ENTITY FINDING IN A DOCUMENT COLLECTION USING ADAPTIVE WINDOW SIZES

By

Fawaz Khaled A. Alarfaj

A thesis submitted for the degree of Doctor of Philosophy

School of Computer Science and Electronic Engineering

University of Essex

February 2016

بسم الله الرحمن الرحيم

This thesis is dedicated to the memory of my beloved grandfather

Ahmed A. A. Alarfaj (1920 – 2008)

Acknowledgements

Completing my Ph.D. has been a long journey full of invaluable educational experiences, both academic and personal. At the end of this journey, it is my pleasure to extend my deepest appreciation to all of those who were there for me during this time.

The first and biggest thanks goes to my parents, whose love, support and encouragement made it possible for me to succeed. Next is my darling son, Khaled, whose smiles were the remedy to my greatest problems. To my cherished uncle, Ibrahim, to whom I turned during my dilemmas, and who's never been short on kindness or wisdom.

A great deal of gratitude is due to my supervisors, Udo Kruschwitz and Chris Fox, for their time, constructive suggestions and insights. Without their professional guidance and assistance, this work would not have been achieved. In addition, thanks go to my two examiners, Professor Stefan Ruger and Dr. Michael Gardner, who provided useful feedback and an interesting Viva.

A special note of thanks goes to my dear friends and colleagues in Colchester who made living away from my home and family more bearable.

Last, but not least, I am grateful to my employer, Imam Mohammed bin Saud University, for supporting my studies.

Fawaz Khaled A. Alarfaj
Colchester, UK
29 February 2016

Abstract

Traditional search engines work by returning a list of documents in response to queries. However, such engines are often inadequate when the information need of the user involves entities. This issue has led to the development of entity-search, which unlike normal web search does not aim at returning documents but names of people, products, organisations, etc. Some of the most successful methods for identifying relevant entities were built around the idea of a proximity search. In this thesis, we present an adaptive, well-founded, general-purpose entity finding model. In contrast to the work of other researchers, where the size of the targeted part of the document (i.e., the window size) is fixed across the collection, our method uses a number of document features to calculate an adaptive window size for each document in the collection. We construct a new entity finding test collection called the ESSEX test collection for use in evaluating our method. This collection represents a university setting as the data was collected from the publicly accessible webpages of the University of Essex.

We test our method on five different datasets including the W3C Dataset, CERC Dataset, UvT/TU Datasets, ESSEX dataset and the ClueWeb09 entity finding collection. Our method provides a considerable improvement over various baseline models on all of these datasets. We also find that the document features considered for the calculation of the window size have differing impacts on the performance of the search. These impacts depend on the structure of the documents and the document language.

As users may have a variety of search requirements, we show that our method is adaptable to different applications, environments, types of named entities and document collections.

Contents

List of Figures

List of Tables

Acronyms and Technical Terms

AP Average Precision

ANOVA A variance analysis method

BM25 A document ranking function developed by Robertson et al. [1994]

bpref A metric that evaluates an ordered set to find out whether relevant items are ranked above irrelevant ones

Candidate Entity Any possible entity of the type specified by the user in the query

Candidate Frequency The number of candidates found in a document

Candidate Probability The probability that the candidate is an expert

Candidate Proximity The proximity of pieces of candidate evidence in the document to the query

CERC CSIRO Enterprise Research Collection

CRF Conditional Random Field

Document-based model A proximity model that uses the whole document as a window

EEL Expert-Expert-Locator

Entity Evidence A piece of text information that will allow us to identify an entity

ExB Expertise Browse

ER Expert Recommender

FBM Frequency Based Model

Gold Standard The judgement of relevance

GT Ground Truth

HTTP Hypertext Transfer Protocol

IR Information Retrieval: a field of study concerns with the representation, storage, organisation of, and the access to, information items

Language Model A probabilistic model of text for a document

LM Language Modeling

MAP Mean Average Precision

MMR Mean Reciprocal Rank

Narrative A field of a TREC topic

NER Named Entity Recognition

NLP Natural Language Processing

Passage Retrieval A retrieval strategy in which each document is broken up into contiguous blocks of text called passages which are compared to the query terms. Each query is answered with a list of passages

PBM Proximity Based Model

PDF Portable document format

Precision The proportion of a retrieved set that is relevant

Priors In Bayesian statistics, a prior is the belief held about a quantity before some evidence is taken into account

P@n A metric measuring precision at the recall level n

RDBMS Relational Database Management System

Relevance The extent to which a candidate entity matches the user's query

Recall The proportion of all relevant documents from the collection which are included in the retrieved set

r**-precision** A metric measuring precision at level r

Stop Words Words that are relatively meaningless and may be filtered out from the list of indexing terms

Test Collection The data required to successfully replicate a set of experiments

Topic a query

TREC Text Retrieval Conference

TU An updated version of the UvT collection

URL Uniform Resource Locator

UvT A test collection developed using public data about the employees of Tilburg University in the Netherlands.

VSM Vector Space Model

Window Size The number of words around the query term that are considered in the proximity search

XML Extensible Markup Language

Symbols

Notation used in Chapter 2

q	a query, which is a description of the user's information need in free text
d	a document in the collection
t	a term: the indexed unit; it is usually a word
C	a document collection
$R(q, d_i)$	The IR model rank for the document d_i given the query q
tf	the term frequency in any given document
idf	the inverse document frequency
$\text{score}(d_i, q)$	the relevance score of a document d_i for the query q
θ_d	a document model for the document d
θ_c	a candidate model for the candidate c
λ_c, λ_d	smoothing parameters used in the language models
$n(t, q)$	the number of times that term t occurs in the query q
$n(t, C)$	the number of times the term t occurs in the document collection C
$n(C)$	the total number of words in the document collection C
$p(q)$	the probability that the query is q
$p(c)$	the probability that the candidate c is an expert
$p(d)$	the prior belief that document d is relevant
$p(q\|d_i)$	the probability that the query was q, given that the document d is relevant
$p(c\|q)$	the probability that the candidate c is relevant expert for the query q
$p(q\|c)$	the probability that the query is q, given that the identified candidate is c
$p(t)$	the probability that the term t appears in the document collection
$R(q)$	the set of documents returned for the query q
$\text{profile}(C)$	the set of documents that belong to the candidate profile C
qtf	the frequency of a term within a specific query

dl	a document length, represented by the number of terms in the document
$avdl$	the average document length in a collection of documents
R	number of documents known to be relevant to a specific query
N	the total number of documents in the collection
$score\_cand(C, q)$	the score for each candidate with respect to the query q, used in the voting model
$r(n)$	the number of relevant information items included in the top n retrieved items
$r-$precision	precision at level r
AP	the average precision metric
$rel(E)$	an indicator function that returns 1 if the information item at position E is relevant and 0 otherwise

Notation used in Chapter 3

c	A candidate expert/entity
D_c	The set of supporting documents
λ_c	A general smoothing parameter equal to $\frac{\beta}{\beta+n(ca)}$
$P(A)$	Probability function, probability of event A
$P(A\|B)$	Conditional probability function, probability of event A given event B occurred
θ	A language model inferred by the multinomial probability distribution
$p(c\|d, q)$	relevance of candidate entity c to query q when only document d is considered
$n(A)$	Counter, the number of times event A occurred
t_q	a token of the query q in a document
t_c	a token of the candidate c in a document
$\delta(t_1, t_2)$	the distance between terms t_1 and t_2 in a document
$n(c, d)$	the number of pieces of evidence found for the candidate c in the document d
$\mathscr{D}$	a document collection
$\|\mathscr{D}\|$	the total number of documents in the collection $\mathscr{D}$

$t_p(c)$	the total proximity of candidate c; obtained by aggregating the proximity related to each query
$t_p(c, Q, d)$	the total proximity value for the candidate c in the document d
k	a kernel function
α_i	the weighting of document feature i in the adaptive window size calculation

Notation used in Chapter 5

WindowSize	the size of the adaptive window
σ	a variable that allows us to scale the window size
α_l	the weighting of the document-length feature
α_c	the weighting of the candidate-frequency feature
α_v	the weighting of the average-sentence-size feature
α_r	the weighting of the readability-index feature

Part I

Theoretical Background

1 Introduction

This chapter provides a general motivation of the entity search problem which is at the core of this thesis. I will present the problems with current approaches which then leads to the formulation of two research questions. The Chapter concludes with an outline of the thesis.

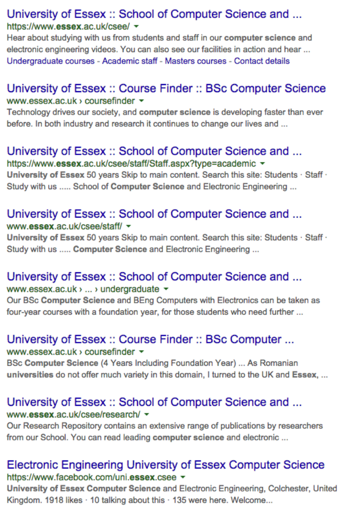
1.1 Motivation

Traditionally, the majority of research in the field of information retrieval (IR) has been directed at retrieving documents. Here the task is to identify a ranked list of documents relevant to the user's query. Nowadays, the process of generating data has become much easier and more rapid than before, resulting in the proliferation of the digital content available online [Marz and Warren, 2015; Zikopoulos et al., 2015]. To keep pace with this enormous growth in data, a broad range of IR-related areas which go beyond basic document retrieval have been introduced. A proportion of this new interest has been directed at special IR tasks and, in particular, at entity-oriented search [Fang and Si, 2015].

Users resort to search engines for a range of information needs. Typically, search engines return a list of documents, but these documents do not provide the most satisfactory answer. If they are seeking specific entities, users will need to skim through a large number of documents to find what they are looking for. This problem highlights the need for a special type of search, commonly known as an entity-oriented search.

Figure 1.1 allows a comparison of the results provided by a standard document retrieval system and an entity-oriented retrieval system on a sample search. In this example, the information need is to find some experts in the field of *computer science* at the University of Essex. The list of documents returned by the standard document retrieval system is not ideal, as extracting the required information could prove to be both an effort and a time-consuming task. By contrast, the entity-oriented retrieval system has retrieved only the required information.

Various studies have shown that users are increasingly interested in searching for entities rather than documents [Hertzum and Pejtersen, 2000]. This is particularly true in the case of web searches (see, for example, Blanco et al., 2013).



University of Essex :: School of Computer Science and ...
https://www.**essex**.ac.uk/csee/
Hear about studying with us from students and staff in our **computer science** and electronic engineering videos. You can also see our facilities in action and hear ...
Undergraduate courses - Academic staff - Masters courses - Contact details

University of Essex :: Course Finder :: BSc Computer Science
www.essex.ac.uk › coursefinder
Technology drives our society, and **computer science** is developing faster than ever before. In both industry and research it continues to change our lives and ...

University of Essex :: School of Computer Science and ...
https://www.**essex**.ac.uk/csee/staff/Staff.aspx?type=academic
University of Essex 50 years Skip to main content. Search this site: Students · Staff · Study with us ..... School of **Computer Science** and Electronic Engineering ...

University of Essex :: School of Computer Science and ...
www.**essex**.ac.uk/csee/staff/
University of Essex 50 years Skip to main content. Search this site: Students · Staff · Study with us ..... **Computer Science** and Electronic Engineering ...

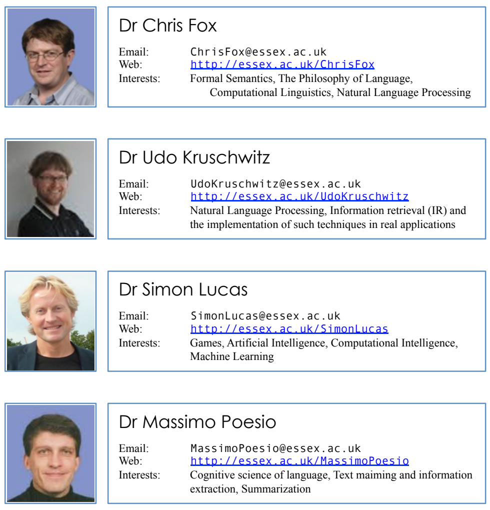
University of Essex :: School of Computer Science and ...
www.essex.ac.uk › ... › undergraduate
Our BSc **Computer Science** and BEng Computers with Electronics can be taken as four-year courses with a foundation year, for those students who need further ...

University of Essex :: Course Finder :: BSc Computer ...
www.essex.ac.uk › coursefinder
BSc **Computer Science** (4 Years Including Foundation Year) ... As Romanian **universities** do not offer much variety in this domain, I turned to the UK and **Essex**, ...

University of Essex :: School of Computer Science and ...
www.**essex**.ac.uk/csee/research/
Our Research Repository contains an extensive range of publications by researchers from our School. You can read leading **computer science** and electronic ...

Electronic Engineering University of Essex Computer Science
https://www.facebook.com/uni.**essex**.csee
University of Essex Computer Science and Electronic Engineering, Colchester, United Kingdom. 1918 likes · 10 talking about this · 135 were here. Welcome...

(a) Results of a standard document retrieval system



Dr Chris Fox
Email: ChrisFox@essex.ac.uk
Web: http://essex.ac.uk/ChrisFox
Interests: Formal Semantics, The Philosophy of Language, Computational Linguistics, Natural Language Processing

Dr Udo Kruschwitz
Email: UdoKruschwitz@essex.ac.uk
Web: http://essex.ac.uk/UdoKruschwitz
Interests: Natural Language Processing, Information retrieval (IR) and the implementation of such techniques in real applications

Dr Simon Lucas
Email: SimonLucas@essex.ac.uk
Web: http://essex.ac.uk/SimonLucas
Interests: Games, Artificial Intelligence, Computational Intelligence, Machine Learning

Dr Massimo Poesio
Email: MassimoPoesio@essex.ac.uk
Web: http://essex.ac.uk/MassimoPoesio
Interests: Cognitive science of language, Text maiming and information extraction, Summarization

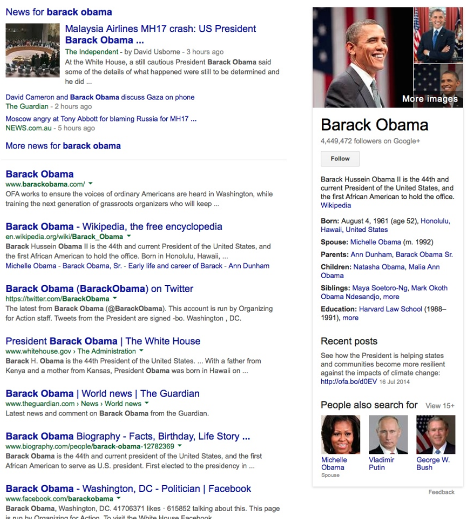
(b) Results of an entity oriented retrieval system

Figure 1.1: A comparison between a standard document retrieval system and an entity-oriented retrieval system

A number of internet search engines have recently begun to recognise this need; thus, in many instances, entity search results are now integrated with the result list [Ganesan and Zhai, 2012].

Figure 1.2 presents an example of the results returned by two search engines (Google and Bing) when asked to search for "Barack Obama". These search engines recognised that the user was probably more interested in an entity (in this case, a person), and consequently they generated a "slider" on the right-hand side of the page containing some basic information about this entity.

Web search is not the only scenario in which we search for entities. In fact, much entity-finding research is centred on enterprise search. Regardless of the type of organisation, enterprises typically need to deal with a vast amount of data including internal documents, emails, web pages, etc. Such documents need to be organised, indexed, and retrieved by any enterprise search application. It has been established that improving the quality of such applications reduces business costs and produces positive business outcomes [Frischmuth et al., 2012; Hawking, 2004]. However, despite the progress made in improving web search over



News for **barack obama**

Malaysia Airlines MH17 crash: US President **Barack Obama ...**
The Independent - by David Usborne - 3 hours ago
At the White House, a still cautious President **Barack Obama** said some of the details of what happened were still to be determined and he did ...

David Cameron and **Barack Obama** discuss Gaza on phone
The Guardian - 2 hours ago

Moscow angry at Tony Abbott for blaming Russia for MH17 ...
NEWS.com.au - 5 hours ago

More news for **barack obama**

Barack Obama
www.**barackobama**.com/
OFA works to ensure the voices of ordinary Americans are heard in Washington, while training the next generation of grassroots organizers who will keep ...

Barack Obama - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/**Barack_Obama**
Barack Hussein **Obama** II is the 44th and current President of the United States, and the first African American to hold the office. Born in Honolulu, Hawaii, ...
Michelle Obama - Barack Obama, Sr. - Early life and career of Barack - Ann Dunham

Barack Obama (**BarackObama**) on Twitter
https://twitter.com/**BarackObama**
The latest from **Barack Obama** (**@BarackObama**). This account is run by Organizing for Action staff. Tweets from the President are signed -bo. Washington , DC.

President **Barack Obama** | The White House
www.whitehouse.gov › The Administration
Barack H. **Obama** is the 44th President of the United States. ... With a father from Kenya and a mother from Kansas, President **Obama** was born in Hawaii on ...

Barack Obama | World news | The Guardian
www.theguardian.com › News › World news
Latest news and comment on **Barack Obama** from the Guardian.

Barack Obama Biography - Facts, Birthday, Life Story ...
www.biography.com/people/**barack-obama**-12782369
Barack Obama is the 44th and current president of the United States, and the first African American to serve as U.S. president. First elected to the presidency in ...

Barack Obama - Washington, DC - Politician | Facebook
www.facebook.com/**barackobama**
Barack Obama, Washington, DC. 41706371 likes · 615852 talking about this. This page

More images

Barack Obama
4,449,472 followers on Google+
Follow

Barack Hussein Obama II is the 44th and current President of the United States, and the first African American to hold the office. Wikipedia

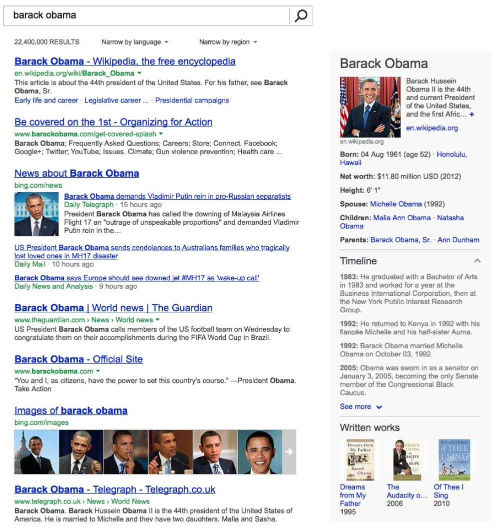
Born: August 4, 1961 (age 52), Honolulu, Hawaii, United States
Spouse: Michelle Obama (m. 1992)
Parents: Ann Dunham, Barack Obama Sr.
Children: Natasha Obama, Malia Ann Obama
Siblings: Maya Soetoro-Ng, Mark Okoth Obama Ndesandjo, more
Education: Harvard Law School (1988–1991), more

Recent posts
See how the President is helping states and communities become more resilient against the impacts of climate change: http://ofa.bo/d0EV 16 Jul 2014

People also search for View 15+
Michelle Obama Spouse
Vladimir Putin
George W. Bush

Feedback

(a) Google Hits for Barack Obama query



barack obama

22,400,000 RESULTS Narrow by language Narrow by region

Barack Obama - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/**Barack_Obama**
This article is about the 44th president of the United States. For his father, see **Barack Obama**, Sr.
Early life and career · Legislative career ... · Presidential campaigns

Be covered on the 1st - Organizing for Action
www.**barackobama**.com/get-covered-splash
Barack Obama; Frequently Asked Questions; Careers; Store; Connect. Facebook; Google+; Twitter; YouTube; Issues. Climate; Gun violence prevention; Health care ...

News about **Barack Obama**
bing.com/news
Barack Obama demands Vladimir Putin rein in pro-Russian separatists
Daily Telegraph · 15 hours ago
President **Barack Obama** has called the downing of Malaysia Airlines Flight 17 an "outrage of unspeakable proportions" and demanded Vladimir Putin rein in the...

US President **Barack Obama** sends condolences to Australians families who tragically lost loved ones in MH17 disaster
Daily Mail · 10 hours ago

Barack Obama says Europe should see downed jet #MH17 as 'wake-up call'
Daily News and Analysis · 9 hours ago

Barack Obama | World news | The Guardian
www.theguardian.com › News › World news
US President **Barack Obama** calls members of the US football team on Wednesday to congratulate them on their accomplishments during the FIFA World Cup in Brazil.

Barack Obama - Official Site
www.**barackobama**.com
"You and I, as citizens, have the power to set this country's course." —President **Obama**. Take Action

Images of **barack obama**
bing.com/images

Barack Obama - Telegraph - Telegraph.co.uk
www.telegraph.co.uk › News › World News
Barack Obama. **Barack** Hussein **Obama** II is the 44th president of the United States of America. He is married to Michelle and they have two daughters, Malia and Sasha.

Barack Obama
Barack Hussein Obama II is the 44th and current President of the United States, and the first Afric... +
en.wikipedia.org
en.wikipedia.org

Born: 04 Aug 1961 (age 52) · Honolulu, Hawaii
Net worth: $11.80 million USD (2012)
Height: 6' 1"
Spouse: Michelle Obama (1992)
Children: Malia Ann Obama · Natasha Obama
Parents: Barack Obama, Sr. · Ann Dunham

Timeline
1983: He graduated with a Bachelor of Arts in 1983 and worked for a year at the Business International Corporation, then at the New York Public Interest Research Group.
1992: He returned to Kenya in 1992 with his fiancée Michelle and his half-sister Auma.
1992: Barack Obama married Michelle Obama on October 03, 1992.
2005: Obama was sworn in as a senator on January 3, 2005, becoming the only Senate member of the Congressional Black Caucus.
See more

Written works
Dreams from My Father 1995
The Audacity o... 2006
Of Thee I Sing 2010

(b) Bing Hits for Barack Obama query

Figure 1.2: Modern search engines recognise some types of entities.

the years, there are still many unresolved challenges in enterprise search development. Identifying and finding specific entities is one such challenge. Another challenge lies in the heterogeneous nature of data sources. Enterprise data includes a mixture of documents with various structural levels, the text of which is often embedded with many entities and their relationships.

As discussed above, entity oriented searches, also known as entity finding applications, aim to assist users with their specific information needs. These applications suggest possible *candidate entities* with some degree of *relevance* in response to a query. The term, "candidate entity", is used to describe any possible entity of the type specified by the user in the query. For instance, if the user's information need is to find experts in "computer science" working at a given university, then the candidate set would be the academic staff members of that university. Moreover, candidate relevance is defined to be the extent to which a candidate entity matches the user's query. In the previous example, a relevant candidate would be a member of the academic staff at the given university with some knowledge of and expertise in the query topic, “computer science”.

Numerous systems have been developed to address the issue of entity finding. A common shortcoming of these systems is that most of them are dependant on a particular environment or a specific document type.

In this thesis, our aim is to develop and test the validity of an adaptive, well-founded, general-purpose approach to entity finding. We have increased the generality of our search by focusing attention on the documents in which the candidate entities have been mentioned, rather than incorporating external sources such as organisational hierarchies and social networks. The proximity between query terms and the candidate entities in these documents is considered to be an important factor which could give a strong indication of the relevance of candidates.

A number of state-of-the-art entity finding models have employed the notion of proximity in their searches. However, these models have measured proximity within a fixed-sized span of text called a *window*. We argue that each document has its own unique features that distinguish it from the other documents in the collection, and that this may lead to the need for differing window sizes for different documents. We utilise a number of the distinguishing features of each document in order to determine the optimal size of the window for each document. Thus, the window sizes for different documents in the collection will vary, depending upon their distinguishing features.

