16%.

**Results for AGEwinredlin**

For the AGEwinredlin prior we experimented with window sizes ranging between 25 and 4511. Figure 6.5 shows the performance on a logarithmic scale. We see an almost steady increase of the performance up to a window size of 175 after which the performance drops and converges. We show the performance with window size 175 in table 6.4.

The MAP is compared to the model without document priors increased with 27% and the MnDCG with almost 14%.
Table 6.3: Performance by query of the AGEwinred document prior with $\omega = 150$ and $\delta = 100$. The average of the AP and nDCG columns are the MAP and MnDCG, respectively.

<table>
<thead>
<tr>
<th>Query</th>
<th>P@5</th>
<th>P@10</th>
<th>AP</th>
<th>nDCG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Android</td>
<td>1,000</td>
<td>1,000</td>
<td>0.583</td>
<td>0.662</td>
</tr>
<tr>
<td>ICM</td>
<td>0.500</td>
<td>0.750</td>
<td>0.583</td>
<td>0.765</td>
</tr>
<tr>
<td>Mobile Development</td>
<td>1,000</td>
<td>1,000</td>
<td>1,000</td>
<td>1,000</td>
</tr>
<tr>
<td>Overheid</td>
<td>0.000</td>
<td>0.000</td>
<td>0.081</td>
<td>0.277</td>
</tr>
<tr>
<td>Royal Haskoning</td>
<td>1,000</td>
<td>1,000</td>
<td>0.950</td>
<td>0.981</td>
</tr>
<tr>
<td>Sharepoint</td>
<td>0.200</td>
<td>0.833</td>
<td>0.553</td>
<td>0.574</td>
</tr>
<tr>
<td>Smartsite</td>
<td>0.400</td>
<td>0.600</td>
<td>0.504</td>
<td>0.529</td>
</tr>
<tr>
<td>Zorg</td>
<td>0,250</td>
<td>0,500</td>
<td>0,200</td>
<td>0,462</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td>0.544</td>
<td>0.710</td>
<td><strong>0.557</strong></td>
<td><strong>0.656</strong></td>
</tr>
</tbody>
</table>

Figure 6.3: The performance of the AGEwinred document prior for various $\delta$ and window sizes.

Figure 6.4: The performance of the AGEwinred document prior with $\delta = 100$ for various window sizes.
6.3 Observations

The results from DTmap show a slight increase in performance whereas DTdcg shows almost none. We do not consider these significant improvements.

Our document priors based on the age of documents all had a possible affect on the performance. The best performing document prior is the AGEwinred with an increase of 33% for the MAP and 16% of the MnDCG. This showes that the age of documents can indeed be used to improve the performance.

The results of the different priors show that not all document should be stored or indexed forever. We increased the performance by using only newer documents. For Tam Tam the optimum was a window size of 150. Reducing the relevance for documents within the window increased the performance even further. Reducing the relevance with a non-linear function performed in our experiment better than using a linear function.

In table 6.5 we show the top 10 for each query for the AGEwinred prior. For 5 out of 8 queries we placed a relevant employee at rank 1. For 7 out 8 queries we placed a relevant employee either at rank 1 or 2. Considering that the query Zorg

<table>
<thead>
<tr>
<th>Query</th>
<th>P@5</th>
<th>P@10</th>
<th>AP</th>
<th>nDCG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Android</td>
<td>0.500</td>
<td>1.000</td>
<td>0.417</td>
<td>0.618</td>
</tr>
<tr>
<td>ICM</td>
<td>0.500</td>
<td>0.500</td>
<td>0.563</td>
<td>0.701</td>
</tr>
<tr>
<td>Mobile Development</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
</tr>
<tr>
<td>Overheid</td>
<td>0.000</td>
<td>0.000</td>
<td>0.073</td>
<td>0.267</td>
</tr>
<tr>
<td>Royal Haskoning</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
<td>0.997</td>
</tr>
<tr>
<td>Sharepoint</td>
<td>0.200</td>
<td>0.833</td>
<td>0.541</td>
<td>0.574</td>
</tr>
<tr>
<td>Smartsite</td>
<td>0.400</td>
<td>0.400</td>
<td>0.493</td>
<td>0.518</td>
</tr>
<tr>
<td>Zorg</td>
<td>0.250</td>
<td>0.500</td>
<td>0.186</td>
<td>0.447</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>0.481</strong></td>
<td><strong>0.654</strong></td>
<td><strong>0.534</strong></td>
<td><strong>0.641</strong></td>
</tr>
</tbody>
</table>

Table 6.4: Performance by query of the AGEwinredlin document prior with $\omega = 175$. The average of the AP and nDCG columns are the MAP and MnDCG, respectively.

Figure 6.5: The performance of the AGEwinredlin document prior for various values of $\omega$.

6.3 Observations

The results from DTmap show a slight increase in performance whereas DTdcg shows almost none. We do not consider these significant improvements.

Our document priors based on the age of documents all had a possible affect on the performance. The best performing document prior is the AGEwinred with an increase of 33% for the MAP and 16% of the MnDCG. This showes that the age of documents can indeed be used to improve the performance.

The results of the different priors show that not all document should be stored or indexed forever. We increased the performance by using only newer documents. For Tam Tam the optimum was a window size of 150. Reducing the relevance for documents within the window increased the performance even further. Reducing the relevance with a non-linear function performed in our experiment better than using a linear function.