1.2 Research Questions

This thesis investigates the following research questions:

RQ1: Can the state of the art in entity finding be pushed forward by employing document features to determine the size of the window in proximity-based approaches? We call this approach adaptive-window-size approach to distinguish it from current approaches that employ fixed-sized windows.

RQ2: Will an adaptive-window-size approach to entity finding be robust across different types of named entities and different types of document collections?

1.3 Main Contributions

- We introduce a novel entity finding algorithm that employs a notion of proximity developed by exploiting a document's features to set an adaptive window of text for capturing the association between candidate entities and queries.
- We use a variety of expert and entity finding experiments to thoroughly evaluate the performance of our algorithms using standard test collections.
- We provide evidence showing that our approach is robust and capable of delivering a very competitive performance.

1.4 List of Publications

This thesis contains work from the following papers which have previously been published in the refereed literature:

1. **Fawaz Alarfaj**, Udo Kruschwitz, David Hunter, and Chris Fox, "Finding the right supervisor: Expert-finding in a university domain", *In Proceedings*

of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics Human Language Technologies - Student Research Workshop, NAACL-HLT, pp. 1–6, Montréal, Canada, May 2013.

2. **Fawaz Alarfaj**, Udo Kruschwitz, and Chris Fox, "Adaptive Window Size Selection for Proximity Search" *In Proceedings of the Fifth BCS-IRSG Symposium on Future Directions in Information Access*, BCS, Granada, Spain, September 2013.

3. **Fawaz Alarfaj**, Udo Kruschwitz, and Chris Fox, "An adaptive window-size approach for expert-finding" *In Proceedings of the 13th Dutch-Belgian Workshop on Information Retrieval (DIR 2013)*, Delft, The Netherlands, September 2014. [Best paper award]

4. **Fawaz Alarfaj**, Udo Kruschwitz, and Chris Fox, "Exploring adaptive window sizes for entity retrieval" *In 36th European Conference on IR Research, ECIR 2014*, Springer, pp. 573-578, Amsterdam, The Netherlands, April 2014.

5. **Fawaz Alarfaj**, Udo Kruschwitz, and Chris Fox, "Experiments with query expansion for entity finding" *In the 16th International Conference on Intelligent Text Processing and Computational Linguistics, CICLing 2015.*, Cairo, Egypt, April 2015.

1.5 Organisation of the Thesis

This thesis is organised into two main parts, each part including a number of chapters.

Part I, "Theoretical Background", introduces the general domain and discusses related work in this area. It also presents a discussion of the theory behind the models proposed in this thesis. This part includes the following chapters:

Chapter 2 briefly outlines the information retrieval field and discusses some of its metrics and techniques including the different document retrieval models which are used as part of the entity finding system. Background information about some of the early entity/expert finding models is presented and details about a number of state-of-the-art methods and techniques are given. Moreover, the evaluation of entity/expert finding models is discussed. An overview of some of the well-known expert and entity finding test collections is provided. In addition, the evaluation paradigms that are usually used in the assessment of such systems are described. Next, proximity search, which forms the basis of our approach, is described in some detail. The chapter concludes with a discussion of the key differentiators of the proposed approach.

Chapter 3 introduces the adaptive window size approach in detail. It begins with an outline of the frequency-based model which will be used as a baseline for, as well as a component of, our approach. Next, the proximity-based model is described. This model includes enhancements to the ranking function. Finally, the chapter introduces the idea of adaptive window size for proximity ranking.

Part II, "Evaluation", provides an assessment of the method proposed in Part I. Readers who wish to skip the introductory chapters are referred to this part which describes the experiments performed on the method proposed in Part I and discusses the findings from these experiments. This part includes the following chapters:

Chapter 4 describes the setup of the experiments. It considers the general system architecture and the process of preparing the document collection, including document pre-processing and index-creation. In addition, the approach to algorithm evaluation used in this thesis is discussed. The chapter concludes with a discussion of the baseline models used for comparison.

Chapter 5 details the experiments performed on the adaptive window approach. In particular, we describe a pilot study that investigates the validity of the various components of this approach. This pilot study makes a comparison between the effects of three document features, namely document length, number of candidates in the document, and average sentence size on the generation of the adaptive window. The method is systematically investigated in various environments, using thorough experimentation to determine their effect on retrieval performance. Finally, we study the effects of using different query expansion techniques and document retrieval methods in the entity finding task.

Chapter 6 presents a general discussion of our approach. It compares a number of methods used and described in this work and investigates the effect of using the adaptive window approach on different tasks, environments and data types. Furthermore, we consider some particular issues that influence this approach, including the use of different document features and documents in various languages. We also discuss the main limitations of our work.

Chapter 7 concludes this thesis by revisiting the initial research questions, and discussing our contributions towards these. In addition, it discusses possible further directions for research, building on the investigations presented in this thesis.

2 Background

The aim of this chapter is to review existing work in the field of entity retrieval and its recent advances. We introduce some key concepts from the field of information retrieval which is the main focus of this thesis. These include document retrieval models and techniques. This is followed by a detailed discussion of the entity and expert finding tasks, and an overview of the state-of-the-art models for these tasks. We also describe how IR systems are evaluated before defining the main test collections designed to evaluate entity-finding systems. Next, we discuss the proximity search on which our approach relies.

2.1 Information Retrieval Models

Information retrieval (IR) is a field of computer science which deals primarily with providing easy access to information of interest for the person interacting with the system. IR involves the representation, storage, organisation of, and access to information items [Baeza-Yates and Ribeiro-Neto, 2011].

The person interacting with the system (the "user") expresses their information need as a request which is called a *query*. Queries are usually bags of keywords in a natural language that summarise the user's information need. To satisfy this information need, the role of the IR system is twofold. After retrieving the relevant information items, it ranks these items according to their degrees of *relevance* to the user's query.

The notion of relevance is a fundamental concept in information retrieval that lies at the heart of any IR system. A relevant information item is an item which contains the information a user is looking for when submitting the query. The purpose of an IR system could be defined as to retrieve all of the relevant documents at the same time, while retrieving the smallest number of non-relevant documents that is possible [van Rijsbergen, 1979]. For a human, appraising an information item as relevant or non-relevant could be considered a simple task. This task is more complicated for a machine to accomplish. There are many factors that may influence a person's decision, and an IR system requires a *retrieval model* to assess these factors and quantify the degree of relevance of a given information item [Croft et al., 2015]. Baeza-Yates and Ribeiro-Neto [2011] note that most research in information retrieval has been concerned with some aspects of such a model.

It is important to distinguish between the task of information retrieval and that of data retrieval. The aim of an IR system is to retrieve information items with high degrees of relevance and to rank these items more highly than non-relevant items. On the other hand, the aim of a data retrieval system is to retrieve all exact matches [Kowalski and Maybury, 2002]. The notion of relevance has less impact here as data retrieval relies on a deterministic model [Frakes, 1992], and so every item is either a match or not. The appearance of a single erroneous item in the result set would mean a total failure for the data retrieval system [Baeza-Yates and Ribeiro-Neto, 2011].

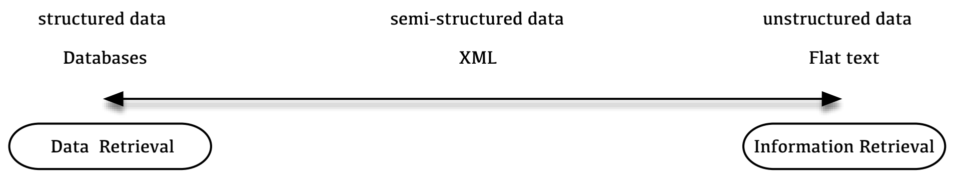


Figure 2.1: The data retrieval and information retrieval on the data landscape.

Figure 2.1 shows a representation of the data landscape. On the left is the structured data to which data retrieval is usually applied, as is the case for relational database management systems (RDBMS). On the right is the unstructured data on which information retrieval is used. A common example of an information retrieval system is a web search engine. The middle of the diagram shows semi-structured data. Depending on the task at hand, either information retrieval or data retrieval systems may be applied to this data [Consens and Baeza-Yates, 2005].

The document retrieval task lies at the core of IR and is considered to be a vital component of many advanced IR applications, including question answering and entity finding systems. In this task, the information items retrieved by the system are documents which can include any type of unstructured text, such as news articles and web pages, or text with some structure such as XML documents or e-mail messages. Search engines are common examples of document retrieval applications.

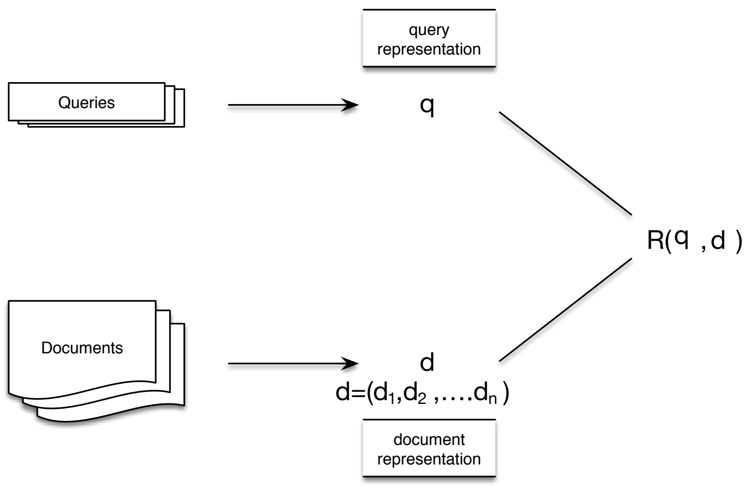


Figure 2.2: A general overview of the document retrieval process. The output of this process is a list of document rankings, $R(q, d)$.

The function of an IR model is to rank documents with respect to a query (see Figure 2.2). There are a number of ways of achieving this. Next, we will give a quick review of the history of document retrieval including the early IR models, followed by a review of some state-of-the-art IR models. These state-of-the-art models will be used in our experiments.

IR, and in particular, the document retrieval task can be traced back to the early age of electronic computers [Singhal, 2001]. Beginning with Herman Hollerith's tabulating machine in the early 1900's, information retrieval became one of the tasks that were commonly explored using the breakthrough technology of the time [Williams, 2002]. The idea of automatic access to large amounts of stored knowledge was articulated by Vannevar Bush in his 1945 ground breaking article entitled "As We May Think" [Bush, 1945]. As early as the 1950s, different approaches were proposed and tested to solve the problem of document retrieval. Most notably, the work of Luhn [1957, 1958] and Maron et al. [1959] used probabilistic approaches and established the idea of term frequency (*tf*) weighting. Among the major developments in the 1960s were the idea of representing documents as vectors [Salton, 1968; Switzer, 1964], the introduction of the relevance feedback concept [Rocchio, 1965], and the development of the Cranfield evaluation methodology for retrieval systems [Cleverdon, 1967]. The advances of the 1950s and 1960s continued to influence research throughout the 1970s. Statistical models were enhanced by the introduction of the inverse document frequency concept (*idf*) [Spärck Jones, 1972] and the development of the *tf* · *idf* weighting scheme [Salton and Yang, 1973]. Moreover, the vector space model for document retrieval was introduced by Salton et al. [1975]. Among the main advances in IR from the 1980s is the development of the Porter stemmer [Porter, 1980] which is still used by some IR systems today. Developments from the early 1990s include the introduction of the latent semantic indexing (LSI) model [Deerwester et al., 1990] and the BM25 ranking function [Robertson et al., 1994].

Up until the mid 1990s, most retrieval systems have been evaluated and proved to be successful on small collections of documents. This motivated the development of systems that could handle larger collections. Two important developments arose to resolve this issue. First, the Text Retrieval Conference (TREC) [Harman, 1992] was introduced. This is concerned with the generation of test sets of documents and questions, and creates a common platform for the evaluation and testing of IR systems [Singhal, 2001; Voorhees and Harman, 2005]. The second important factor giving the IR research community access to larger test collections was the rise of the World Wide Web, which rapidly increased the scale of data and created an urgent need to access and retrieve this data [Sanderson and Croft, 2012].

2.1.1 Vector-Space Model (VSM)

The Vector-Space Model (VSM) was one of the earliest models developed for IR. In this model, the degree of similarly between every document and the user query is calculated [Salton and McGill, 1986]. As noted by Sanderson and Croft [2012], the VSM model was adopted by many research retrieval systems for many years. Some forms of this model are still in use as the default retrieval algorithm for open source search engines[1].

In the VSM model, both documents and queries are represented as vectors in a t-dimensional space, where t is the number of terms in the collection. A document d_i in the collection is represented by a vector of length t:

$$d_i = (d_{i1}, d_{i2}, d_{i3}, \ldots, d_{it}).$$

Here d_{in} is the weight of the n-th term in document d_i. A query q is represented by

$$q = (q_1, q_2, q_3, \ldots, q_t),$$

where q_n is the weight of the n-th term in the query.

There are various ways to define the term weight. One of the simplest approaches is to assign this weight to equal the number of occurrences of the term t in the document d [Manning et al., 2008].

Using this representation, documents are ranked by computing the similarity between the document and the query. Different similarity functions can be used. Commonly a cosine function is used, as functions of this form have shown better performance than other similarity measures [Croft et al., 2015]. The cosine

[1]`http://lucene.apache.org/core/4_9_0/core/org/apache/lucene/search/similarities/TFIDFSimilarity.html`

similarity function is given in Equation (2.1):

$$\text{cosine}(d_i, q) = \frac{\sum_{j=1}^{t} d_{ij} \cdot q_j}{\sqrt{\sum_{j=1}^{t} {d_{ij}}^2 \cdot \sum_{j=1}^{t} {q_j}^2}}. \tag{2.1}$$

VSM provides a simple, yet effective, ranking formula for IR [Baeza-Yates and Ribeiro-Neto, 2011]. However, the VSM model has come in for some criticism, especially because of its entirely heuristic nature. This shortcoming has motivated the development of models that are claimed to be more solidly grounded in theoretical frameworks such as probabilistic modelling and statistical language modelling [Büttcher et al., 2010].

2.1.2 Probabilistic modelling

The Okapi team at City University, London have developed a series of probabilistic weighting models called Best Match (BM) models [Robertson et al., 1992a,b]. These models led to the development of BM25 [Robertson et al., 1994] which is considered to be one of the best term-weighting algorithms for document retrieval [Goker and Davies, 2009].

The BM25 algorithm defines a measure called the *relevance score*, which indicates the similarity between a document and a search query and uses this measure in the document ranking process. BM25 estimates the relevance score of a document d_i for a query q as follows:

$$\text{score}(d_i, q) = \sum_{t \in q} \log \frac{(r_t + 0.5)/(R - r_t + 0.5)}{(n_t - r_t + 0.5)/(N - n_t - R + r_t + 0.5)} \cdot \frac{(k_1 + 1) tf_d}{K + tf_d} \cdot \frac{(k_2 + 1) qtf}{k_2 + qtf}. \tag{2.2}$$

Here the summation is taken over every term t in the query. The frequency of a term within a specific query is denoted by qtf and the number of documents known to be relevant to a specific query is denoted by R. The number of relevant documents containing the term t is denoted by r_t. The overall number of documents containing the term t is denoted by n_t and N denotes the total number of documents in the collection. The frequency of the term t within the document d is denoted by tf_d. The above equation contains the following constant parameters: k_1, k_2, and K, whose default settings are $k_1 = 1.2$ and $k_2 = 1000$ [Robertson et al., 1994]. The final parameter, K, is responsible for normalising the term frequency component by the document length and is given by:

$$K = k_1 \left((1 - b) + b \cdot \frac{dl}{avdl} \right).$$

Here dl is the document length in tokens and $avdl$ is the average document length in the collection. The constant b is the term frequency normalisation hyper-parameter, for which the default setting is $b = 0.75$ [Robertson et al., 1994].

2.1.3 Language modelling

The traditional probabilistic approach ranks documents by modelling the probability of each document being relevant. In other words, it estimates the probability that a document d_i is of relevance $R = 1$ as follows: $p(R = 1|d_i)$. The idea in language modelling is to define a probabilistic model of text (i.e., a probability distribution over the sequences of words) for each document in the collection. This is known as a *language model*. The documents are ranked by the probabilities of their document language models generating the query [Croft and Lafferty, 2003; Zhai, 2008]. Unlike the traditional probabilistic approach, language modelling does not estimate relevance directly, rather it asks a different question: "How likely is it that document d_i would produce the query q?" [Spärck Jones et al., 2003].

Language modelling was first introduced into information retrieval by Ponte and Croft [1998]. Before that, language modelling had been successfully used in other areas including speech recognition [Jelinek, 1997; Rabiner, 1989], and machine translation [Brown et al., 1990].

In this approach, Bayes' rule is used to rank documents based on their probability of generating the query, so that

$$p(d_i|q) = \frac{p(q|d_i) \cdot p(d_i)}{p(q)}. \tag{2.3}$$

In the above Equation, $p(q)$ does not influence the ranking, and can, therefore, be safely discarded [Croft et al., 2015]. Further, $p(d_i)$ is the prior belief that d_i is relevant. If we assume $p(d_i)$ to be equal for all documents, then it will not affect the ranking [Berger and Lafferty, 1999; Song and Croft, 1999].

With these simplifications, the retrieval model can rank documents by query likelihood $p(q|d_i)$. First, a document model θ_d is inferred for each document. Next, the probability of the query, given the document model, $p(q|\theta_d)$ is computed. Because we have assumed term independence, we may estimate $p(q|\theta_d)$ using the equation

$$p(q|\theta_d) = \prod_{t \in q} p(t|\theta_d)^{n(t,q)}, \tag{2.4}$$

where $n(t, q)$ is the number of times the term t occurs in the query q. One way to calculate $p(t|\theta_d)$ is by using the maximum likelihood (ML) approach, which provides the simplest method for inferring a document model [Zhai, 2008]. We calculate

$$p(t|\theta_d) = \frac{n(t, d_i)}{n(d_i)},$$

where $n(t|d_i)$ is the number of times that term t appears in document d_i, and $n(d_i)$ is the total number of terms in that document.

One drawback of this approach is that if a term t from the query is not present in the document model θ_d, then the document will be assigned a zero probability. This sparseness problem means that all queries containing an unseen word have zero probability $p(Q|\theta_d)$. This is undesirable, as a document may still be considered relevant even if it does not contain all of the query words. To prevent this undesirable effect, it is important to smooth the ML approach so that it assigns a non-zero probability to these documents. Different smoothing methods can be used. In our work, we use Jelinek-Mercer Smoothing [Jelinek and Mercer, 1980]. This method smooths unseen words by using a coefficient λ to control the influence of unseen words based on the frequency of the occurrence of words in entire document collection as follows:

$$p(t|\theta_d) = (1 - \lambda) \cdot \frac{n(t, d_i)}{n(d_i)} + \lambda \cdot \frac{n(t, C)}{n(C)}.$$

Here $n(t, C)$ is the number of times the term t occurs in the collection of documents, and n(C) is the total number of word occurrences in the collection. The default value of λ is 0.15 [Hiemstra, 2001].

2.2 Entity and Expert Finding

Although document retrieval might be considered to be the main task in the information retrieval field, information retrieval goes beyond merely retrieving documents. In fact, there is a renewed interest in a broad range of IR-related areas including question answering, topic detection and tracking, and multimedia retrieval [Allan et al., 2003].

Entity Search has become an active research topic in the field of information retrieval [Balog et al., 2010; Demartini et al., 2010]. Recent studies have shown that a large proportion of web queries are, in fact, about entities. For instance, Kumar and Tomkins [2010] showed that about 52.9% of web search queries are entity-oriented queries. Moreover, Guo et al. stated that, according to their analysis, about 71% of web search queries contain named entities [Guo et al., 2009]. The bulk of research in this area began in the enterprise domain with the introduction of the expert finding task which is a sub task of entity finding. In this section we discuss the expert finding task and describe some expert-finding models.

People are the kinds of entities most frequently sought using entity search [Borgatti and Cross, 2003; Hertzum, 2014; Hertzum and Pejtersen, 2000]. Often users search for people when the required information is not readily available in an accessible form, perhaps because the information is not publicly available in a digital form, or because the requirements are somewhat esoteric. In such situations, the only way to acquire the required information is to find the right person, who we call *an expert.* One effective way to find experts is by extracting this knowledge from textual sources, i.e., documents. The need for locating experts is shown more clearly in enterprise settings where expert-oriented search has become an important part of many search applications [Bailey et al., 2007b; Balog et al., 2008].

Expert-finding has been applied in a range of settings including collaborator discovery [Tang et al., 2012], reviewer assignments [Karimzadehgan et al., 2008], and co-author prediction [Han et al., 2013]. There has been a substantial body of research motivated by the need to find experts within organisations as a tool for supporting knowledge-sharing and transfer [Ackerman et al., 2003; Yimam-Seid and Kobsa, 2009]. Since the 1980s, a number of expert-finding systems have been developed, such as the *Answer Garden* system [Ackerman and McDonald, 1996], *ContactFinder* [Krulwich and Burkey, 1996], and *Expert-Expert-Locator (EEL)* [Streeter and Lochbaum, 1988]. These systems suffer from a number of shortcomings. One is the need for manual involvement. The types of manual involvement include the creation, maintenance, and updating of profiles for every possible expert. Performing these tasks at an organisational level will result in high maintenance costs. Another shortcoming lies in the lack of semantic connections: in some cases a query needs to be provided with an additional context in order to return relevant results.

During the 1990s and early 2000s some commercial systems for finding experts started to appear in a number of organisations including *Hewlett-Packard (HP)* [Hansen et al., 1999], *Microsoft* [McCampbell et al., 1999], *MITRE* [Mattox et al., 1999], and *NASA* [Becerra-Fernandez, 2000a]. However, little academic research was published on this topic. The situation changed with the introduction of the Enterprise Track at TREC 2005 [Craswell et al., 2005]. Between 2005 and 2008, TREC hosted a series of workshops aimed at fostering research on the topic of finding experts in enterprises. An expert was defined to be the employee in the enterprise with the most knowledge about a given query [Craswell et al., 2005]. An important outcome of the TREC expert finding task was the creation of a general platform to develop and test expert-finding algorithms which include a number of datasets and evaluation benchmarks. This task has helped to foster research in expert finding, resulting in the introduction of a number of expert finding algorithms including the statistical models [Balog et al., 2006; Fang and Zhai, 2007], voting model [Macdonald and Ounis, 2006], and graph-based models [Serdyukov et al., 2008].

Expert finding continues to attract the attention of many researchers, most recently in the study of social media websites like Facebook [Hecht et al., 2012] and Twitter [Cheng et al., 2014; Ghosh et al., 2012], and community question answering websites such as Yahoo! Answers [Aslay et al., 2013] and Stack Overflow [Yang et al., 2013].

In the following section we will discuss some of the main models developed for expert finding. We follow our description of the earliest models with a discussion of the state of the methods.

2.2.1 Early models

A number of models have previously been developed to address the problem of expert finding. Traditionally, this problem has been a concern of the knowledge management field [Davenport and Prusak, 2000]. The main idea was to generate a unified data warehouse for the storage of the organisation's required expertise that can easily be accessed and searched. These early systems commonly formed a component of much larger knowledge management or computer supported systems. Some common names for this component were "yellow pages"[2], people-finding systems or expertise-management [Ackerman and Halverson, 2004; Davenport and Prusak, 2000]. Some examples of these early systems are the *SPUD* system at Microsoft [Davenport and Prusak, 2000], HP's *CONNEX* system [Davenport and Prusak, 2000], and the *People Finder* system at SAGE [Becerra-Fernandez, 2000b]. The main drawback of the aforementioned systems was that candidates' profiles were manually created.

Many attempts have been reported to develop automatic expert finders. For instance, the work of McDonald and Ackerman [2000] describes a study in which

[2]Yellow Pages was SUN's original name for the Unix implementation of X.500-based hierarchical directory services (cf. LDAP), which could be used to find experts (and services) in an organisation in the same that a physical Yellow Pages directory could. It is now called NIS.).

experts were located within the technical development and support departments of a medium-sized software company. They developed a system called Expertise Recommender (ER) and identified the following three aspects of expert finding:

Expertise identification: finding a set of candidates who are likely to have the desired expertise.

Expertise selection: the way participants picked one person (or a small number of people) to approach for help.

Escalation: the mechanism that fixes breakdowns in identification and selection.

The system is comprised of four major parts: (1) profiling of the supervisor responsible for creating and maintaining profiles, (2) identification of the supervisor that picks a set of items, or people who are reasonable candidates for a recommendation, (3) selection of the supervisor for refining the recommendation by re-ordering and possibly removing items, and (4) interaction with the managers that maintain and manipulate expertise profiles. The modules developed by McDonald and Ackerman are tailored very specifically to the given organisation (i.e., Medical Software Corporation), and employ several heuristics.

In later work, McDonald developed the Expertise Recommender system evaluation [McDonald, 2001]. The evaluation was limited to the assessment of only two expertise identification heuristics. The participants of the study were asked to judge their colleagues' expertise in some topic domains, and then the Expertise Recommender system was evaluated against these judgements. The findings in [McDonald, 2001] suggest that the participants made relatively good judgements about the expertise of their colleagues. The study also showed that the agreement between the participants was much stronger than their agreement with the system.

Another early example of work in this field is a system called Expertise Browse (ExB) which operates in a collaborative software engineering environment to locate experts [Mockus and Herbsleb, 2002]. The aim of this web-based tool is twofold: (1) to identify experts in terms of their degree of expertise, and (2) to determine the expertise profile of a particular person or a group of people. In their work, Mockus and Herbsleb quantitatively interpreted expertise by counting experience atoms (EAs) (i.e. elementary units of experience).

Mockus and Herbsleb used the following measures of expertise: (1) software delivery, (2) type and functionality of the product part, (3) the technology used, and (4) the purpose or type of change. Data sets for these measures were obtained from the change management systems of various software projects. These heuristics were manually generated, based on current working practices. The usage data from the Expertise Browser tool was analysed, and feedback was collected from the system's users. However, no formal evaluation of the accuracy of this tool was conducted.

Other early systems included ContactFinder [Krulwich and Burkey, 1995, 1996], *Answer Garden* [Ackerman and McDonald, 1996], and the *MEMOIR* system [Pikrakis et al., 1998]. A common limitation of these early models was their dependence on specialised settings. Models were usually developed for a specific environment, or particular document types.

One of the early attempts at a general purpose system was proposed by Craswell et al. [2001] who introduced the P@noptic system. This system was designed with large organisations in mind. The system created virtual documents for each candidate (or employee). These virtual documents were simply made up of the concatenated texts of all documents from the corporation that were associated with each particular candidate. During a search, the system matched the query against these virtual documents. The results would include a list of the ten best matching experts, together with a list of matching intranet documents as supporting evidence for the suitability of each expert. A limitation of this system was its bias toward candidates whose virtual documents contained large chunks of text. It is noteworthy that this system has formed the starting point for many current expert-finding models.

As mentioned before, the introduction of an expert-finding task at TREC attracted a significant amount of attention from the research community. TREC has provided a common platform for researchers to empirically assess methods and techniques devised for expert finding.

Two prominent approaches to expert-finding have emerged from the TREC workshops, namely, the *candidate model* and *document model*, which we shall henceforth refer to as *Model 1* and *Model 2*, respectively. The main difference between these models is that Model 1 builds a textual representation of candidate experts, and then ranks the candidates, based on the given query, whereas Model 2 first finds documents that are relevant to the query, and then locates the associated experts in these documents. A study comparing these models Balog et al. [2006], showed that that Model 2 outperformed Model 1 on all measures. As Model 2 proved to be more efficient, it was used as the basis of many other expert-search systems [Fang and Zhai, 2007; Petkova and Croft, 2007; Yao et al., 2008]. The models have been formalised and compared by Balog et al. [2006]. The two models have been discussed in the literature under various names. For example, Fang and Zhai [2007] referred to them as '*Candidate Generation Models and Topic Generation Models*', whereas Petkova and Croft [2008] differentiated between Model 1 and Model 2 by referring to them as the '*Query-Dependent Model and Query-Independent Model*', respectively.

Various refinements of both models which estimate the association between a candidate and a topic of expertise have been suggested in the literature. For instance, some frameworks capture the associations at multiple levels such as the document level, paragraph level, or snippet level [Rode et al., 2008]. Other refinements include extra forms of evidence such as document and candidate evidence through the use of *priors*[3] [Fang and Zhai, 2007], the document structure [Zhu et al., 2006], and the use of hierarchical, organisational, and topical context and structure [Petkova and Croft, 2008]. Fang and Zhai developed a mixture model which used proximity-based document representation. In this mixture model it is possible to put different weights on different representations of a candidate expert [Fang and Zhai, 2007].

Petkova and Croft [2008] propose extra refinements to the TREC models in which the search topic is modelled in accordance with the relevance models for document retrieval [Lavrenko and Croft, 2001]. In this framework, a pseudo-relevance feedback is used to create the topic model which, in turn, is matched against the document and candidate models. Fang and Zhai [2007] apply an enhanced framework to the models by adapting some query expansion techniques. Petkova and Croft [2007] introduce an effective method for modelling the dependency between the candidates' evidence and terms in a document. A candidate-centred document representation is used which employs positional information using proximity kernels.

Another mixture of personal and global language models was proposed by Serdyukov et al. [2008]. Their so-called person-centric method combined two criteria for personal expertise in the final ranking: the probability that the query would be generated by the personal language model, and a prior probability based on the level of activity of candidate experts in important discussions on the query topic.

[3]In Bayesian statistics, a prior is a belief held about a quantity before some evidence is taken into account

Zhu et al. [2010] claimed that previous language models did not consider document features. They proposed an approach that incorporated the following five document features: internal document structure, document URLs, page rank, anchor texts, and multiple levels of association between experts and topics.

In the following sub-sections we will give an overview of some state-of-the-art models for expert finding.

2.2.2 Language models

Balog et al. [2006] developed a statistical model for expert finding, which employed language models to rank experts. They introduced two different approaches to the problem: Model 1 and Model 2, as discussed in the previous section.

In this work, Bayes' Theorem was used to estimate the probability $p(c|q)$ of a candidate c being an expert, given the query q:

$$p(c|q) = \frac{p(q|c) \cdot p(c)}{p(q)}. \tag{2.5}$$

Here $p(q)$ is the query probability and $p(c)$ is the candidate probability. The candidates with the highest probability are considered to be the *most likely* experts for the query q. As discussed in Section 2.1.3, it is apparent that, for a given query, the value of $p(q)$ cannot be changed and, consequently, it can safely be ignored in the ranking process. Therefore, the probability that candidate c is a relevant expert for a given query q can be determined as the probability $p(q|c)$ that the query is q, given that the identified candidate is c, weighted by the probability $p(c)$ that candidate c is an expert:

$$p(c|q) \sim p(q|c) \cdot p(c). \tag{2.6}$$

The candidate model θ_c is calculated for each candidate c. Using such a model, the probability $p(t|\theta_c)$ of a term in the query, given the candidate model is evaluated. Next, the model θ_c is used to predict the likelihood that the query q is produced by a candidate c. The assumption is that each term in the query is sampled independently from the other terms. Hence, the probability of a query is obtained by calculating the following product across all terms:

$$p(q|\theta_c) = \prod_{t \in q} p(t|\theta_c)^{n(t,q)}. \tag{2.7}$$

Here $n(t, q)$ denotes the number of times term t is present in query q. To obtain an estimate of the candidate model $p(t|\theta_c)$, the probability $p(t|c)$ of a term, given a candidate, should first be determined. To ensure that there are no zero probabilities using the function, this probability is smoothed as follows:

$$p(t|c) = (1 - \lambda_c) \cdot p(t|c) + \lambda_c \cdot p(t), \tag{2.8}$$

where $p(t)$ is the probability that the term t appears in the document collection. The probability, $p(t|c)$, is calculated using the following formula:

$$p(t|c) = \sum_{d \in D_c} p(t|d, c) \cdot p(d|c). \tag{2.9}$$

Here D_c is the set of all documents associated with the candidate c.

In Equation 2.9, the probability of a term t, given a candidate c could be determined by weighting the probability $p(t|d, c)$ of the co-occurrence between a term and a candidate in a particular document by the probability $p(d|c)$ of the document, given the candidate.

2.2.2.1 Model 1

In this model, a textual representation of a candidate's expertise is built using the documents with which he is associated. The probability of the query topic can be estimated from this textual representation, given the candidate model. The textual representation used in this model is a multinomial probability distribution over the vocabulary of terms, according to which the probability of a query, given the candidate model, would be:

$$p(q|\theta_c) = \prod_{t \in q} \left\{ (1-\lambda_c) \cdot \left(\sum_{d \in D_c} p(t|d) \cdot p(d|c) \right) + \lambda_c \cdot p(t) \right\}^{n(t,q)} . \tag{2.10}$$

The smoothing parameter λ_c is set to $\frac{\beta}{\beta + n(c)}$, where $n(c)$ is the number of term occurrences in the documents associated with the candidate. Model 1 (Equation 2.10) aggregates all of the terms from all documents associated with the candidate to form a pseudo document that represents the candidate. The query likelihood is then estimated from the candidate model.

2.2.2.2 Model 2

In this Model, documents are modelled rather than candidates. The document model is then queried and the candidates most closely associated with the highest ranked documents are returned as the possible experts. In this scenario, the document acts like a hidden variable in the process (see Figure 2.3). It used as

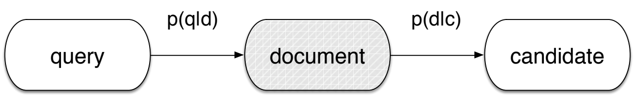


Figure 2.3: Documents act like a hidden variables in Model 2

the connection between the querying process (retrieving the relevant document given the query) and the candidate-finding process (ranking the candidate using only the retrieved documents).

The process of finding an expert is as follows. Assume documents $d_1, \ldots, d_n$ have been retrieved and ranked in response to a query. Each document d_i is processed by identifying the candidates associated with that document. This association is considered as evidence of their knowledge about the topic.

The probability $p(q|c)$ of a query, given a candidate, can be found by taking the following sum over all documents in which the candidate appears

$$p(q|c) = \sum_{d \in D_c} p(q|d,c) \cdot p(d|c). \tag{2.11}$$

Using the assumption that all query terms are sampled independently, the probability of the query, given a candidate and a document, can be found using the formula:

$$p(q|d,c) = \prod_{t \in q} p(t|d,c)^{n(t,q)} \tag{2.12}$$

For each document d, a document model θ_d is derived. Using such a model, the probability that the term t is generated, given that we have document model θ_d would be:

$$p(t|\theta_d) = (1 - \lambda_d) \cdot p(t|d) + \lambda_d \cdot p(t), \tag{2.13}$$

where λ_d is determined from the length n_d of the document d, using the formula $\lambda_d = \frac{\beta}{\beta + n_d}$. Here β is the average length of all documents in the collection. Finally, the equation used in Model 2 for the probability that the query is q, given that the candidate is c, is as follows:

$$p(q|c) = \sum_{d \in D_c} \prod_{t \in q} \{(1 - \lambda_d) \cdot p(t|d) + \lambda_d \cdot p(t)\}^{n(t,q)} \cdot p(d|c). \tag{2.14}$$

2.2.3 Voting Model

In contrast to those frameworks that rank candidates based on the frequency of their mentions in relevant documents, the Voting model developed by Macdonald et al. [2008] used a voting process to rank candidates based on their profiles and a list of documents generated by the user query. In this model, a candidate profile is constructed from a set of documents that represent a candidate's expertise. It does not matter whether these profiles are generated manually or automatically.

In this model, expert search is seen as a voting problem. For a given query, each document retrieved by the underlying search engine that also forms part of the candidate profile can be considered to be a vote for that candidate to have expertise relevant to the query. The ranking of the candidate is then determined from the number of accumulated votes.

Figure 2.4 shows the application of this model on a simple example where $R(q)$ is the set of documents retrieved for a given query q. The set of documents that belong to candidate profile C is denoted by profile(C). In this example, the search engine has ranked the documents in the following order: rank(D_x) > rank(D_y) > rank(D_z) > rank(D_w). When we look at the candidate profiles, we see that candidate C_1 has 2 votes, candidate C_2 has 1 vote, candidate C_3 has 3 votes, and candidate C_4 has no votes.

For the sake of simplicity, we shall assume that all votes are counted equally, and that each document is equally weighted. A possible candidate ranking would be C_3 first, then C_1, and finally C_2.

The score for each candidate with respect to the query is denoted here by $score\_cand(C, q)$. It is the aggregation of votes from all documents d having two features. First, d must be retrieved by the search engine in response to the query.

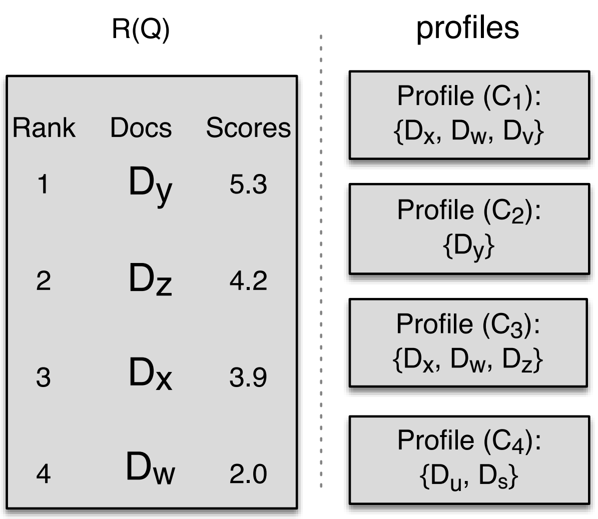


Figure 2.4: A Voting model example for expert finding as mentioned in [Macdonald and Ounis, 2006]

Second, d must belong to the profile of the candidate. The set of eligible voting documents d is then defined to be: $(d \in R(q) \cap \mathit{profile}(C))$.

Several vote aggregation or voting techniques can be used, each of which may produce different rankings of the candidates. One of these voting techniques is the Reciprocal Rank (RR). In this method, the rank of a document is determined by taking the sum of the reciprocals of the corresponding ranks received by the document in each of the individual rankings:

$$score\_cand_{RR}(C,q) = \sum_{d \in R(q) \cap \mathit{profile}(C)} \frac{1}{rank(d,q)}. \tag{2.15}$$

Another method of aggregation is *CombSUM* or the "score aggregation technique". In this technique, the score of a document is the sum of the normalised scores received by the document. The candidate's score is given by:

$$score\_cand_{CombSUM}(C,q) = \sum_{d \in R(q) \cap \mathit{profile}(C)} score(d,q) \tag{2.16}$$

Macdonald et al. [2008] define the twelve voting techniques shown in table 2.1 for use with their model.

Table 2.1: Summary of the voting techniques.

Name	Relevance score of candidate is
Votes	$\|D(C, q)\|$
RR	sum of inverse of ranks of docs in $D(C, q)$
BordaFuse	sum of ($\|R(q)\|$ - ranks of docs in $D(C, q)$)
CombMED	median of scores of docs in $D(C, q)$
CombMIN	minimum of scores of docs in $D(C, q)$
CombMAX	maximum of scores of docs in $D(C, q)$
CombSUM	sum of scores of docs in $D(C, q)$
CombANZ	CombSUM $\div$ $\|D(C, q)\|$
CombMNZ	$\|D(C, q)\|$ $\times$ CombSUM
expCombSUM	sum of exp of scores of docs in $D(C, q)$
expCombANZ	expCombSUM $\div$ $\|D(C, q)\|$
expCombMNZ	$\|D(C, q)\|$ $\times$expCombSUM

2.2.4 User Preferences

Some researchers have argued that it is not enough to find the experts by looking only at the queries, and not taking the characteristics of the users into consideration. They claim that there are several factors that may play a role in decisions concerning which experts to recommend. Some of these factors are the users' expertise level, social proximity and physical proximity [Borgatti and Cross, 2003; McDonald and Ackerman, 1998; Shami et al., 2008]. McDonald and Ackerman [1998] emphasised the importance of the accessibility of the expert. They argued that people usually prefer to contact experts who are physically or organisationally close to them. Moreover, Shami et al. [2008] found that people prefer contact experts they know, even when they could potentially receive more information from other experts who are located outside their social network.

Woudstra and van den Hooff [2008] identified a number of factors affecting the selection of experts that are related to quality and accessibility. They argued that the process of choosing which candidate expert to contact might differ depending on the specific situation.

Hofmann et al. [2010] showed that many of these factors can be modelled. They claimed that integrating user preference modelling with retrieval models can improve retrieval performance. Smirnova and Balog [2011] provided a user-oriented model for expert-finding in which emphasis was placed on the social distance between the user and the expert. They considered a number of social graphs based on organisational hierarchy, geographical location and previous collaboration.

2.2.5 Extraction of Candidate List

One requirement for the application of entity finding systems is the availability of a list of potential candidate entities which contains all of the forms in which information about a potential candidate may appear in the text.

A candidate list for expert finding applications provides the information needed to recognise experts in the collection. It is common in enterprise settings to use the organisation's staff list to build candidate lists [Hertzum, 2014], as in the W3C collection. Alternatively, the system may automatically generate the list of candidates from a corpus of documents. The list may initially include only one of the possible representations of a candidate. For example, it may only include an email address for each candidate. However, the candidate might also be represented in other forms in the organisation's documents. We usually refer to the different forms that represent a candidate in a document as the *candidate evidence.* Common evidence usually includes different variations of the candidate's name [Azzopardi et al., 2005], which can be derived from the full name. An example of candidate evidence is shown in Table 2.2.

Table 2.2: Given the name "Simon M. Lucas", the table lists examples of candidate evidence, as used in this study.

Simon M. Lucas	Lucas, Simon	Simon Lucas
S. Lucas	Lucas, S.	Lucas, S. M.
Lucas, Simon M.	Lucas	Simon.Lucas@domain.com

Many researchers have used only unambiguous evidence in their models. However, Fang and Zhai [2007]; Petkova and Croft [2007] among others, consider ambiguous evidence in their models, by associating it with different weights. The weightings for given pieces of evidence are defined, based on the probability of whether they have been shared by more than one person in the collection. For instance, Bao et al. [2006] identified six evidence types of candidate occurrences along with their ambiguity weights for the W3C dataset. Moreover, Jiang et al. [2008] proposed a model where the probability of each piece of evidence is estimated separately, and then these probabilities are combined to produce the final result.

In entity finding applications, the process of generating a candidate list is more challenging as different kinds of entities may be involved, not just individuals. For such applications, the candidate list is generated by aggregating all entities of the same type that have been mentioned in the documents retrieved by the underlying search engine. More details about candidate evidence will be given in Chapter 3.

2.3 Expert Finding Evaluation

In IR, user satisfaction can be interpreted as the maximisation of effectiveness, which is achieved by retrieving the maximum number of relevant information items, while simultaneously retrieving the minimum number of irrelevant information items. As discussed in Section 2.1, the measurement of correctness of IR systems differs from that of database systems. When responding to a query, IR systems should return relevant information items before irrelevant ones, whereas database systems must return nothing but correct results.

It is important for all scientific experiments to be reliably repeatable. This makes it possible to compare, and evaluate different retrieval models for the same task. The goal of reliable repeatability motivated the design of the Cranfield evaluation paradigm back in the 1960s [Cleverdon, 1967]. The first Cranfield experiment, known as *Cranfield-1*, was designed to make a comparison between four indexing systems. In this experiment, 18,000 papers were manually indexed in each of the systems, and the results of 1,200 queries were compared. Each query was designed to be satisfactorily answered by a single paper. If this paper was found in the catalogue, then the search was considered to be successful [Cleverdon, 1991]. The Cranfield paradigm became a standard for information retrieval evaluation [Harman, 2011].

The Cranfield evaluation paradigm was built around the idea of having a *test collection* which was used for benchmarking, and allowed for the comparison of different methods and practices [Sanderson, 2010; Voorhees, 2002; Voorhees and Harman, 2005].

In general, a test collection is the data required to successfully replicate a set of experiments. For information retrieval tasks, a test collection typically consists of three parts [Manning et al., 2008]:

1. A corpora of documents (i.e., dataset), over which a search is performed
2. A test suite of information needs, usually referred to as *queries*
3. A set of relevance judgements indicating the correct answers to each query

It is common for relevance judgements to come in the form of binary assessments [Voorhees, 2002]. This is a simplification of the assumption that an information item is either relevant, or not relevant with respect to a user's information need. In this work, we use the phrases *gold standard* or *ground truth* to refer to the judgement of relevance.

The evaluation of information retrieval systems using these three parts makes it easy to observe and quantify the comparison of different algorithms, and even the effects of altering algorithm parameters [Clough and Sanderson, 2013].

In addition to the main three parts of a test collection mentioned above, the evaluation of an entity finding task requires a list of possible candidates, which we refer to as the candidate list. It is not always the case that this list is explicitly given in advance. In some cases, as we will see in later chapters, the list may be constructed from the document collection before the experiment. The candidate list is considered to be a vital component of a test collection for an entity finding task, as it can be used to recognise the occurrences of entities in documents. Even when the list is given, it may need to be expanded to include other representations, such as e-mail addresses and employee numbers. This allows for the recognition of occurrences of entities based on a number of different representations.

2.3.1 Test Collections

In this section, we will describe some of the test collections that have been designed and used to evaluate entity-finding applications.

2.3.1.1 W3C

The W3C dataset was developed for the TREC Enterprise Track and was used to evaluate the Expert Search task in 2005 and 2006. The collection was generated from a crawl of a large portion of w3.org[4] sites in June 2004. The data described in Table 2.3 consists of about 330,000 documents in six categories: *web pages*, *source code*, *mailing lists* etc. Some categories have less importance than others because of duplication as in the *www* category, or due to the fact that the data are not useful for a particular task as in the *dev* category.

Table 2.3: W3C collection by category. For each category, the table shows the size in GB, document count, and average document size.

	Description	Size (GB)	No. of Docs	Avg Doc Size (KB)
lists	public e-mails	1.855	198,394	9.8
dev	source code	2.578	62,509	43.2
www	web pages	1.043	45,975	23.8
esw	wiki	0.181	19,605	9.7
other	miscellaneous	0.047	3538	14.1
people	personal homepages	0.003	1016	3.6
all		5.7	331,037	18.1

This test collection includes a list of 1092 candidates defined in terms of their full names and email addresses. Two distinct test sets were used for the Expert Finding task in TREC 2005 and TREC 2006 (Table 2.5). The TREC 2005 test collection included sample queries consisting of the names of the working groups in W3C. The gold standard responses to these queries were assumed to be the names of the individuals that worked in each of these groups.