In table 6.5 we show the top 10 for each query for the AGEwinred prior. For 5 out of 8 queries we placed a relevant employee at rank 1. For 7 out 8 queries we placed a relevant employee either at rank 1 or 2. Considering that the query Zorg
was ambiguous we conclude that again for 6 out of 7 queries we were able to find an expert within the top 3 retrieved candidates. Using AGEwinred we retrieved 10 experts in total for the unambiguous queries compared to 7 using our standard solution. Unfortunately we cannot reflect that in a comparison with the manual approach since we do not have more performance information of the manual process.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>#1</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>-</td>
</tr>
<tr>
<td>#2</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
<td>-</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>#3</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>#4</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td>X</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>#5</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>#6</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td>-</td>
<td>-</td>
<td>X</td>
<td>-</td>
</tr>
<tr>
<td>#7</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td>-</td>
<td>X</td>
<td>-</td>
<td>X</td>
</tr>
<tr>
<td>#8</td>
<td>-</td>
<td>X</td>
<td>-</td>
<td></td>
<td>X</td>
<td>X</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>#9</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>X</td>
</tr>
<tr>
<td>#10</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
<td>-</td>
<td>-</td>
<td>X</td>
<td>-</td>
</tr>
<tr>
<td>Max R</td>
<td>2</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>6</td>
<td>5</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 6.5: Top 10 per query for AGEwinred with $\delta = 100$ and $\omega = 150$. X denotes a relevant employee and Max R is the maximum number of relevant employees for that query.
This chapter concludes the work done in this thesis. We state our conclusions in section 7.1. We describe the future work in section 7.2.

7.1 Conclusions

Finding employees with a specific level of expertise, experts, is vital for knowledge sharing and information reuse. A company benefits when its employees have access to an efficient system for expert finding, because time spent searching is time lost. Many companies, however, do not have such a system. In this thesis we have shown that we can automate the manual expert finding process of a company.

We performed a case study where we automated the manual expert finding process of the company Tam Tam. Two requirements were posed on our solution. The solution should be scalable and perform as good as or better than the manual approach. To meet these requirements we used the Candidate Model, a model from existing Expert finding research, in our solution. We chose to use this model to determine the association between topics, documents and employees. Results from our experiments show that we indeed fulfilled these requirements.

The Candidate Model was previously tested on the TREC Enterprise Track dataset. The performance of our implementation of the standard model applied to the document collection of Tam Tam is similar to the performance of that test. We also investigated how to increase the performance of our standard model. We show (in various ways) that using the age of documents as measure for prior relevance increases the performance. Using the AGEwinred document prior in our extended model we increase the performance by 33% compared to the standard model. In practise this means that for 85% of the queries, within the top 2 retrieved employees, there is at least one expert.

The main advantage of our solution is that all steps in the expert finding process are automated. Discovering, processing and indexing documents is all done automatically. Furthermore, our solution does not require employees to create documents or to do extra work. Only documents that would be created anyway during their normal daily jobs are needed as input.

In our solution using our standard model we did not make assumptions based on properties specific for Tam Tam. We based our solution on a proven model and
achieved similar result. This motivates our belief that our solution can also be applied to other companies resulting in the same performance.

7.2 Future work

In this section we differentiate between future work related to Tam Tam and future work related to the research field of Expert Finding. The future work for Tam Tam is focussed on techniques that can improve the performance of the current solution and is practically oriented. The future work described for the research field is more scientifically oriented.

7.2.1 Future work for Tam Tam

We implemented and tested our solution at Tam Tam and achieved a good performance. We used simple approaches to determine the document-topic association and the document-candidate association. More sophisticated approaches exist and we believe that these approaches can also increase the performance compared to the current implemented solution.

For Tam Tam we have seen that Twitter documents and Email documents are the most useful sources for Expert Finding. We did not utilize the specific properties of these documents. A Twitter document (a tweet) can, for example, be retweeted by another user. The main social meaning of such an action is that the other user values the content of the tweet. We believe that this information is valuable and can be used to increase the performance of the Expert Finding process. Emails contain a subject and a body which, with some effort, can be split in the original (replied to) text and new text. Using this structure of email was previously researched on emails from publicly available mailing lists. We believe this can also improve the performance using private corporate emails.

7.2.2 Future work for the field of Expert Finding

Ambiguous queries have a lower performance than unambiguous queries in information retrieval systems. We have also seen this in our experiments. How to deal with ambiguous queries is a difficult problem and needs to be solved. A possible research question is: How can Expert Finding systems understand the meaning of an ambiguous query? Possible approaches are making the EF system context-aware or using a taxonomy for query terms.

In the Tam Tam document collection we have seen that besides document-topic and document-employee associations there are also employee-employee associations. There are many situations in which a connection between employees can be made. Employees working on the same project, employees that co-author documents, employees that check-in files in the same source code repository and employees have contact via email or chat are some of the many examples. For the field of Expert Finding it is very interesting to know if these relations between employees are correlated to expertise. A possible research question for this topic is: How can relations between candidate experts be used to increase the performance of Expert Finding systems?
We have shown in our experiments that utilizing the age of documents leads to significant increases in the performance. Our research is, to the best of our knowledge, the first to use such information as document prior. Further research has to be done to show if the performance increase of this document prior is similar on other datasets.
Bibliography