The TREC 2006 test collection queries were contributed by TREC participants. These queries were judged manually using a set of documents that supported the selection of each candidate as an expert for the given query.

[4]`http://www.w3.org`

```
<top>
<num> EX05 </num>
<title> Hypertext Coordination
</title>
<desc> Hypertext Coordination
</desc>
</top>
```

2005 topic

```
<top>
<num> Number: EX79
<title> Semantic Web </title>
<desc> Description:
Locate individuals with expertise
regarding the Semantic Web.
</desc>
<narr> Narrative:
This topic attempts to locate individuals
with expertise regarding Semantic Web,
especially those who had experience on
design and developing Semantic Web systems.
</narr>
</top>
```

2006 topic

Figure 2.5: Examples of the topics used in TREC expert finding task in 2005 and 2006.

The queries for both test sets follow the TREC topic format [Craswell et al., 2005; Soboroff et al., 2006]. Usually, in TREC terminology, a *query* is also referred to as a *topic*. We use these two terms interchangeably here. A sample query for the TREC 2005 and TREC 2006 W3C test collections is shown in Figure 2.5. Table 2.4 shows the main fields usually present in a TREC topic.

Table 2.4: Main fields in a TREC topic.

Field	Description
top	each topic is contained in *top* tags
num	is the number of the query
title	is a short query. Generally, this field is used as the query topic.
desc	is the description which contains a longer version of the query.
narr	is the narrative part. This part contains more information about the query. Similar to the description part, it may used to expand the search.

Note that the topic title and description are the same in TREC 2005 topics. This gives limited freedom for query expansion.

Table 2.5: Statistics for the W3C test collections.

	TREC 2005	TREC 2006
No. of documents	331,037	331,037
No. of people	1092	1092
Avg. Doc Length	≈700	≈700
No. of queries	50	49
No. of qrels	1509	8351
Avg. Experts/Topic	≈29	≈26

2.3.1.2 CSIRO Enterprise Research Collection (CERC)

The use of names of working groups as queries and the names of the members of these groups as relevant judgements in the W3C test collection is not a very realistic scenario: it is easy to influence the model's performance by targeting documents related the working groups. It is also subject to debate whether the gold standard judgements are always appropriate: an individual may have an administrative role in the group, such as a minute taker, without having any expertise in the field.

These potential problems have been rectified with the introduction of a more realistic dataset which is a crawl of the publicly available web pages of Commonwealth Scientific and Industrial Research Organisation (CSIRO)[5]. This dataset is known as the CSIRO Enterprise Research Collection (CERC) [Bailey et al., 2007a]. This dataset is a medium-sized corpus, slightly larger than the W3C dataset. It consists of 370,715 documents with an average document length of 405.9 terms, which is considerably smaller than the average document length in W3C (i.e., 972.2 terms). See Table 2.6 for the dataset statistics.

The CERC test collection represents some real-world search activity within CSIRO. The topics and the expert judgements were made by internal staff. As with the W3C queries, each query has different parts including the query number,

[5]`http://es.csiro.au/cerc/`

```
<top>
<num> CE-013</num>
<query>human clinical trials</query>
<narr>
Overview of the Human Nutrition Clinic
facility and the latest clinical nutrition
trials which are recruiting volunteers.
</narr>
<page>CSIRO145-08477954</page>
<page>CSIRO145-11110519</page>
<page>CSIRO142-02910122</page>
</top>
```

2007 topic

```
<top>
<num> CE-055</num>
<query>case moth identification</query>
<narr>
I have photographs of several different
Case Moths and have been endeavouring
to find photographs or drawings that could
assist in identification.
</narr>
</top>
```

2008 topic

Figure 2.6: Examples of the topics used in the TREC expert finding task in 2007 and 2008.

query terms, and a narrative which provides extra detail about the information-need for the query. The sample queries in TREC 2007 are accompanied by some examples of web pages judged to be related to them. In TREC 2008, the queries were composed from requests for information received by enquiries staff at CSIRO. These queries were extracted from a log of real email enquiries. Selected emails were used after removing any identifying information and greetings, etc. Figure 2.6 shows some examples of topics from the TREC 2007 and TREC 2008 Enterprise Track.

The CERC judgements should be more accurate than those of TREC 2005 and 2006, because they were made with the benefit of first-hand knowledge. Furthermore, the set of relevant experts for each query is small compared to W3C[6]. This simulates reality, as users are usually interested to see the main contacts for their query rather than a long list of people with various levels of knowledge about the topic [Woudstra and van den Hooff, 2008].

[6]W3C has an average of 40 experts per query.

Table 2.6: Statistics for the CERC Enterprise research test collections.

	TREC 2007	TREC 2008
No. of documents	370,715	370,715
No. of people	3480	3480
Avg. Doc Length	354.8	354.8
No. of queries	50	55
No. of qrels	152	2710
Avg. Experts/Topic	3.06	10.38

2.3.1.3 UvT

The "UvT" collection was developed using public data about employees of Tilburg University (UvT) in the Netherlands. It is based on the Webwijs (Webwise) system. The UvT collection contains documents in both English and Dutch[7]. The collection includes information about 1,168 experts. The document collection includes material from different sources such as home pages, research descriptions, course descriptions, and publications. In total, it contains about 38,000 documents.

In addition to being a bilingual collection, there are a number of other differences between it and the other datasets considered here (W3C and CSIRO). First, it contains both structured and unstructured documents. The structured documents contain rich metadata including information such as the document authority and document-author associations. The corpus also covers a broad range of expertise areas which have been collected from heterogeneous information sources. The relevant judgements are provided by University employees themselves, whereby each expert can select expertise areas from a list of 1,491 topics (5.8 topics/expert on average) [Balog et al., 2007].

[7]Although this is a bilingual collection, we have not applied any language-dependent normalisation to the documents

In 2014, an updated version of the collection (which is now called the "TU" collection) was released [Berendsen et al., 2013]. This new version was formed from a new crawl of the site. Furthermore, it includes different combinations of documents, providing 544 courses, 23624 papers, 534 Dutch profiles, 495 English profiles, and 5250 theses. The updated version consists of a total of 30,447 documents. In this version, 2,507 knowledge areas, or topics, have been defined, with five different sets of relevance judgements (GT). These include judgements made by the experts themselves and judgements made by the experts in an assessment experiment. These are described in Table 2.7.

Table 2.7: Relevance Judgement details for the TU Collection. More detailed descriptions can be found in [Berendsen et al., 2013]

ID	Title	No. qrels	No. queries	No. Experts
GT1	Self-selected profiles	4868	1662	761
GT2	Self-selected profiles in assessment experiment	1661	937	239
GT3	Pooled subsets of self-selected profiles	1099	715	239
GT4	Judged system-generated profiles (binary)	2112	1266	239
GT5	Judged system-generated profiles (graded)	2112	1266	239

We need to index these datasets for the experiments we describe later in this thesis. The two datasets (i.e., UvT and TU) have been indexed separately. In both cases, we have included the candidates profiles after removing the relevance-judgements part. If this was not done, then the task would be trivial as all of the profile would be retrieved by the search engine, giving artificially high accuracy and recall figures for any experiment. All other parts of the candidates' profiles were preserved. Figure 2.7 gives an example of a candidate's profile.

```
<person>
  <anr>120413</anr>
  <name>Tessa de Wit LL.M.</name>
  <name>Tessa de Wit</name>
  <name>T. deWit</name>
  <job>PhD student</job>
  <institute>Department of European & International Public Law</institute>
  <faculty>Faculty of Law</faculty>
  <room>Room M 533</room>
  <address>P.O. Box 90153, NL-5000 LE Tilburg, The Netherlands</address>
  <tel>+31 13 466 803</tel>
  <fax>+31 13 466 804</fax>
  <email>T.deWit@uvt.nl</email>

  <expertise>
    <topic id="1548">international law</topic>
    <topic id="1549">international criminal law</topic>
  </expertise>

</person>
```

Figure 2.7: An example of a candidate profile. The part in the grey box was removed from all profiles where it was present.

2.3.1.4 Essex

One of the contributions presented in this thesis is the creation of a new test collection. This dataset consists of data obtained from a 2012 crawl of the publicly accessible pages (*.essex.ac.uk) of the University of Essex [Alarfaj et al., 2012]. The dataset has 446,038 documents, with a total size of approximately 8.4 gigabytes. A variety of documents are represented, including personal pages for the university academics (i.e., candidate experts). These pages include research interests, links to the courses they teach, and a list of their publications. The dataset also includes 3,095 full-text publications which are stored as plain text.

We followed the TREC format when creating the documents. Figure 2.8 shows a sample document. Every document is delimited by `<DOC>` tags, and given a document ID, `<DOCNO>`. The first part of the document representation, under the `<DOCHDR>` tags, contains some general document information including the document URL, crawling date and number of words. Table 2.8 provides a general overview of the Essex dataset.

```
<DOC>
<DOCNO>essex-001-0032031</DOCNO>
<DOCHDR>
URL: http://csee.essex.ac.uk/gig/people/simon-lucas
Date: Tue, 05 Jun 2012 08:22:03 GMT
Connection: close
Content-Type: text/html; charset=iso-8859-1
NumOfWords: 237
</DOCHDR>
<Content>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head><title>Simon M. Lucas | Game Intelligence Group</title></head>
<body class="sidebars">
<h1><a href="/gig/" title="Game Intelligence Group University of Essex">
       <img src="/gig/sites/default/files/garland_logo.png"/>
       Game Intelligence Group</span> University of Essex</a></h1>
<p><img src="http://dces.essex.ac.uk/staff/lucas/CroppedSimon3.PNG" /></p>
<p>
 Simon M. Lucas is a Professor of Computer Science at the
 <a href="www.essex.ac.uk"> University of Essex </a>, Colchester,
 Essex, U.K. His main research interests are in machine learning
 and games. He has published widely in these fields with over <b>
 150 </b> peer-reviewed papers and is the Founding Editor-in-Chief
 of the <i>IEEE Transactions on Computational Intelligence and AI
 in Games </i>.
</p>
 ...
</body>
</html>
</Content>
</DOC>
```

Figure 2.8: Sample document in Essex dataset follows TREC format.

Table 2.8: Characteristics of the Essex collections.

Description	
Number of Documents	446.038
Full text articles	3.095
Unique terms	525.763
Size of collection (GB)	8.4
Average document size (terms)	573.948

Topics and Assessments

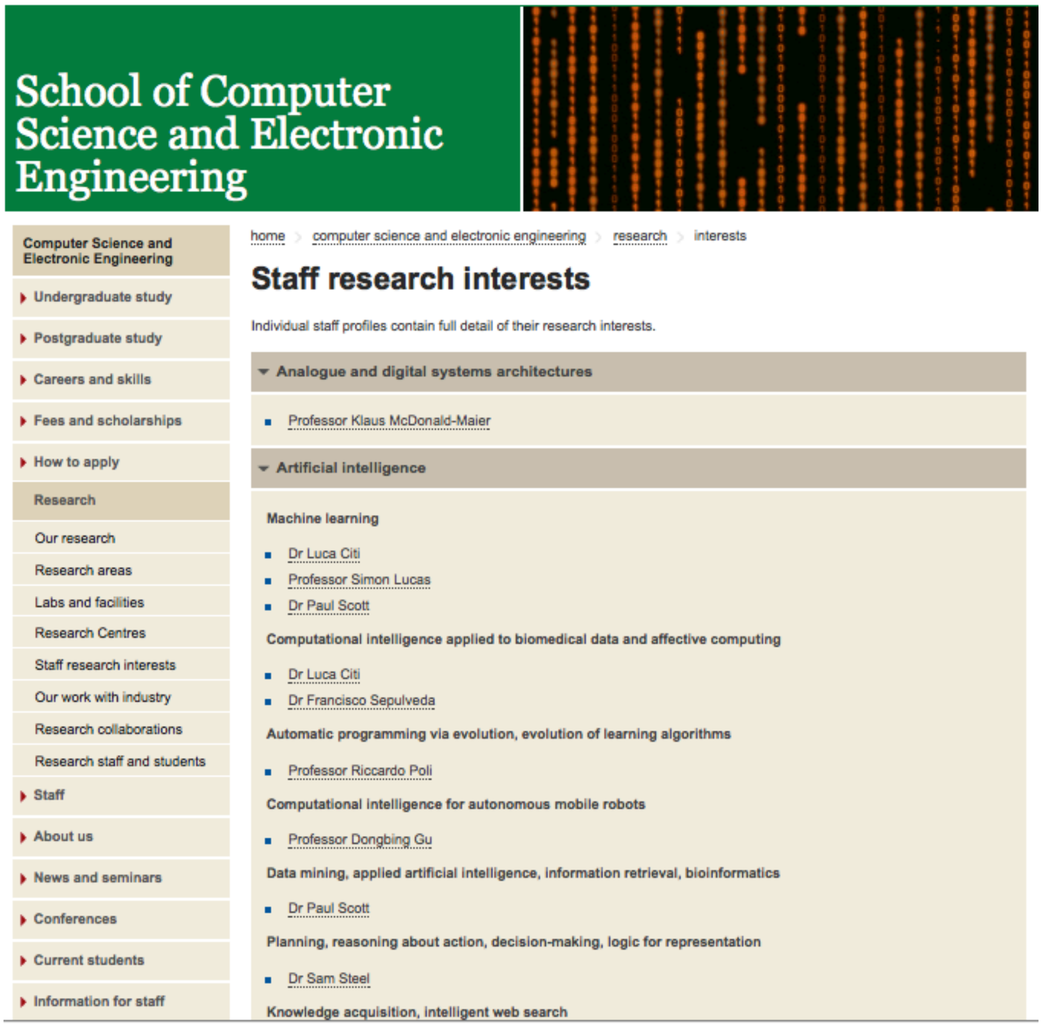
The School of Computer Science and Electronic Engineering at the University of Essex has developed a page called “staff research interests” Figure 2.9. This page links research topics with the corresponding academics. We take this mapping as a complete set of correct matches. Thus, the page is used to compose the queries and relevant judgements for this experiment. For instance, the research topic “Machine Learning” was used as a query and “Dr Citi, Prof. Lucas, and Dr. Scott” as the relevance judgement for this query.

We have made some simplifying assumptions here. In particular, we have assumed that the pages are up to date, and that the topics correctly match the experts. These assumptions are not always correct, but seem to be sensible in this scenario. They seem like reasonable assumptions because one can assume that the research interests described by a staff member provide a good reflection of the research topics that they are interested in. It is evident that one cannot guarantee that these pages are always properly maintained and that all staff members are currently present. The application of these simplifying assumptions helped to prevent any manual involvement when creating the relevance judgement.

On this page, there are 371 topics (i.e. potential queries) divided among 28 general research areas. Table 2.9 shows the distribution of the queries in these 28 research areas. Each query is associated with one or more of the school’s academic staff (see Figure 2.9). It is assumed that these names belong to experts on the corresponding topics. The average number of experts for each query is 1.30.

Table 2.9: Distribution of query terms - N is the number of queries for the corresponding research area.

Research area	N
Analogue & digital systems architectures	2
Artificial intelligence	26
Audio	12
Brain computer interfaces	18
Computational finance, economics & management	1
Computational intelligence	10
Data communications & networking	83
Educational technology	6
E-Learning	2
Embedded systems	12
Human-computer interfaces	4
Intelligent inhabited environments	11
Mathematics, statistics, numerical methods	7
Mixed reality	2
Natural & evolutionary computation	15
Natural language engineering	11
Optical & semiconductor devices	9
Optimisation & constraint satisfaction	6
Radio, radar, electromagnetics	15
Robotics	22
Semiconductors: theory & experiment	8
Signal processing	12
Software agents	13
Software engineering	18
The design & construction of ultrafast systems for terahertz studies	2
Theoretical computer science	19
Hz spectroscopy of molecules	2
Video, image processing and computer vision	23



Staff research interests

Individual staff profiles contain full detail of their research interests.

Analogue and digital systems architectures

- Professor Klaus McDonald-Maier

Artificial intelligence

Machine learning

- Dr Luca Citi
- Professor Simon Lucas
- Dr Paul Scott

Computational intelligence applied to biomedical data and affective computing

- Dr Luca Citi
- Dr Francisco Sepulveda

Automatic programming via evolution, evolution of learning algorithms

- Professor Riccardo Poli

Computational intelligence for autonomous mobile robots

- Professor Dongbing Gu

Data mining, applied artificial intelligence, information retrieval, bioinformatics

- Dr Paul Scott

Planning, reasoning about action, decision-making, logic for representation

- Dr Sam Steel

Knowledge acquisition, intelligent web search

Figure 2.9: Staff research interests at the School of Computer Science and Electronic Engineering at the University of Essex.

2.3.1.5 ClueWeb09

The test collections introduced so far are intended to be representative of enterprise settings. In contrast, ClueWeb09 is intended to represent a more general web-based scenario. It was created by the Language Technology Institute at Carnegie Mellon University to support research on information retrieval and related human language technologies. The dataset consists of about 1 billion web pages in ten languages. The pages were collected in January and February 2009.

```
<query>
<num>1</num>
<entity_name>Blackberry</entity_name>
<entity_URL>clueweb09-en0004-50-39593</entity_URL>
<target_entity>organization</target_entity>
<narrative>Carriers that Blackberry makes phones for.</narrative>
</query>
```

Figure 2.10: Example of an Entity Track topic (TREC 2009).

This dataset was used by several tracks of the TREC conference including the Web Track, Session Track, and Entity Track. [8]

A subset of this collection, known as "Category B" is used in the TREC Entity Track. It consists of approximately 50 million English pages [Balog et al., 2009b]. Table 2.10 describes the test collection. The task is called Related Entity Finding (REF), and can be described as finding entities that have a specific relationship with the query entity. Figure 2.10 shows an example topic used in TREC 2009.

The main focus of the TREC topic is on the "entity_name", that is, the name of the entity for which the user is searching. The document ID of its homepage is indicated by the "entity_URL". The type of the required entities is represented in the "target_entity" element. In TREC 2009, the types were limited to "organisation", "person", or "product". Finally, the "narrative" is a free text element that embodies the nature of the relationship between the given entity and the target entities. In the example in Figure 2.10, we are looking for "Carriers that Blackberry makes phones for". The result set should include a list of organisations such as "Virgin_Mobile", "Etisalat" and "Vodafone".

[8]ClueWeb12 is a successor to the ClueWeb09 web dataset (see `http://boston.lti.cs.cmu.edu/clueweb12/`). However, much Entity Finding research, including Entity Track 2009 and 2010 uses ClueWeb09. Consequently, to ensure comparability of our experiments with other work, we use the older version.

Table 2.10: Statistics for the ClueWeb09 test collections.

	ClueWeb09
No. of docs	50 million
Avg. Doc Length	1420.85
No. of queries	20
No. of qrels	13,488
Avg. No. of Entity in a Topic	17

2.3.2 Evaluation Metrics

This section will introduce some of the evaluation metrics used in information retrieval tasks, in general, and entity finding, in particular.

The two prominent and most commonly used evaluation metrics in IR applications are those of *precision* and *recall* [Sanderson, 2010]. Here, the *precision*, refers to the proportion of a retrieved set that is relevant. The *recall* refers to the proportion of the relevant documents from the collection which are included in the retrieved set. For clarification, Cleverdon and Keen [1966], produced a contingency table, reproduced here as Table 2.11, that shows the possible outcomes for the information items.

Table 2.11: Contingency table for information items as presented by Cleverdon and Keen [1966].

	Relevant	Not Relevant	
Retrieved	a	b	$a+b$
Not Retrieved	c	d	$c+d$
	$a+c$	$b+d$	Total Collection

$$\text{Precision} = \frac{a}{a+b} = \frac{\text{number of relevant items retrieved}}{\text{number of retrieved items}}$$

$$\text{Recall} = \frac{a}{a+c} = \frac{\text{number of relevant items retrieved}}{\text{number of relevant items}}$$

According to Singhal [2001], researchers have used several variants of the *precision* and *recall* metrics to evaluate a ranked set of retrieved results. Some of these variants that are commonly used to evaluate entity finding systems include $P@n$, R-precision, Average Precision (AP), Mean Average Precision (MAP), bpref, and MRR. We describe these variations below.

P@n stands for precision at n where n can be any recall level. This metric is determined by the formula

$$P@n = r(n)/n,$$

where $r(n)$ refers to the number of relevant information items included in the top n retrieved items. The metric $P@n$ measures how well the results at the top of the retrieved list match the query. Its development was based on the assumption that users are only interested in the first couple of pages of the results [Baeza-Yates and Ribeiro-Neto, 2011]. In other words, it measures precision at fixed low levels of retrieved results, such as five or ten candidates. One of the advantages of using this metric is that it does not require any estimate of the size of the relevant set. However, this measure shows less stability when the size of the collection is changed, compared with other evaluation measures as examined by Hawking and Robertson [2003]. Another issue with this measure is that $P@n$ will always be less than one for topics with fewer than n relevant documents in the collection.

R-precision is precision at level R, where R is the total number of relevant items for a particular topic [Voorhees and Harman, 2005]. R-Precision is similar to $P@n$ in the sense that they both calculate the precision at a cut off point. The difference with R-Precision is that the cut off is not fixed across topics. This means that, unlike $P@n$, R-Precision always allows systems to achieve the maximum score of 1 if the top items are relevant.

Average Precision (AP) is one of the standard measures used in TREC tasks. It provides one single figure across all recall levels. AP calculates the precision at the rank position of each relevant item, and then takes the average. For a given query, assume that N is the number of retrieved items and r is the number of relevant items. Then AP is calculated as follows:

$$AP = \frac{1}{r} \sum_{E=1}^{N} P(E) \times rel(E),$$

where $P(E)$ is the precision at position E and $rel(E)$ is an indicator function that returns 1 if the information item at position E is relevant, and returns 0 otherwise.

It has been noted by many researchers [Tague-Sutcliffe and Blustein, 1994; Voorhees and Harman, 1998] that there is a high correlation between average precision and $R - precision$. According to Buckley and Voorhees [2000], this is remarkable, given that $R - precision$ is evaluated at exactly one point in a retrieval ranking, whereas Average Precision represents the entire area underneath the recall-precision curve. Aslam et al. [2005] gives a geometric interpretation of $R - precision$, showing that, under a very reasonable set of assumptions, $R - precision$ also approximates the area under the precision-recall curve. This explains the high correlation between the two measures.

Mean Average Precision (MAP) is the mean of the AP for all of the queries in a set,

$$MAP = \frac{1}{Q}\sum_{i=1}^{Q} AP_i \ , \qquad \text{where Q is the number of queries.}$$

This measure is commonly used to evaluate the overall retrieval performance due to its discrimination and stability [Manning et al., 2008]. According to Sanderson and Zobel [2005], for a predefined relevance judgement, MAP is more reliable than other measures.

For very large collections, it can be challenging to evaluate all documents arising from a query in order to generate the relevance judgements. This problem motivates the development of some evaluation measures that do not rely on a complete set of relevance judgements.

bpref is one of these measures which is designed to work even with incomplete relevance judgements [Buckley and Voorhees, 2004]. It evaluates an ordered result set to find whether relevant items are ranked above irrelevant ones. Using this measure, the experimental outcome with incomplete judgements would be similar to the outcome for Mean Average Precision (MAP) with complete judgements [Yilmaz and Aslam, 2006]. *bpref* is calculated using the set of relevant items, r, as follows:

$$bpref = \frac{1}{r}\sum_{r_i}\left(1 - \frac{\text{number of } n \text{ above } r_i}{r}\right),$$

where r_i is a relevant item and n is a non-relevant item.

Mean reciprocal rank (MRR) is used to evaluate the result set, based on the location of the first relevant item. The reciprocal rank is the inverse position of the first relevant item in the retrieved set. MRR takes the average of the reciprocal ranks over a set of queries [Kantor and Voorhees, 2000; Voorhees, 1999]. It is, therefore, very sensitive to the position of the first relevant item. For a set of queries Q, the MRR would be calculated as follows:

$$MRR = \frac{1}{|Q|} \sum_{q \in Q} \frac{1}{r_q},$$

where r_q is the topmost rank at which an item relevant to the query q is found.

2.3.3 Expert finding task at TREC

As discussed in Section 2.2, the TREC conference series has helped to promote interest in the expert finding task. Many approaches to this task were developed during the 2005 to 2008 Enterprise Tracks [Bailey et al., 2007b; Balog et al., 2008; Craswell et al., 2005; Soboroff et al., 2006]. In this section, we will report the top results of the four TREC expert-finding competitions as a point of reference. Later in the thesis, we shall discuss some experiments which we have conducted using the same datasets and test collections. Presenting the TREC results will serve as benchmarks our own experiments.

The first year in which expert-finding featured at TREC was 2005. The W3C dataset and test collection, (Section 2.3.1.1), was used. The competition task was defined as follows: “Given a topical query, find a list of W3C people who are experts in that topic area. Finding people, not documents, based on analysis of the entire W3C corpus”. Ten systems were submitted for this task. The 2005 expert finding track results are given in Table A.1. The run with the highest MAP score of 0.2749 was submitted by Fu et al. [2005], who proposed a new method to solve the expert finding problem called "document reorganisation". Their method incorporates three main steps to rank experts:

1. Mark up the appearance of all candidates in the given corpus
2. Extract descriptions for candidates using different rules according to their appearances in the corpus and the document type
3. Combine these descriptions to obtain a compilation of descriptions for each expert

In the following year (2006), the W3C collection was again used for the expert finding track. The competition attracted more interest, with 91 runs submitted by 23 groups. The main difference from the previous year was that the track participants contributed to the creation of the topics and relevance judgements. The 2006 expert finding track results are given Table A.2. The highest score obtained by a run that did not consider supporting documents was 0.6431, while the highest score obtained by a run that did consider these documents was 0.4421. Both of these runs were submitted by Zhu et al. [2006]. Their work used a two-stage language model with incremental window sizes to capture the association between query terms and candidate experts.

A new dataset was introduced for the TREC expert finding in 2007. The CSIRO dataset, (Section 2.3.1.2), was created to offer a more realistic setting for the task. Unlike the W3C collection, the CSIRO dataset did not contain a canonical candidate list. Therefore, participants were required to construct their own candidate lists from the data. This made the task significantly more complicated. A total of 55 runs were submitted by the 16 groups participating in the competition. The 2007 expert finding track results are given in Table A.3. The highest MAP score obtained by a run was 0.4632. This was submitted by the Tsinghua group [Fu et al., 2007b], who employed a candidate centric model by building personal description documents (PDD) for each candidate.

The fourth and final year of the TREC enterprise track (2008) used the same document collection as TREC 2007. Topics for TREC 2008 were extracted from a log of real email enquiries. The main difference from the TREC 2007 track was that the results were judged by participants. Eleven groups submitted a total of 42 expert finding runs for the 2008 TREC enterprise track. The best results for each group are given in Table A.4. The best performing run by Balog and de Rijke Balog and de Rijke [2008a] employed a proximity-based version of the candidate-based model and the document-based model to obtain a MAP score of 0.4490. Moreover, they enhanced their results by applying a profile-based query expansion method.

The fourth and final year of the TREC enterprise track (2008) used the same document collection as TREC 2007. Topics for TREC 2008 were extracted from a log of real email enquiries. The main difference from the TREC 2007 track was that the results were judged by participants. Eleven groups have submitted a total of 42 expert finding runs for the 2008 TREC enterprise track. The best results for each group are given in Table A.4. The best performing run by Balog and de Rijke [2008a] employed a proximity-based version of the candidate-based model and the document-based model and score a MAP of 0.4490. Moreover, they enhanced their results by applying a profile-based query expansion method.

2.4 Proximity Search

A very primitive form of expert finding treats all words in a document with equal importance, and assumes that the order of words has no significance. This is known in IR as the bag-of-words assumption. This assumption implies that all candidates mentioned in a document are equally associated with all of the topics discussed in the same document. This assumption may hold if the document has a single author only. However, in enterprise and Web domains, documents are often written by multiple authors (e.g., technical reports, newsletters, Wikipedia pages). Moreover, it would be difficult to apply this assumption to other types of entities such as products and organisations. This motivates the development of a new technique called *proximity search*, which captures this dependency more explicitly. The proximity search representation uses positional information to improve the retrieval performance by taking the proximity between the occurrences of entities and the terms of the query into consideration.

Approaches that incorporate a notion of proximity appear to be among the most successful models developed for the entity-finding task. For instance, the work of Petkova and Croft [2007] directly addresses the use of different *kernel* functions that capture the proximity of candidates and query occurrences. Other work that examines proximity in expert finding includes the findings of Fu et al. [2007a]; Macdonald et al. [2008]; Petkova and Croft [2008].

It has been noticed that many of the aforementioned models show noticeable improvements when applying a proximity search (see Table 2.12). Proximity is defined in terms of parts of the document (i.e., windows of text). The size of the window is defined by the distance between the candidate's evidence and the query.

Table 2.12: Some examples of the improvements reported in the literature when using proximity search.

Method	MAP	Dataset	Ref
Model 2 Proxim. Kernels (Gaussian k)	0.3571 0.6193 (+73%)	W3C TREC 2006	[Petkova and Croft, 2007]
Voting model Voting model + proximity	0.3519 0.4319 (+23%)	CSIRO TREC 2007	[Macdonald et al., 2008]
Model 1 Model 1B (proximity)	0.4518 0.5178 (+14%)	CSIRO TREC 2007	[Balog and de Rijke, 2008b]

In the best run submitted for TREC 2005, [Fu et al., 2005], a window of ten words before and ten words after the occurrence of the candidate name was used. The second best performing system in 2005, [Cao et al., 2005], tested the performance of the window-based model with different sizes of windows. They also compared their results with those found by using the whole document as a window (i.e., a document-based model). A performance improvement of about 10% over the document-based model was shown when a window-based model with a window size of fifty words was used.

At TREC 2006, Zhu et al. [2006] defined a span query co-occurrence model in which query terms and candidate evidence were required to co-occur within the text window. They defined ten incremental text window sizes, i.e., sizes of 10, 28, 48, 88, 160, 280, 360, 660, 1200, and 3200 words. Their work compared the use of a single window and multiple windows. In the multiple window solution, windows of different lengths were each given different weightings, and were simultaneously applied in the search. The findings showed that using multiple windows helps to increase the final performance of the system. They also showed that applying the multiple window solution has the advantage of identifying more correct results at the top of the retrieved list.

The second best system from TREC 2006 was developed by Bao et al. [2008]. They defined five different span relations within which to search for queries and candidate entities as follows:

Section relations: where the queried query and the candidate entities occur in the same section.

Windowed section relations: where the query and the candidate entities occur within a fixed window of text within a section. Bao et al. [2008] used a window of 200 words in their work.

Reference section relations: where some sections should be treated specially like a list of books or authors.

Title-author relations: where the query appears in the *Title* and the candidate entity appears in the *Author*.

Title-body relations: where the query appears in the *Title* and the candidate entity appears in the *Body* of the same section.

Zhu et al. [2009] used a novel weighted multiple-window approach for expert association discovery. Their work investigated the effects of using different fixed window sizes. They discovered a direct correlation between the increasing the size of the window and improvement in the MAP results until the MAP reached a rather stable level at a window size of around 200 words.

More recently, Vechtomova and Robertson used a fixed window size of 100 words [Vechtomova and Robertson, 2012], claiming that a span of 100 words will capture broader contextual associations between words rather than specific lexico-syntactic relationships. Many researchers assert that a query has some relevance for, or influence over, its neighbouring text [Zhao et al., 2011]. It is also assumed that this relevance diminishes with distance.

De Kretser and Moffat [1999] suggested that the influence from each occurrence of a query is additive. The contributions of each occurrence of each query can be summed to arrive at a similarity score for any particular location in the document. This impact may be characterised by the use of kernel functions. We could use arbitrary functions, but there seem to be certain sensible properties that a kernel function, $k(i, j)$, should have, including monotonicity and symmetry. Kernel functions must satisfy a number of properties. The following list identifies some of the properties that are important for proximity search:

non-negative The output of the kernel is always non-negative

symmetric The output should depend only on the distance between the two entities, and not on the direction in which the distance is measured: $k(i, j) = k(j, i)$

monotonic The influence of the kernel function should decrease with an increase in $|j - i|$

position-independent The influence of the kernel function should be the same when i and j are increased by an equal amount, $k(i, j) = k(i+x, j+x)$

Lv and Zhai [2009] use a number of proximity-based density functions (i.e., kernels) in their work. We describe some of these here.

In the following kernel functions, the parameters i and j represent the indices corresponding to the positions of the occurrences of the query and the candidate in the document. The lexical distance between the query and candidate is then given by the difference between the indices[9]. All of the following kernels have one parameter σ to tune. This parameter controls the spread of kernel curves, or how quickly the curve tails off. In general, the optimal setting for σ may

[9]There may be complications in the application of these functions if the query and candidate consist of more than one word. We consider this later.

vary according to the document, query, or entity being searched. However, for simplicity, it is usually assumed that σ is set to a constant value. For instance, Petkova and Croft [2007] set the σ value equal to 80, while Lu et al. [2013] use $\sigma = 300$.

Gaussian Kernel

$$k_G(i,j) = \exp\left[\frac{-(i-j)^2}{2\sigma^2}\right]$$

Triangle Kernel

$$k_T(i,j) = \begin{cases} 1 - \frac{|i-j|}{\sigma} & \text{if } |i-j| \leq \sigma \\ 0 & \text{otherwise} \end{cases}$$

Circle Kernel

$$k_C(i,j) = \begin{cases} \sqrt{1 - \left(\frac{|i-j|}{\sigma}\right)^2} & \text{if } |i-j| \leq \sigma \\ 0 & \text{otherwise} \end{cases}$$

Cosine (Hamming) Kernel

$$k_H(i,j) = \begin{cases} \frac{1}{2}\left[1 + \cos\left(\frac{|i-j| \cdot \pi}{\sigma}\right)\right] & \text{if } |i-j| \leq \sigma \\ 0 & \text{otherwise} \end{cases}$$

Petkova and Croft [2007] also considered a **Step function** defined as follows:

$$k_S(i,j) = \sum_{i=1}^{m} \alpha_j \dot{I}_{A_j}\left(|i-j|\right), \text{ where } \dot{I}_A(x) = \begin{cases} 1 & \text{if } x \in A \\ 0 & \text{otherwise.} \end{cases}$$

Here A is a sequence of m intervals with weight α and $\dot{I}_A$ is an indicator function that returns one if its argument falls within the given interval and zero otherwise. Zhao et al. [2011] introduce more kernel functions that satisfy the kernel function properties mentioned above:

Quartic Kernel:

$$k_Q(i,j) = \begin{cases} 1 - \left(\left(\frac{|i-j|}{\sigma}\right)^2\right)^2 & \text{if } |i-j| \leq \sigma \\ 0 & \text{otherwise} \end{cases}$$

Epanechnikov Kernel:

$$k_E(i,j) = \begin{cases} 1 - \left(\frac{|i-j|}{\sigma}\right)^2 & \text{if } |i-j| \leq \sigma \\ 0 & \text{otherwise} \end{cases}$$

Triweight Kernel:

$$k_{TW}(i,j) = \begin{cases} 1 - \left(\left(\frac{|i-j|}{\sigma}\right)^2\right)^3 & \text{if } |i-j| \leq \sigma \\ 0 & \text{otherwise} \end{cases}$$

In Figure 2.11 we provide the distance versus weight graphs for the aforementioned functions.

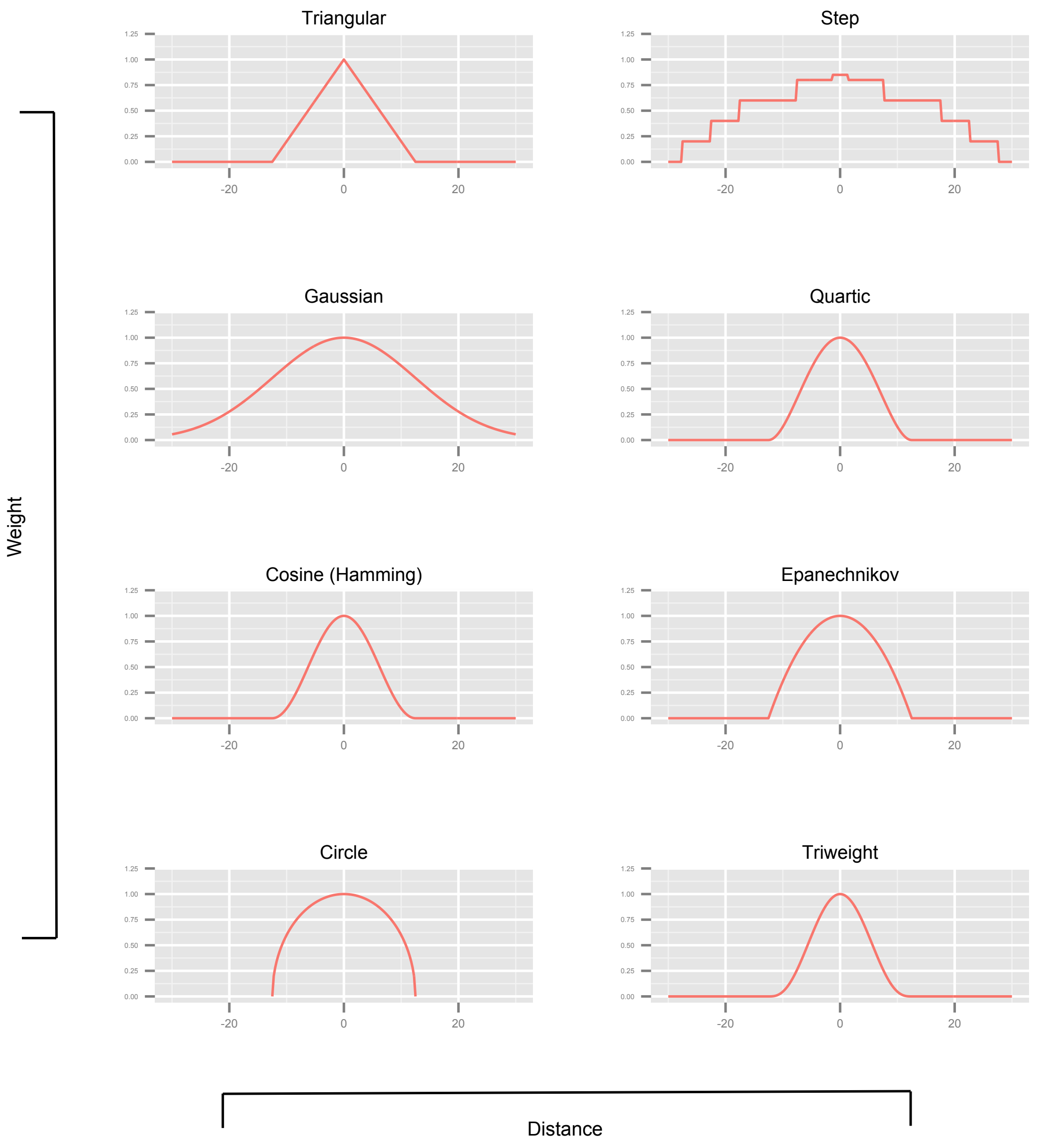


Figure 2.11: Illustrations of weight versus distance for a number of Kernel density functions.

The aforementioned kernels incorporate a number of different ways in which the proximity between the query and candidate entities can be captured. In our pilot study (see Section 5.1), we evaluated a number of these proximity functions.

Note that symmetric kernels that incorporate functions of i and j will either take the modulus or the square of their difference. This corresponds to a notion of "distance" (if the indices are assumed to be ordered). We can generalise these kernel functions by reformulating them in terms of a general notion of distance, rather than indices. If we assume that Δ represents the distance between two terms, then we may rewrite the function definitions by replacing $k(i,j)$ by $k(\Delta)$ and $|i-j|$ by Δ. Assuming that the distance Δ between two terms t_1 and t_2 is given by $\delta(t_1,t_2)$, then $k(\delta(t_1,t_2))$ would give the evaluation of the kernel function for the two terms t_1 and t_2. This allows an elegant generalisation of the kernels to work with multi-word queries and multi-word entities[10].

2.5 Summary

This chapter has discussed several relevant research advances in the area of entity and expert finding. Some of these advances, such as techniques for document ranking and candidate frequency calculation have been incorporated into our approach. Many of the relevant approaches discussed in this chapter depend heavily on different heuristics.

The approach proposed in this thesis aims to improve on the state-of-the-art approaches by enhancing the proximity function for the ranking of candidate entities. We propose to adaptively select the size of the context window in order to boost the retrieval scores for the entities that are close to the query terms. Consequently, the size of the window will not be fixed for all documents, but, rather, will be dependent upon the features of the document under examination. In order to select sizes for such windows, we will utilise a number of the document features. Our suggested method can be distinguished by its ability to enhance the results by applying a notion of proximity with little dependence on document

[10]Abstracting away from indices to distances also has the potential to allow measures of distance other than just the number of words (e.g. number of characters, graphemes, morphemes, constituent lexemes, number of words excluding clitics, or some other measure of syntactic or semantic distance). We do not explore this in the current thesis, but such alternative measures may turn out to be appropriate for some languages and applications.

types. This will enable it to be applied in different environments with only minimal tuning.

In the following chapter, we will discuss the formalisation of our entity finding task. We shall formally define how candidate entities are associated with the user query. To this end, we will introduce two main models, namely, frequency-based and proximity-based associations.

3 Formalisation

In this chapter, we focus on formalising the entity finding task (providing a list of relevant entities in response to a given query). In order to model this task, the association between the candidate entities and the query is formally defined. In this work, we use two forms of associations: frequency and proximity based associations. We employ a number of models to estimate the degree of relevance of a candidate based on these two forms of associations.

3.1 Introduction

The goal of entity finding applications is to retrieve the entities which are most relevant for a given query. In a document collection, there may be many possible candidate entities that have some relationship with the query. The task is to identify which of these candidate entities are most likely to be relevant.

It is possible to formally state this problem as follows: what is the degree of *relevance* of a candidate entity c to the query q? Specifically, we aim to determine the relevance $p(c|q)$, and rank candidate entities accordingly. The estimation of the relevance of candidate entities lies at the core of any entity finding application. We employ the two-stage model proposed by Cao et al. [2005] to estimate entity relevance in our models.

It is possible to view documents as lists of words. As was discussed for kernel functions in Section 2.4, the positions of words in this list could subsequently be used to express proximity between terms. This may not always be appropriate, as in the case when we wish to consider multi-word queries and candidates. Instead, we introduce variables $t_q \in d$ and $t_c \in d$ to represent some instance (token) of the query q or the candidate c in the document d. We then introduce a function $\delta(t_1, t_2)$ to represent the distance (i.e. number of words) between instances of terms t_1 and t_2, as discussed in Section 2.4.

Now, we can determine the relevance of a candidate entity using the two-stage model [Cao et al., 2005] as follows:

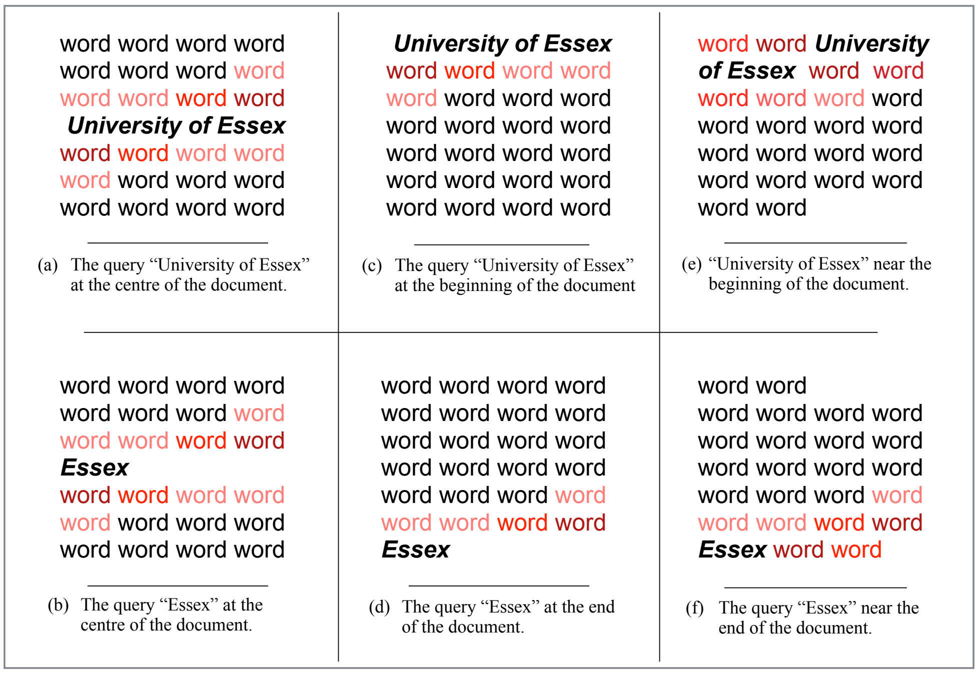
$$p(c|q) = \sum_d p(c, d|q) = \sum_d p(c|d, q) \cdot p(d|q). \tag{3.1}$$

The first term, $p(c|d, q)$, measures the relevance of a candidate entity c to the query q when only document d is considered. This relevance is then weighted by the relevance, $p(d|q)$, of document d to the query. This will bestow more importance on candidate entities mentioned in highly relevant documents.

Whenever a query is mentioned within a document, an adaptive window will be placed around it. Half of the window will be located to the left of the query and half to the right of the query. This process is performed regardless of the query size. For example, if the query is composed of one word only (e.g., “Essex”) and the size of the adaptive window is ten words, then five words will be considered on either side of the query. Similarly, if the query is composed of more than one word, then five words will be considered to the left of the first word of the query and five words will be considered to the right of the last word of the query. For example, if the query is “University of Essex”, then five words will considered to the left of the word “University” and five words will be considered to the right of the word “Essex”.

In some cases, the query might be located towards the beginning or the end of a document. In such instances, part of the window could extend beyond the document limits. In such cases, the part of the window that extends outside the document will be discarded. This should not affect the ranking of the entities located on the other side. Figure 1 shows some examples in which the window is located at different places in the document. In examples 3.1-(c) and 3.1-(d), only half of the window is accommodated within the document, whereas in examples 3.1-(e) and 3.1-(f), half of the window is located inside the document, but only two words from the other half are located inside the document, and so we must discard the three remaining words.

Our approach also allows for the candidate evidence to consist of more than one word. In the case of multi-word evidence, the weighting function considers the part of the evidence that is closest to the query and scores the entity correspondingly.



(a) The query "University of Essex" at the centre of the document.

(c) The query "University of Essex" at the beginning of the document

(e) "University of Essex" near the beginning of the document.

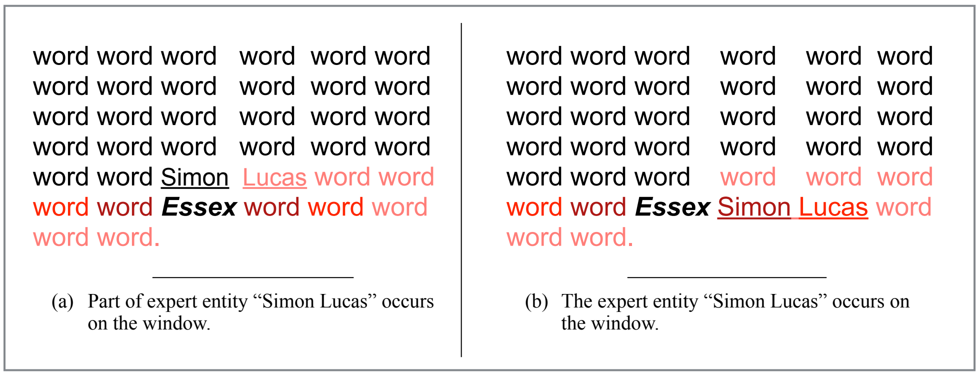
(b) The query "Essex" at the centre of the document.

(d) The query "Essex" at the end of the document.

(f) The query "Essex" near the end of the document.

Figure 3.1: A window of ten words placed around the queries "University of Essex" and "Essex" at different places in the document. The query is shown in bold face, while the window is marked with a different colour.

In example 3.2-(a), the score for the entity is based on the location of the word "Lucas" even though the word "Simon" is located outside of the window. Similarly, in example 3.2-(b), the score for the entity is based on the location of the word "Simon" because it has the higher weight.


word word word word word word
word word Simon Lucas word word
word word Essex word word word
word word.
word word word word word word
word word word word word word
word word Essex Simon Lucas word
word word.

(a) Part of expert entity "Simon Lucas" occurs on the window.

(b) The expert entity "Simon Lucas" occurs on the window.

Figure 3.2: Examples of multi-word evidence occurring at different places in the window.

3.2 Entity evidence

When searching for an entity, we are, in fact, scanning the text for something that will allow us to identify this entity. We call this identifying information *entity evidence*. The entity evidence found in a text can take on a number of forms including a name, an email, or an ID number. As an entity may have different descriptions, it will have different pieces of evidence or references based upon these descriptions associated with it. Therefore, any entity evidence we encounter in a text is, in fact, a reference to the actual entity under some description.

Figure 3.3 provides an example of an entity and some of the evidence relating to it. The entity used in this example is a person. This entity has a number of different descriptions. For instance, he is an employee at the *University of Essex*, where he has a position, an email address, and an office number. The evidence that we may find in the text under this description is the position "Head of the School", the email address "Simon@essex.ac.uk", and the office number"NL1-11". Other descriptions could include being a UK citizen, or a member of some particular group.

Thus, it is important to be aware that an entity may have different descriptions, and that there are different ways of referring to the same entity. For simplicity, in this thesis, we will use the word "entity" to refer to all the different forms that evidence might take for a given entity.

The extraction of such evidence is an important task as every entity is evaluated using the aggregation of its evidence. It is generally assumed that entities with more evidence in the relevant text are more relevant to the query, and thus should be ranked more highly. Another important assumption is that entities with evidence that appears very close to a query instance in the text are also considered relevant. Therefore, failing to recognise some evidence for an entity will

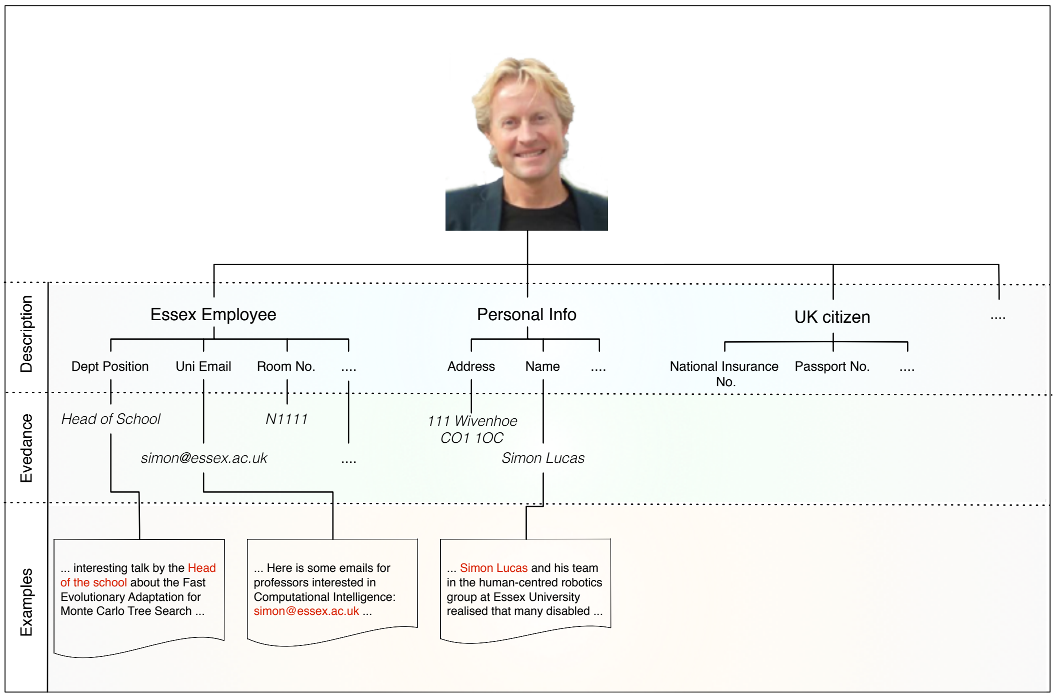

Description
Evedance
Examples
Essex Employee
Personal Info
UK citizen
....
Dept Position
Uni Email
Room No.
....
Address
Name
....
National Insurance No.
Passport No.
....
Head of School
N1111
simon@essex.ac.uk
....
111 Wivenhoe CO1 1OC
Simon Lucas
... interesting talk by the Head of the school about the Fast Evolutionary Adaptation for Monte Carlo Tree Search ...
... Here is some emails for professors interested in Computational Intelligence: simon@essex.ac.uk ...
... Simon Lucas and his team in the human-centred robotics group at Essex University realised that many disabled ...

Figure 3.3: Simon Lucas is a professor at the University of Essex. This figure highlights some of his descriptions and evidence and some examples of these in the sample text.

result in an erroneously low score for this entity and, subsequently, an inaccurate ranking of entities.

3.3 Frequency-based model

In entity finding applications, we are most interested in the value of the relevance, $p(c|d, q)$. There are many ways to calculate this relevance. A basic, but effective, approach is to rank candidate entities by how many times they are mentioned in the top-ranked documents. Using a frequency-based model (FBM) with a TF-IDF weighting scheme, this model estimates a candidate's importance in a particular document, based on the candidate's overall importance. Moreover, it discriminates against those candidates whose evidence appears in all documents by the inclusion of the log-term in Equation 3.2 below [Balog et al., 2009a]. In this model, the candidate's importance in the document d is given by

$$P_{frequency}(c|d,q) = \frac{n(c,d)}{\sum_{c'} n(c'd)} \cdot \log \frac{|\mathscr{D}|}{|\{d' : n(c,d') > 0\}|}, \tag{3.2}$$

where $n(c,d)$, the number of pieces of evidence found for the candidate c in document d is given by:

$$n(c,d) \triangleq \Big|\{t_c | t_c \in d\}\Big|.$$

Here, $|\mathscr{D}|$ denotes the total number of documents in the collection, and the expression $|\{d' : n(c,d') > 0\}|$ is used to indicate the number of documents in which the candidate evidence has been found.

This model has formed the basis for many expert/entity finding approaches including [Forst et al., 2007; Ru et al., 2005; Shen et al., 2008].

3.4 Proximity-based model

As discussed in Section 2.4, to establish the relevance level of a candidate entity, it is not only necessary to determine the number of times it appears in the retrieved documents, but also to ascertain how closely these appearances relate to the query (we call this the *candidate proximity*). With this in mind, we can calculate the candidate relevance, $p(c|d,q)$, from Equation 3.1, using the candidate proximity rather than the candidate frequency. To measure the candidate proximity, we use kernel functions[1]. Candidate entities can be ranked, based on the normalised score of their total proximity $t_p()$ as in Equation 3.3:

$$P_{proximity}(c|d,q) = \frac{t_p(c,Q,d)}{\sum t_p(c',Q,d)}. \tag{3.3}$$

[1]The kernel function $k(i,j)$ could be any of the functions defined on pages 63.

For each candidate entity c in the document, we calculate the total proximity $t_p(c)$ by aggregating the proximity related to each query. Here we take a full query (e.g., “machine learning”) and treat it as a single unit so that every time we find a match for the full query, we impose a proximity window on it. The resulting value of $t_p(c, Q, d)$ represents the total proximity value for the candidate c in the document d. The total proximity t_p is then calculated as follows:

$$t_p(c, Q, d) = \sum_{t_q \in d} \sum_{t_c \in d} k(\delta(t_c, t_q)). \tag{3.4}$$

Here, the term $k(\delta(t_c, t_q))$ denotes the result of applying a given kernel function to occurrences of the query q and candidate c that are separated by a distance $\delta(t_c, t_q)$. We assume that the distance here is measured by the number of words separating the query from the candidate, although some other measure may be more appropriate for some languages. The kernel function k is assumed to be a function that calculates some notion of proximity for a given distance between terms. The result of the nested sum is the total of all kernel function values (i.e., proximity values) for all possible pairings of occurrences of a query q and a candidate c in the document d.

In the example shown in Figure 3.4, two instances of the query, “Essex”, are found at two different positions. Two candidates $c_1 = \textit{Simon Lucas}$ and $c_2 = \textit{Udo}$ are found in the same document. It can be seen that c_1 is mentioned twice in the document, whereas c_2 is only mentioned one time. Using the formula above, the total proximity value for the first candidate $t_p(c_1)$ is $0.01+0.05+0.15 = 0.21$, and, for the second candidate, $t_p(c_2)$ is 0.10. The second mention of c_1 appears at a close distance from both instances of the query. Thus, it has been boosted twice. To ensure that the resulting value is a probability, we divide the result

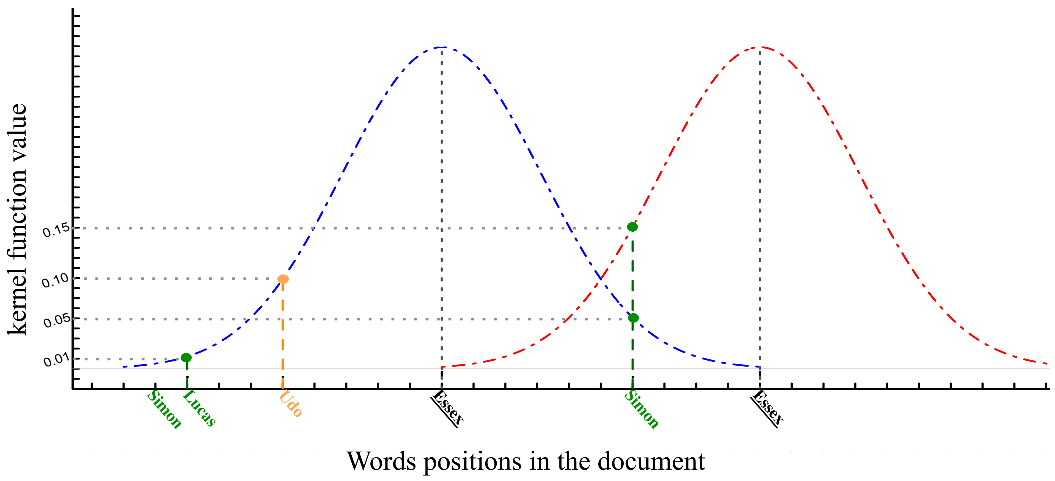


Figure 3.4: An illustration of the proximity method for two queries. This example includes three occurrences of two candidates. For simplicity, we represent each query by one word only.

for each candidate by the sum of the results for all candidates. Thus, the result for c_1 would be $0.21/0.31 = 0.678$ and for c_2 would be $0.10/0.31 = 0.322$.

The proximity-based model (PBM) is generated by adding the normalised candidate proximity, $P_{proximity}(c|d,q)$ as per Equation 3.3 to the candidate frequency, $P_{frequency}(c|d)$ as per Equation 3.2. In the PBM, we calculate the candidate score[2], given a document as follows:

$$candidate\_score = \frac{P_{frequency}(c|d) + P_{proximity}(c|d)}{\sum_{i=1}^{N}(P_{frequency}(c_i|d) + P_{proximity}(c_i|d)),}. \tag{3.5}$$

The normalising denominator in the above equation is used to ensure that the resulting value lies between zero and one[3].

[2]The term 'score' is used here rather than 'probability' because it cannot be guaranteed that the outcome of equation is a probability distribution.

[3]The candidate probability is not defined for a document length of zero.

3.5 Adaptive Window Size for Proximity Ranking

It is possible to evaluate the proximity between candidate entities and query terms in a certain document at different levels. One option is to use the whole document for a proximity search. A different approach, however, is to perform the proximity search in a smaller radius around the query terms (we call these levels *window sizes*). Using a smaller window size will further emphasise candidates that appear very close to the query terms which, in turn, is expected to increase precision. On the other hand, larger windows will include more candidates, and thereby increase recall.

As discussed in Chapter 2, researchers have considered a number of different window sizes when calculating proximity. However, the best window size for one collection does not always perform optimally for other datasets. The optimal window size is strongly influenced by the nature of the documents that constitute the collection. Moreover, the best window size for each document might not be defined at the collection level. Indeed, it is possible that every document in the collection could have its own optimal window size. In a document collection of heterogeneous information sources, there will be a wide variety of document types. It could be argued that, in general, each document has features distinguishing it from the other documents in the collection. These features could be utilised in the process of setting the window size in order to improve the overall ranking function. While many document features can be exploited, our focus in this work is based on four main features: document length, candidate frequency (i.e., the number of candidates that appear in a document), average sentence length, and readability index. One reason for the choice of these features was that they have been investigated in previous work, such as the investigation of document length in [Miao et al., 2012]. They also appeared to be good candidates as they have been shown to affect performance in prior research on expert finding or information retrieval in general (i.e., number of entities in the frequency-based model [Balog et al., 2006] and average sentence length in the BM25 ranking method [Robertson et al., 1994]). Other more sophisticated features may include document structure and semantic relationships within documents. Future work might look at broadening the collection of document features considered.

The ways in which the window size is adjusted, based on these factors, are described below.

Document Length : according to Miao et al. [2012], larger documents are likely to contain more occurrences of a query topic. It is also more likely that these documents will include irrelevant words (noise). Thus, to minimise the negative influence of noise, the window size should be reduced as the document increases in size.

Candidate Frequency : this refers to the number of candidates found in a document. The window sizes for documents containing more occurrences of candidate evidence should be increased to accommodate these occurrences.

Average Sentence Length : the window size is adjusted in proportion to the average sentence length (in number of tokens) in the document.

Readability Index [4]: the window size is adjusted using the readability index; the window should get larger as the index gets smaller.

After establishing the size of the window using Equation 3.6, the window is applied to every full match for the query found in the document.

[4]The Flesch-Kincaid test was used to calculate the Readability Index in this experiment.

$$\text{WindowSize}(d) = \sum_{i \in \text{features}} \alpha_i \cdot f_i(d), \qquad \text{where } \sum_i \alpha_i = 1 \;\text{ and }\; \alpha_i \geq 0. \quad (3.6)$$

Next, the candidate evidence neighbouring the query is extracted. Each piece of candidate evidence within the window will be given a weight, depending on its distance from the query. The advantage of this method is that it provides a graded proximity boost. Candidate evidence that is close to the query will receive the highest boost. As the candidate evidence drifts further and further away, the boost will gradually decrease until it reaches the end of the window.

Note that a document may contain multiple query terms. In this case, we place a window around each occurrence of a query term. If, for example, a document has two mentions of the query, two windows are placed, each centred at different locations. If the two windows are close to each other, both windows will boost the candidates that appear between them.

The main function of a window is to create a threshold distance from the query terms after which entities will not be considered. This approach gives more emphasis to those entities that appear closer to the query terms. Using the kernel function alone would include more distant entities, especially if the employed kernel has long tails. Different window sizes could be used along with the same kernel to give emphasis to different threshold levels (See Figure 3.5). An alternative approach, which we did not adopt in this thesis would be to *first* determine the size of the window and *then* impose a kernel within that window (i.e., shape the kernel by the window size). This would be achieved by including the window size as a parameter for the kernel.

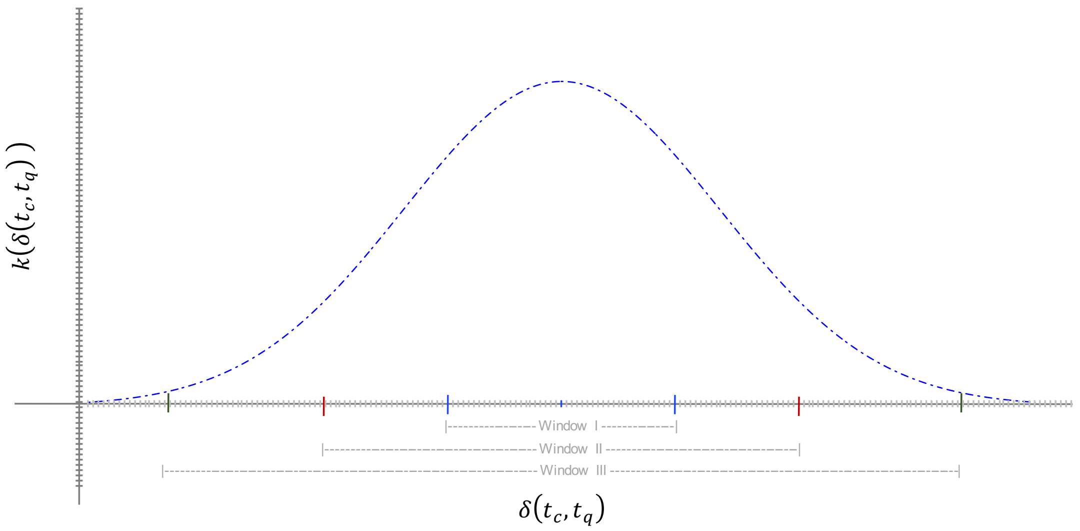


Figure 3.5: The difference between kernel width and window size.

3.6 Summary

In this chapter, we formally defined the task of finding an entity. We outlined two main models for accomplishing this task. In the Frequency-based model, candidates are ranked, based on the frequency of the occurrence of their evidence in the highest-ranked relevant documents. The second model was the Proximity-based model which employs a notion of proximity between the query and the candidate evidence in its ranking process. We outlined a novel approach for use in the proximity association process. This approach can be used to generate an adaptive window of text for each document. The size of the adaptive window is determined from unique document features including the document length, the number of entities in the document, the average sentence length, and the readability index.

This chapter concludes the theoretical background required for an understanding of the experimental part of this thesis. In the second part of this thesis we will evaluate our proposed models from a practical view point. First, we will describe the experimental setup which was used for all of our experiments. Second, we shall discuss a pilot study which was conducted to assess the validity of our proposed method. Finally, we shall thoroughly evaluate our approach under different settings and in various environments.

Part II

Evaluation

4 Experimental setup

The previous chapters developed the idea of using an adaptive window to improve the overall entity finding process. In this chapter, we outline the setup of the experiments that we shall use to validate our approach. We begin by discussing the abstract architecture of an entity finding system. Next, we give details of the indexing process including the data pre-processing and index construction procedures. After this, we give an overview of our approach to the evaluation of our model. The chapter concludes with a description of the baseline models used in our work.

4.1 General Architecture

As mentioned in the introduction, one of our research goals is to develop an entity finding model that is applicable in a diverse range of scenarios. Consequently, we want to simplify the system's framework so that it will adapt to different environments. We use the same framework across all experiments rather than implementing specific frameworks for each one.

Figure 4.1 shows the general framework of an entity finding system. The framework comprises two main stages. In the first stage, the highest ranked documents matching the user's query are retrieved, while in the second stage, candidate entities are extracted from these documents and then ranked. The presented framework could be applied to any type of entity. It is also language independent.

The input for the entity finding system is the user's submitted query, which is typically a piece of plain text describing their information need. Following normalisation and the application of various query expansion techniques, the query is passed to an underlying search engine whose task it is to retrieve the documents in which the query appears. The candidate entities are then extracted from the retrieved documents and ranked using one of a number of different ranking methods. The process concludes with the generation of a ranked list of relevant entities.

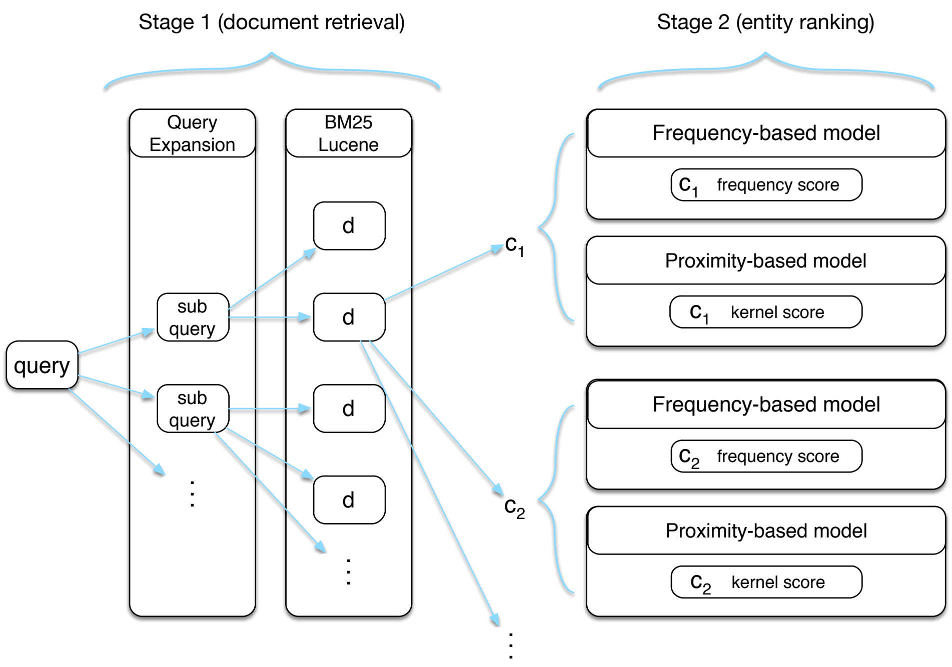


Figure 4.1: A general entity finding framework. Any kernel function, including Gaussian, Triangle, Circle, etc., may be used to generate the kernel function score. The final ranking for each candidate is computed using a combination of the frequency-based model and the proximity-based model.

4.2 Indexing

4.2.1 Data Preprocessing

The document collection may contain documents in various formats (e.g., HTML, XML, PDF, Word, etc.). Regardless of the original format, each document is converted into plain text prior to processing. For instance, we remove all HTML markup, and scripts, etc from HTML pages. Moreover, as our aim is to create a general system, we treat all document types in the same way and do not exploit any internal document structure that may be present.

We shall explain our pre-processing steps using the sample document shown in Figure 2.8, (p. 47). After removing the HTML markup, the document has the form:

```
Simon M. Lucas — Game Intelligence Group
Game Intelligence Group University of Essex
Simon M. Lucas is a Professor of Computer Science at the University of Essex,
Colchester, Essex, U.K. His main research interests are in machine learning and
games. He has published widely in these fields with over 150 peer-reviewed papers
and is the Founding Editor-in-Chief of the IEEE Transactions on Computational
Intelligence and AI in Games.
```

For efficient document retrieval, IR systems typically undertake a procedure known as indexing which enables the system to quickly identify the documents from a given corpus that match the user's query. According to Croft et al. [2015], indexing is the process that builds the structures that enable searching. The ranking process uses these structures, together with the user's query, to produce a ranked list of documents.

The indexing procedure begins with a step known as tokenisation which is the process of breaking the input text into small indexing elements (or tokens). All characters in each token are converted into lower case (i.e., case folded). At this stage, all punctuation is removed. The next step is called stopping or stop word filtering. According to Grossman and Frieder [2004], stop words are those terms that are deemed relatively meaningless in terms of document relevance (e.g., "the"), and thus they may be filtered out from the list of indexing terms. Removing these words may simplify the index construction process by reducing its size, time consumption, and storage costs.

In some cases, a user may include a word in their query and the relevant document may only contain a variant of this word. This may include plurals (e.g., "games"), gerund verb forms (e.g., "learning"), and past tense suffixes (e.g., "published"), to name but a few. These syntactical variations could prevent a perfect match between a user query and a prospective document [Baeza-Yates and Ribeiro-Neto, 2011]. The process of reducing the different forms of a word that occur because of inflection or derivation to a common stem is known as stemming or conflation [Croft et al., 2015]. In this work, we employ Porter's stemming algorithm [Porter, 1980] which is, according to Jurafsky and Martin [2009], considered to be one of the most widely used stemming algorithms due to its simplicity and efficiency.

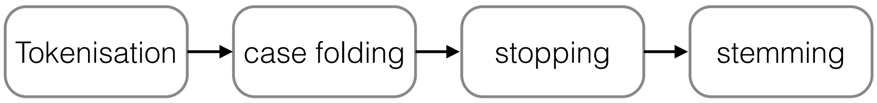


Figure 4.2: Main pre-indexing steps. These steps will also be applied to user queries at retrieval time.

After the steps shown in Figure 4.2 are completed, our sample document will look like:

```
simon lucas game intellig group game intellig group univers essex simon lucas
professor comput scienc univers essex colchest essex uk research interest machin
learn game publish wide field 150 peer review paper found editor chief ieee
transact comput intellig ai game
```

4.2.2 Index Construction

We employ the Lucene[1] search engine to index and query our datasets. The main role of an index is to provide a map of terms and documents which allows the efficient retrieval of documents from a corpus.

We apply a typical inverted index implementation to our document collection [van Rijsbergen, 1979]. The input for the indexing process is a list of normalised tokens for each document, (i.e., the output of the pre-processing pipeline shown

[1]see `http://lucene.apache.org/`

in Figure 4.2). The process begins by sorting this list alphabetically. Next, any duplicate terms appearing in the individual documents are merged together. The instances of each term are grouped. Since a term is often mentioned in numerous documents, this way of organising data will significantly help to reduce the size of the index. The index entries for each term consist of a list of documents containing that term. This grouping provides a mechanism for scoring search results: if a number of search terms map to a certain document, then this document is likely to be considered relevant.

Once a collection of documents has been indexed, there is a need to rank the documents in response to a query. This ranking is performed at retrieval time, immediately after each query is received. In section 2.1, we described several state-of-the-art approaches for matching and ranking documents in response to a query. We have chosen BM25 as the default document retrieval approach for our experiments. Only the top 100 documents returned by the search engine are considered for each query.

4.3 Evaluation Approach

One way of evaluating an IR system is to use the Cranfield evaluation paradigm, as discussed in Section 2.3 (p. 38). This type of evaluation is conducted offline and does not involve the interaction of real users with the system. Another type of evaluation is known as user-centric evaluation, or online evaluation. In online evaluation, users interact with the system and evaluation data is collected via user interviews or surveys. In this thesis, we limit our evaluation to the offline approach, and leave the online approach for future research.

Table 4.1 provides a general overview of the test collections used in this thesis, all of which have been indexed using the reverse indexing process described earlier.

Table 4.1: General overview of the test collections used in this thesis

Test Collection	Date created	Domain	Public
W3C	2005	Enterprise - Organisation	✓
CERC	2007	Enterprise - Organisation	✓
UvT	2009	Enterprise - University	✓
TU	2011	Enterprise - University	✓
Essex	2012	Enterprise - University	
ClueWeb	2007	Web	✓

As discussed in Section 2.3, the evaluation of IR systems requires not only document collections, but also a specified set of queries and their relevant judgements which enables comparisons with the system's output results. Figure 4.3 depicts part of the relevance judgement for topic number '75' of the CSIRO dataset. In this example, the first column represents the query number, the third column is the candidate identifier, (the candidate's email in this case), and the final column shows the judgement for this candidate (i.e., 0 is irrelevant and 1 is relevant). Thus, for this topic, we have five candidates identified as experts by the gold standard. When we test and compare the performance of IR systems, each system's results are evaluated on the basis of the same gold standards and the same metrics.

```
...
75  0  chris.margules@csiro.au  0
75  0  clare.peddie@csiro.au  0
75  0  darla.hattonmacdonald@csiro.au  1
75  0 daryl.stevens@csiro.au  1
75  0  david.ellis@csiro.au  1
75  0  di.peter@csiro.au  0
75  0  geoff.syme@csiro.au  1
75  0  grace.mitchell@csiro.au  1
...
```

Figure 4.3: An example of CSIRO relevance judgements

We utilise different evaluation measures in our experiments in order to compare our results with those reported in the literature. However, for all of our experiments, the primary metrics employed are the mean average precision (MAP) and the mean reciprocal rank (MRR) (Section 2.3.2).

We selected MAP as one of our primary metrics since it provides a more robust and stable measure when aggregating results from multiple queries than other metrics [Manning et al., 2008]. MRR, on the other hand, was chosen as an appropriate measure since entity finding can be regarded as one of those applications in which achieving high accuracy in the top ranks is more important than finding all of the relevant entities for a topic (i.e., recall).

4.4 Baseline Models

The performance of our entity finding approach is based on the premise that this approach can return the candidate entities with the highest degrees of relevance to the query at the top of the ranked list. In this work, we compare our entity finding approach with several baseline systems in order to evaluate its effectiveness. We can categorise these baselines into three groups.

The most elementary baseline model considered is the frequency-based model (Section 3.3). In this baseline, candidates are ranked by how frequently they appear in the relevant documents (Equation 3.2, p. 74).

Some better performing baselines apply a proximity-based model, (Section 3.4), using fixed window sizes. In these approaches, candidate entities are ranked by how frequently they appear in the relevant documents as well as by the proximity of these mentions to the query terms. In our evaluations, we use the following two methods for calculating the window sizes:

Table 4.2: A description of the models used in this study.

Model	Proximity	Window Type	Features
FBM	No	N/A	N/A
PBM_Fix200	Yes	Fixed - 200 words	N/A
PBM_FixBest	Yes	Fixed - based on the collection	N/A
PBM_Full	Yes	Adaptive	N/A
PBM_α_l	Yes	Adaptive	Document length
PBM_α_c	Yes	Adaptive	Candidate frequency
PBM_α_v	Yes	Adaptive	Average sentence length
PBM_α_r	Yes	Adaptive	Readability index
PBM_Adaptive	Yes	Adaptive	All four features

Fixed at a window size of 200 words: a window size of 200 words was suggested in several previous studies (e.g., [Bao et al., 2008]), as discussed in Section 2.4.

Best performing window: since datasets vary, each one may have its own optimum window size. We perform some initial runs (or "training runs") to determine which fixed window size has the best performance for a given collection. We select part of the query set to perform this task. These queries are used to tune the parameters, thus providing a clear distinction between training and testing data. We use a subset of queries to evaluate a range of window sizes and rank them using the MAP metric.

The third group of baselines uses the idea of adaptive window sizes (Section 3.5). Here the window size for each document is determined from the features of the document. We first set the adaptive window size equal to the actual length of the document. We then test the adaptive window sizes found by considering each document feature individually. Finally, we consider the combined influences of all document features taken into account by the *PBM_Adaptive* model to achieve a better window size. A summary of the models employed in this study is provided in Table 4.2.

4.5 Summary

In this chapter, we described the experimental setup used for all of our experiments. We began our discussion with an abstract architecture for an entity finding application that was divided into two main stages: a document retrieval stage and an entity ranking stage. Although the contributions of this thesis mainly pertain to the second stage of this architecture, we have also set up datasets for the document retrieval stage. Consequently, we discussed our methods for preprocessing documents in order to prepare them for the indexing process.

We outlined our approach to indexing our data sets and gave an overview of the six test collections used in this thesis to represent different entity retrieval scenarios. We also discussed the ways in which topics and relevance judgements are used in the evaluation process. After identifying the main metrics that we shall use to measure retrieval effectiveness, we gave details of the baseline models used in our experiments to evaluate the effectiveness of our adaptive window size approach. In the next chapter, we discuss the implementation of the material introduced in this chapter and present an experimental evaluation of our entity finding approach.

5 Experiments

In this chapter, we present the experimental evaluation of our entity finding approach using the experimental setup discussed in the previous chapter. We evaluate the adaptive window-size approach in different environments and under various settings. We begin with a discussion of the pilot studies which were performed to evaluate the validity of the approach. This is followed by a discussion of the outcomes of our expert finding and entity finding experiments. We conclude with a discussion of our experiments exploring query expansion and alternative document retrieval algorithms.

5.1 Pilot Study of Entity Finding

In this pilot study, we explored the practicality of the adaptive window size approach in an entity finding application. Although we focused on the entity of type "Person" in this study, the approach could equally well be applied to any other type of entities. The approach automatically suggests the size of the window for each document within which the proximity function is applied. We compared this approach with a number of baselines to determine its capability, evaluated a number of proximity functions (kernels) and also tested a range of parameters for these functions.

We selected the W3C dataset and test collection (Section 2.3.1.1) for our pilot study. This dataset was one of the first publicly available collections produced to evaluate expert finding systems. It was used at the TREC Enterprise Track in 2005 and 2006 [Craswell et al., 2005; Soboroff et al., 2006].

As an initial experiment, we considered three document features, namely document length, number of candidates, and average sentence size. The adaptive window size was calculated as follows:

$$\begin{aligned}\text{WindowSize} = \frac{\sigma}{3} \cdot (\log(\frac{1}{\text{DocLength}}) * \alpha_l \; &+ \text{CanFreq} * \alpha_c \\ &+ \text{AvgSentSize} * \alpha_v).\end{aligned} \tag{5.1}$$

Here σ is a variable that allows us to scale the window size. In addition, α_l, α_c, and α_v are the weighting factors for the document-length, candidate-frequency and average-sentence-size features, respectively.

We explored a wide range of values for σ (see Table 5.1). The α weighting factors, which determine each feature's contribution in the equation, were set empirically, using values satisfying the condition $\alpha_l + \alpha_c + \alpha_v = 1$. The TREC

2005 data includes ten training topics[1]. We used these topics as training data to find the optimal α values, thereby making a clear distinction between our test and training data. Although the proposed model uses all three document features, we will also report on experiments in which we used each feature individually. For our proximity model, three different kernel functions were used to calculate the weight, namely the Gaussian, Triangle, and Cosine kernels.

We used our adaptive window-size method (Equation 5.1) in conjunction with the three proximity functions and different σ values ranging from 0 to 1,000 by increments of 50. We only report the results for the σ values between 350 and 650. The results below 350 and above 650 drop gradually and are not reported here as they are less interesting. Furthermore, we calculated a baseline for each proximity function using the *PBM_Full* model (see Table 4.2). In this baseline, we set the window size to be equal to the document length.

For comparison, we also used a range of fixed window sizes *PBM_Fix*. We calculated MAP for fixed windows with sizes in the range from 100 to 1,000 words in increments of 50. We repeated the experiments using the three proximity functions. The Gaussian kernel was seen to be significantly better than the other two functions, with a top result of MAP=0.27 at a window size of 200 (see Figure 5.1).

To test the effect of each document feature separately, we generated an adaptive window size on the basis of only the feature under consideration and applied it in conjunction with the Gaussian proximity function. In Table 5.2, we report the best runs for each feature separately (i.e., CanFreq with $\sigma = 250$, AvgSentSize with $\sigma = 600$, and DocLength with $\sigma = 450$).

Our results are summarised in Table 5.3. The top MAP results of 0.3454 and 0.591 (for the 2005 and 2006 test sets, respectively) are achieved using a

[1]`http://trec.nist.gov/data/enterprise/05/ent05.expert.trainingtopics`

Table 5.1: The performance of the adaptive window size approach for different proximity functions. The highest scores for each category are typeset in boldface and the best run overall is typeset in boldface and underlined.

		σ	MAP	r-prec	bpref	P@5	P@10	P@20
	FBM	N/A	0.1532	0.2531	0.2749	0.3210	0.2519	0.1908
Gaussian	*PBM_Fix*	N/A	0.3001	0.3554	0.4297	0.5092	0.3595	0.3089
	PBM_Adaptive	350	0.3363	0.3808	0.4787	0.5200	0.3900	0.3350
		400	0.3342	**0.3975**	0.4737	0.5200	0.4000	0.3300
		450	0.3454	0.3955	**0.4954**	0.5200	0.4099	**0.3450**
		500	**0.3454**	0.3905	0.4861	0.5200	0.4199	0.3350
		550	0.3443	0.3905	0.4890	0.5200	**0.4299**	0.3400
		600	0.3402	0.3905	0.4851	0.5200	0.4199	0.3350
		650	0.3357	0.3821	0.4792	0.5200	0.4099	0.3350
Triangle	*PBM_Fix*	N/A	0.2358	0.3331	0.3602	0.4023	0.3329	0.2750
	PBM_Adaptive	350	0.3126	0.3642	0.4494	0.4800	0.4099	0.3199
		400	0.2974	0.3509	0.4427	0.4800	0.4099	0.3199
		450	**0.3261**	0.3793	**0.4623**	0.5199	**0.4299**	**0.3300**
		500	0.3169	**0.3804**	0.4330	0.5600	0.4200	0.3050
		550	0.3144	0.3776	0.4209	0.5600	0.4099	0.3050
		600	0.3036	0.3767	0.4093	**0.5800**	0.3800	0.2950
		650	0.2836	0.3490	0.3869	0.5400	0.3900	0.2800
Cosine	*PBM_Fix*	N/A	0.2700	0.3605	0.4078	0.4102	0.3495	0.3095
	PBM_Adaptive	350	0.2735	0.3557	**0.4494**	0.4219	0.3999	0.3499
		400	0.2757	0.3414	0.3149	0.4191	0.4199	0.3599
		450	0.2761	0.3498	0.3149	0.4191	0.4199	0.3599
		500	**0.2811**	0.3639	0.3199	**0.4241**	**0.4399**	**0.3599**
		550	0.2800	0.3639	0.3199	0.4232	0.4399	0.3599
		600	0.2756	0.3639	0.3149	0.4155	0.4399	0.3599
		650	0.2744	0.3639	0.3149	0.4155	0.4199	0.3599

Gaussian proximity function with an adaptive window size and $\sigma = 500$. We found that the difference between our best run and the baseline was statistically significant for both test collections.

A comparison of our results with the best TREC runs shows that our results are on par with the best-performing submissions for 2005. However, our results are below the best-performing runs for 2006[2]. The low scores are due to the fact that, in this experiment, we used the same pipeline for both test collections. This

[2]Our results here are comparable with those of the 5 best performing systems from TREC 2006 [Soboroff et al., 2006].

Table 5.2: The performance of the adaptive window size approach using a single feature. Only the best result according to the MAP measure is reported.

Feature	Doc. Length	Candidate Frequency	Avg. Sentence Size
Best σ value	*450*	*250*	*600*
MAP	0.2777	0.2806	0.2798
bpref	0.3452	0.3452	0.3269
r-prec	0.4112	0.4147	0.4199
P@5	0.4199	0.4199	0.4189
P@10	0.3499	0.3599	0.3499
P@20	0.3050	0.3100	0.3100

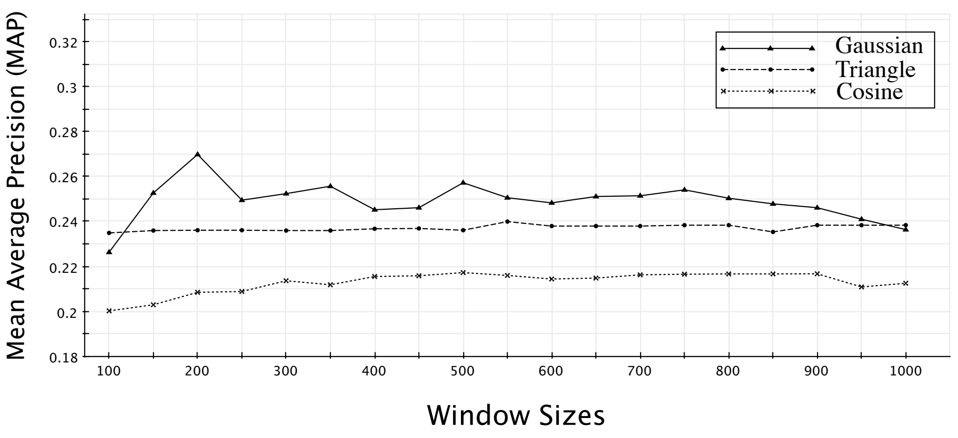


Figure 5.1: Comparison of the MAP results for fixed window sizes using three kernel functions, namely Gaussian, Triangle, and Cosine kernels.

pipeline does not include any query expansion and formulation steps. On the other hand, the best systems at TREC 2006 employed several query expansions and formulation methods [Bao et al., 2006; Zhu et al., 2006] which enhanced the performance of these systems.

The potential of the adaptive window size method is evident from the results of this pilot study. Our method has proven to be more effective than the frequency based baseline and proximity-based methods employing a range of fixed window sizes. The results show an improvement, even when applying the adap-

Table 5.3: Summarised results of the W3C dataset experiment. The highest scores are typeset in boldface. We use △ and ▲ to denote statistically significant improvement compared to the baseline using the t-test at p-values of 0.05 and 0.01, respectively.

	TREC 2005		TREC 2006	
	MAP	MRR	MAP	MRR
PBM_Fix200	0.290	0.748	0.536	0.704
PBM_Adaptive	**0.349▲**	**0.797**	0.591 △	0.765
Best TREC	0.275	0.727	**0.643**	**0.961**

tive window size found using only a single document feature. This suggests that each feature is individually important. This motivates us to seek other document features that might contribute to the performance of this model. Our subsequent experiments will incorporate an additional document feature, namely the readability index. Furthermore, the findings show that using multiple features of the document when generating the window size can lead to significant improvements over the baselines. Finally, of the three kernel functions considered in this study, the Gaussian function provided the best results, so we will use this function in future experiments.

In the following sections, we will discuss the experiments that we have performed using the four test collections: CSIRO, UvT, Essex, and ClueWeb09. The first test collection was used for the TREC Enterprise Track in 2007 and 2008 [Bailey et al., 2007b; Balog et al., 2008]. A description of this collection is provided in Section 2.3.1.2. The second collection, UvT, has been used by many researchers to evaluate expert profiling and expert finding systems [Berendsen et al., 2013; Gollapalli et al., 2011; Hofmann et al., 2010]. This collection is described in Section 2.3.1.3. The Essex data set was developed for this thesis. A description of this dataset appears in Section 2.3.1.4. Finally, we experimented with ClueWeb09 which was used at the TREC 2009 Entity Track [Balog et al., 2009b]. Some details about this test collection are presented in Section 2.3.1.5.

As mentioned in Section 5.1, we have extended the set of document features considered in the calculation of the adaptive window size. The additional feature considered is the readability index (see Section 3.5). To accommodate this feature, we have expanded Equation 5.1 to include this index and its weighting factor α_r:

$$\begin{aligned} WindowSize = \frac{\sigma}{4} * (\log(\frac{1}{\text{DocLength}}) * \alpha_l \; + \text{CanFreq} * \alpha_c \\ + \text{AvgSentSize} * \alpha_v + \text{ReadabilityIndex} * \alpha_r) \end{aligned} \tag{5.2}$$

5.2 Expert Finding Experiments

5.2.1 CERC Dataset

Although the CERC test collection provides no master list of candidates, the process of generating such a list is straightforward as all email addresses conform to a standard format (i.e., `firstname.lastname@csiro.au`) which can largely facilitate the recognition process. By locating the email addresses, we can also discover the full names of the candidates since the email addresses contain this information. After locating all email occurrences in the collection, we removed the duplicates and any emails that did not refer to an individual, such as organisational email addresses. The resulting list contained 3,480 candidates.

We randomly selected ten (training) topics for use in finding the optimal settings for the variables in Equation 5.2. We selected N different α combinations. Every combination had four values corresponding to the four document features, each of which was set discreetly between 0 and 1 in increments of 0.1. The sum of the four α values in each combination was always equal to one. Table 5.4 shows some of the combinations of α values that were used in this study.

Table 5.4: Example of some α combinations used in this study.

α_l	α_c	α_v	α_r
0.2,	0.0,	0.3,	0.5
0.1,	0.5,	0.1,	0.3
0.0,	0.5,	0.1,	0.4
0.0,	0.6,	0.0,	0.4
0.1,	0.7,	0.0,	0.2

To test the effect of each document feature separately, we used the training topics to first generate the adaptive window size obtained by considering that feature in isolation.

Figure 5.2 shows the MAP at σ values between 0 and 1,000. The variance analysis method, ANOVA, tests the statistical significance (main effects) at $p < 0.05$. It suggested a statistically significant difference between the results found using the different features. To investigate these differences further, we used pairwise t-tests at $p < 0.01$ which revealed that, for the TREC 2007 results, the consideration of the number-of-candidates feature on its own improves the results significantly over those obtained when the other three features are

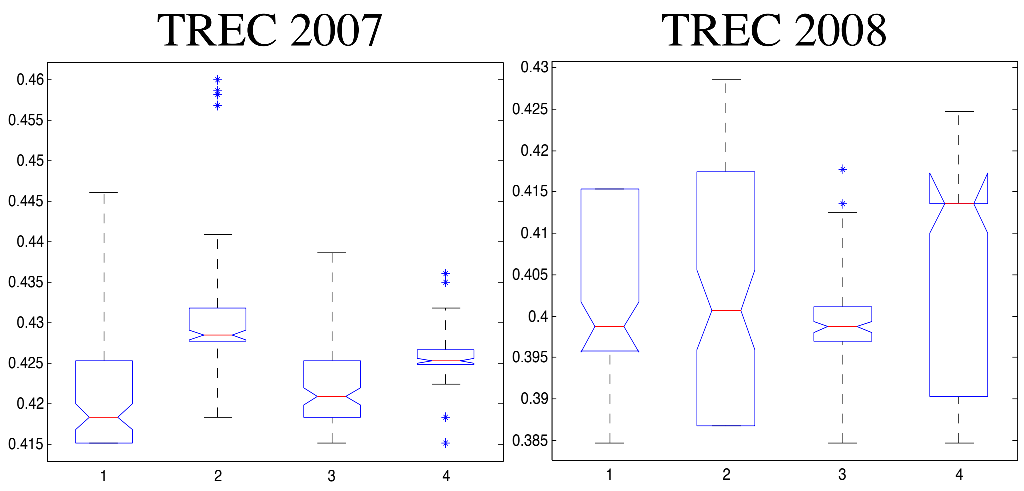


Figure 5.2: The MAP results with an adaptive window using a single feature, where 1 is the document length, 2 is the number of candidates in the document, 3 is the average sentence length, and 4 is the readability index.

considered in isolation. The results for the readability-index feature also showed a statistically significant difference from the results using the document-length and average-sentence-size features. The TREC 2008 results showed similar results with statistically significant differences between the results found using the number-of-candidates feature and those found using the other features. However, in TREC 2008, the results for the readability-index feature showed a statistically significant difference from the results found using the document-length feature only. It can be seen from Figure 5.2 that the results found using the number-of-candidates feature were the highest for both test sets.

We selected the proximity model, Equation 3.3 (p. 74), with a fixed window as a baseline for this experiment. We used the highest-scoring results from the TREC Enterprise Track for 2007 and 2008 as a basis for comparison.

From the results tables (Table 5.5) it can be seen that the use of the adaptive window size method resulted in an improvement ranging from 10% to 20% over the fixed window baseline. Moreover, the adaptive window method which

Table 5.5: Results of the adaptive window method compared to the fixed window baseline and highest-scoring systems from TREC Enterprise Track. The best scores are in boldface. We indicate statistically significant improvement compared to the baseline using the t-test where $p < 0.01$ using ▲.

		TREC 2007			TREC 2008	
		MAP	MRR		MAP	MRR
PBM_Fix200	(baseline)	0.474	0.643		0.411	0.791
$PBM\_Adaptive$		**0.521▲**	**0.763**		**0.457▲**	0.816
Best TREC#1	[Fu et al., 2007b]	0.463	0.633	[Balog and de Rijke, 2008a]	0.449	**0.872**
Best TREC#2	[Duan et al., 2008]	0.442	0.613	[Shen et al., 2008]	0.421	0.724
Best TREC#3	[Zhu et al., 2008]	0.434	0.580	[He et al., 2008]	0.413	0.761

combined all four document features was awarded a higher MAP than all of the methods which considered these features individually. Using a paired t-test on the average precision values, we found the difference between our method and the corresponding baseline to be statistically significant. This significant improvement was reported for MAP only. The table shows that, according to the MAP measure, our results were on par with the best-performing TREC submissions.

5.2.2 UvT Dataset

Of the UvT collection topics, 981 English and 978 Dutch topics include a relevance judgement and were therefore used in this experiment. We followed the approach of Fang et al. [2011] by randomly selecting a subset of 200 (training) topics to optimise our parameters. The remaining 781 English and 778 Dutch topics were used for testing, thereby creating a clear distinction between the topics used for training and testing. We applied the weights established using the UvT collection to the TU collection. This allowed us to use the whole testbed of the TU collection for testing. Table 5.6 includes some examples of the English and Dutch queries used in the evaluation process.

Table 5.6: Example of English and Dutch queries used in the evaluation process.

English	auditing	accounting	information systems organisation
	feminism	european tax law	corporate governance
	foreigners	anorexia nervosa	general economics
Dutch	handhaving	sportrecht	beleggingsmaatschappijen
	expertsystemen	conflictbemiddeling	godsthematiek hedendaagse
	tijdreeksen	vakbeweging	vroege middeleeuwen

We used the training topics to search for the best fixed window size for use in the *PBM_Fix* baseline. We selected window sizes ranging between 0 and 10,000 words by increments of 10. Figure 5.3 shows the MAP and MRR for the selected window sizes. We noted that the results became constant after a window size of approximately 3,000 words as, after this point, the window sizes began to reach the document length. The best fixed window sizes were found to be 930 and 480 words for English and Dutch documents, respectively.

Next, the σ parameter was tuned by selecting values between 10 and 1,500 in increments of 10. During this process, the α values were set to the values used in the previous experiment (i.e., $\alpha_l = 0.2$, $\alpha_c = 0.5$, $\alpha_v = 0.3$, & $\alpha_r = 0.1$). Figure 5.4 shows a plot of the MAP and MRR against the σ values. The highest scores

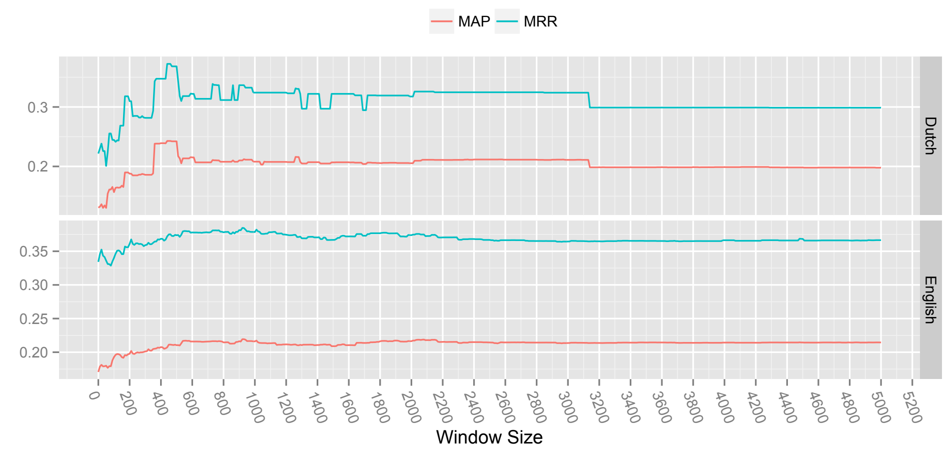


Figure 5.3: UvT MAP and MRR for fixed window sizes between 0 and 5,000.

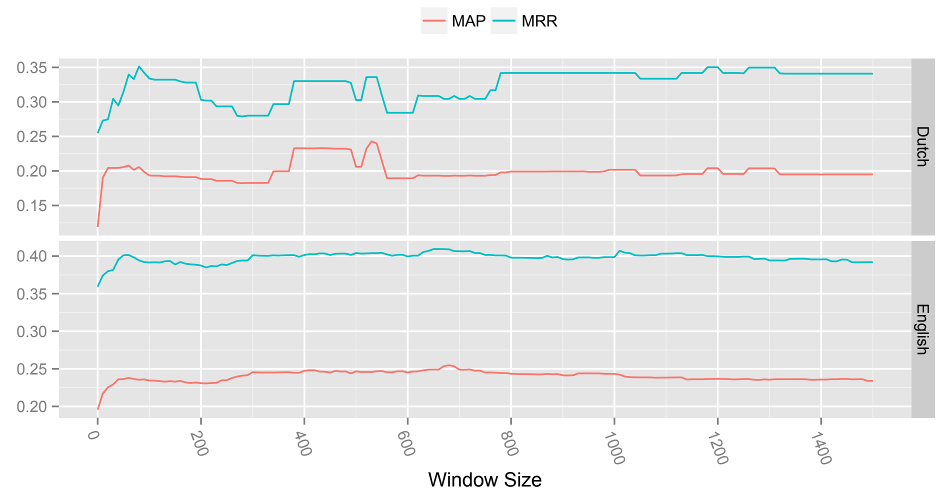


Figure 5.4: UvT MAP and MRR scores plotted against σ values between 0 and 1,500.

for English documents were achieved when $\sigma = 680$, while the highest scores for Dutch documents were achieved when $\sigma = 530$.

In the UvT experiment, we compared the performance of the adaptive window model with that of a range of alternative baselines and other methods. Tables 5.7 and 5.8 summarise the results for the UvT and TU collections, respectively.

Table 5.7: UvT collection - comparison of the experimental results for the proposed adaptive window model with the results of various baselines and other methods. The results with a significant improvement at the $p < 0.05$ level are indicated in italics. The best result for each measure is indicated in boldface.

		English		Dutch	
	run	MAP	MRR	MAP	MRR
Other methods	Model 1	0.2008 ⋆	0.3551	0.1730	0.3382
	Model 2	0.1994 ⋆	0.3564	0.1737	0.3374
Baselines	FBM	*0.1862*	0.3480	*0.1784*	0.3294
	PBM_Fix_200	*0.2211*	0.3909	0.2150	0.3999
	PBM_Fix_Best	*0.2313*	0.3922	0.2255	0.3800
	PBM_Full	0.2327	0.3912	0.2254	**0.4160**
Single Feature	α_l	*0.1909*	0.3508	*0.1801*	0.3325
	α_c	0.2351	0.3902	0.2305	0.3833
	α_v	0.2304	0.3900	0.2269	0.3821
	α_r	*0.1945*	0.3567	*0.1803*	0.3325
	PBM_Adaptive	**0.2517**	**0.4527**	**0.2443**	0.3983

⋆We did not test the significance for the other methods as the runs were not available. However, an improvement of 25% on English and 40% on Dutch could be observed in the MAP results.

Table 5.7 consists of four parts. In the first part, we report the results obtained using other methods, namely Model 1 and Model 2, which are considered to be standard methods for the expert finding task defined by Balog et al. [2006]. In the second part, we provide data from the baseline runs, *FBM* and *PBM_Fix*. To obtain this data we tested fixed windows of three sizes: the best performing window size found during training, a fixed window of 200 words and a window equivalent to the size of the document. The third part reports the results found using the adaptive window (Equation 5.2) for a single document feature. To examine each feature individually, we fixed $\alpha_l = 1$ and set $\alpha_c = \alpha_v = \alpha_r = 0$, for the first feature. Corresponding α-values were used for the remaining features. Recall that α_l corresponds to the document-length feature, α_c to the number-of-candidates feature, α_v to the average-sentence-size feature, and α_r to the readability-index feature. The fourth part of the table contains the results for the adaptive window size method (*PBM_Adaptive*). We tested the statistical significance of the differences in our MAP results from those for the *PBM_Fix* baseline by applying a paired t-test.

Table 5.8: TU collection - comparison of the experimental results for the proposed adaptive window model with the results of various baselines. The results are categorised into four relevance judgement sets, GT1 to GT4, as described in Table 2.7.

			English		Dutch	
		run	MAP	MRR	MAP	MRR
GT 1	Baselines	PBM_Fix_200	*0.3047*	0.3890	*0.3264*	0.3979
		PBM_Fix_Best	*0.3052*	0.3906	*0.3156*	0.3918
		PBM_Fix_Full	*0.3378*	0.4171	*0.3544*	0.4201
	Single Feature	α_l	*0.3530*	0.4339	0.3814	**0.4466**
		α_c	*0.2677*	0.3347	*0.3067*	0.3765
		α_v	*0.3131*	0.3981	*0.3353*	0.4058
		α_r	*0.2677*	0.3394	*0.3085*	0.3848
		PBM_Adaptive	**0.3751**	**0.4596**	**0.3816**	0.4445
GT 2	Baselines	FBM_Fix_200	*0.3783*	0.4313	*0.4361*	0.4656
		PBM_Fix_Best	*0.3775*	0.4301	*0.4280*	0.4548
		FBM_Fix_Full	0.4110	0.4575	*0.4523*	0.4689
	Single Feature	α_l	*0.4093*	0.4543	**0.4780**	**0.4962**
		α_c	*0.3537*	0.4051	*0.4168*	0.4420
		α_v	*0.3945*	0.4447	*0.4412*	0.4659
		α_r	*0.3609*	0.4130	*0.4269*	0.4537
		PBM_Adaptive	**0.4331**	**0.4770**	0.4723	0.4886
GT 3	Baselines	FBM_Fix_200	*0.4121*	0.4632	0.4429	0.4703
		PBM_Fix_Best	*0.4062*	0.4545	*0.4256*	0.4460
		FBM_Fix_Full	*0.4327*	0.4825	0.4621	0.4765
	Single Feature	α_l	0.4686	0.5171	**0.4763**	**0.4958**
		α_c	*0.3721*	0.4230	*0.4212*	0.4430
		α_v	*0.4343*	0.4854	*0.4479*	0.4710
		α_r	*0.3893*	0.4414	*0.4356*	0.4580
		PBM_Adaptive	**0.4755**	**0.5263**	0.4641	0.4812
GT 4	Baselines	FBM_Fix_200	*0.3519*	0.4395	*0.3174*	0.3811
		PBM_Fix_Best	*0.3496*	0.4350	*0.3020*	0.3680
		FBM_Fix_Full	0.3709	0.4610	0.3435	0.4062
	Single Feature	α_l	**0.3821**	**0.4726**	**0.3472**	**0.4130**
		α_c	*0.3263*	0.4051	*0.2960*	0.3540
		α_v	*0.3539*	0.4433	*0.3226*	0.3903
		α_r	*0.3330*	0.4118	*0.2944*	0.3603
		PBM_Adaptive	0.3785	0.4703	0.3462	0.4111

5.2.3 Essex Dataset

In institutions which provide higher education, finding an appropriate supervisor is a significant task for research students. This undertaking is extremely important as research students are much more likely to succeed if their supervisor's expertise closely matches their research interests [McAlpine and Norton, 2006]. One of the main problems facing students in this regard is how time-consuming it can be, especially if academics describe their work using terminology with which the student is not familiar. Many students begin the supervisor selection process by seeking peer opinions or by conducting a simple online search for relevant information [George et al., 2006].

A student may build a picture of who is likely to have the relevant expertise by looking for university academic staff who have written numerous documents about the general topic, who have authored documents exactly related to the subject, or who list the area as one of their research interests. Automating this process will not only help research students to find the most suitable supervisors, but it will also enable the university to allocate applications to supervisors, and help researchers to find other professionals interested in the same topic. The recommendation of supervisors to students is often considered to be an important research aid in any educational institution [Lelei, 2013]. The correct matching of student and supervisor interests has been shown to be a vital factor affecting student achievement [Armstrong, 2004; Fang, 2012; McAlpine and Norton, 2006].

As discussed in Section 2.2.5, the candidate list is an important component of an expert finding system. In our experiment with the Essex dataset, we compared and contrasted two scenarios for obtaining the candidate list. In the first scenario, the candidate list was provided beforehand. This is typical in organisational settings where experts are drawn from the organisation's employees (e.g. the university's academic staff). In the second scenario, the system had to generate the candidate list for each query using standard entity recognition tools. Such lists are usually extracted automatically from the pages that are returned for the query. This method promises to be more useful for finding experts from a wider (and possibly more up-to-date) field of candidates. In both scenarios, the same ranking function(s) were applied. As we constructed the dataset and test collection ourselves, we were able to compare the results produced in each scenario.

As described in section 2.3.1.4, the relevance judgements for this test collection consist of academics from the School of Computer Science and Electronic Engineering at the University of Essex. To obtain a realistic list of candidates for the first method, we decided to construct a list of academics spanning the entire university. The University of Essex has 24 different departments and schools[3] and some academics are linked to more than one department. We therefore selected academics from these departments without duplication, obtaining a list of 881 unique names.

As discussed in Chapter 4, the user query was first sent to a document search engine to extract the relevant documents for analysis (Figure 5.5). We retrieved only those documents in which there was at least one mention of a named entity. We limited our analysis to the top 100 of these documents.

For the second scenario, we used the Stanford Named Entity Tagger to recognise named entities [Finkel et al., 2005] from the top ranked documents for each query. This tagger uses a linear chain Conditional Random Field (CRF) sequence model to recognise entities. According to Fang [2012], the Stanford Tagger is considered to be one of the best publicly available named entity recognisers. Al-

[3]`http://www.essex.ac.uk/depts/`

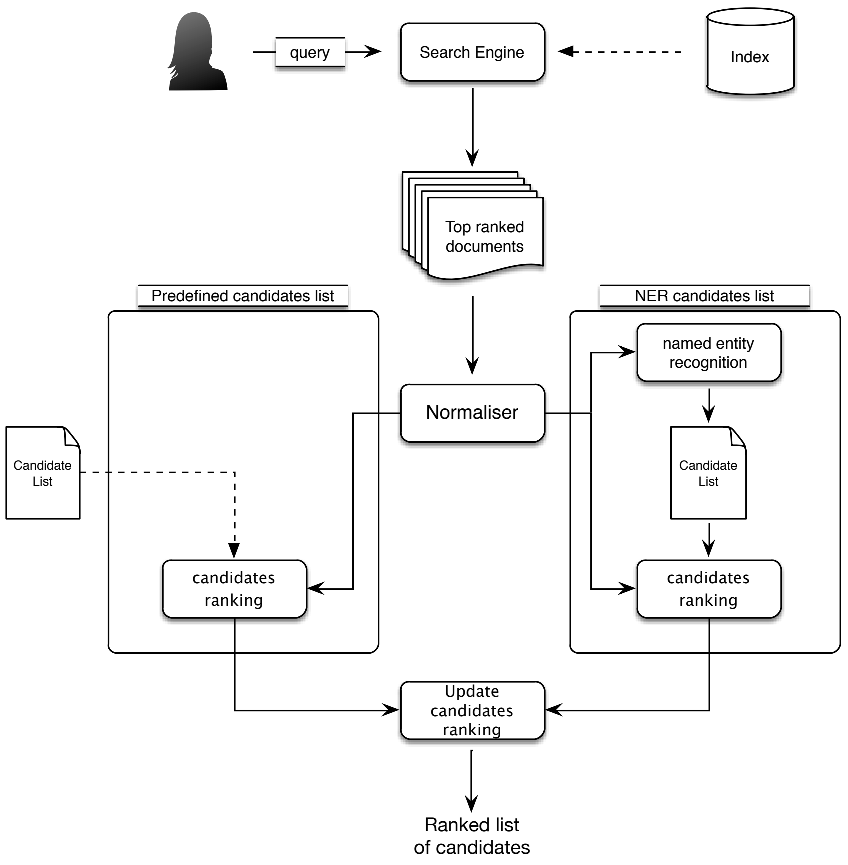


Figure 5.5: General system architecture shows the two methods of using the candidate list (i.e., predefined candidate list and NER candidate list). The dotted lines illustrate the external tools that plugged into the system.

though this tagger identifies different entity types, we concentrated on only the PERSON entity type in this experiment.

We selected the proximity model with a fixed window size of 200 words (see *PBM_Fix200* in section 4.4) as a baseline for this experiment. We also compared our results for the adaptive window size method, *PBM_Adaptive*, with those obtained using the adaptive window sizes found by considering each document feature separately.

The results are summarised in Table 5.9. There were two main findings. First, the results clearly showed that the approach using a pre-defined list of candidates

Table 5.9: Performance of the entity finding models with two methods using the candidate list (i.e., predefined and NER lists).

	run	MAP	MRR	P@10	P@5
NER List	Fix200	0.225,18	0.233,77	0.083,33	0.070,37
	PBM_α_l	0.237,71	0.245,79	0.087,03	0.077,77
	PBM_α_c	0.232,96	0.242,32	0.085,18	0.074,07
	PBM_α_v	0.280,09	0.300,40	0.092,59	0.092,59
	PBM_α_r	0.284,57	0.289,47	0.105,55	0.129,62
	PBM_Adaptive	0.351,13	0.388,87	0.103,70	0.140,74
Predefined List	Fix200	0.430,31	0.453,33	0.112,96	0.188,88
	PBM_α_l	0.451,82	0.468,36	0.116,66	0.211,11
	PBM_α_c	0.539,31	0.559,74	0.127,77	0.240,74
	PBM_α_v	0.529,79	0.536,59	0.127,77	0.240,74
	PBM_α_r	0.490,91	0.511,57	0.122,22	0.229,62
	PBM_Adaptive	0.556,28	0.575,17	0.127,77	0.240,74

outperforms the approach using the NER-generated list in the selected measures. Although the NER method produced worse results, it has its merits. In some cases, we noticed that, when using this method, the experts returned in the results were, in fact, strongly related to the query, even though they were not part of the relevance judgement set. Figure 5.6 shows an example of the results for the query "Computational Intelligence Games". Based on the relevance judgement set, the correct answer for this query set is "Simon Lucas". Some of the names that appeared at the top of the result set were other experts at the University of Essex, PhD researchers and experts from other institutions.

Second, the *PBM_Adaptive* model, which utilises multiple document features to generate an adaptive window for each document, significantly outperformed the baseline approach that used a fixed window size. This was the case in both candidate list scenarios. We tested the statistical significance of this finding using paired t-tests applied to MAP with $p<0.001$. The P@5 values gave a clear indication of the sensitivity of this measure arising from the very low number of relevant entities for each query in the Essex dataset, on average 1.3 per query.

The *PBM_Adaptive* model also outperformed all of the adaptive window models which considered only a single document feature.

It was also observed that the adaptive window model, PBM_α_c, which considered only the number-of-candidates feature performed the best out of all of the single-feature models when using a predefined list of candidates. This finding was in line with other experiments. However, it was interesting to note that the reverse was found when the list of candidates was generated using the NER method. This could be due to the noise injected when generating the candidate list using the NER method. The average size of the candidate list generated with the NER method was 2,664 names, approximately three times larger than the predefined list of 881 names.

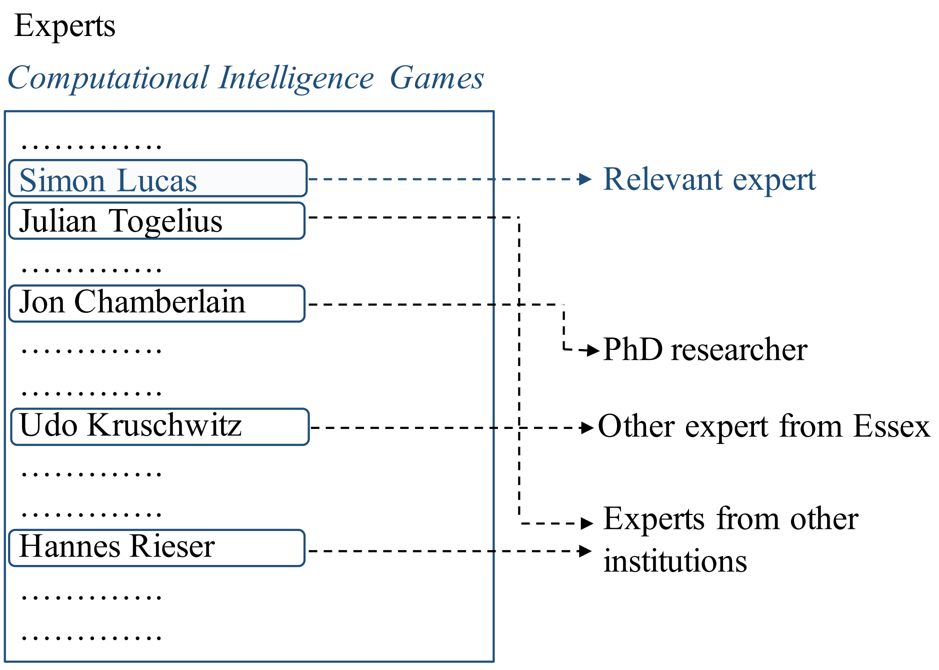


Figure 5.6: An example of the ranked list of experts for the “Computational Intelligence Games” query generated using the NER method.

5.3 Entity Finding Experiment: ClueWeb09 Collection

The goal of entity search is to retrieve a list of relevant entities for a given query. It is similar to expert finding in that both tasks provide users with concise lists of answers. However, in entity search the answers are not limited to people; they could be any type of entity such as organisations, products, and locations etc.

Many of the models developed for expert finding have found their place in entity finding searches, including generative language modelling [Weerkamp et al., 2009], statistical approaches [Fang and Si, 2015], Document-Centered and Entity-Centered models [Wang et al., 2010], and voting models [Santos et al., 2010]. Many systems also depend heavily on the use of external sources such as Wikipedia to identify and extract entities [Kaptein et al., 2010].

Our system architecture for the entity finding experiment was designed as a pipeline. We outline the retrieval framework in Figure 5.7.

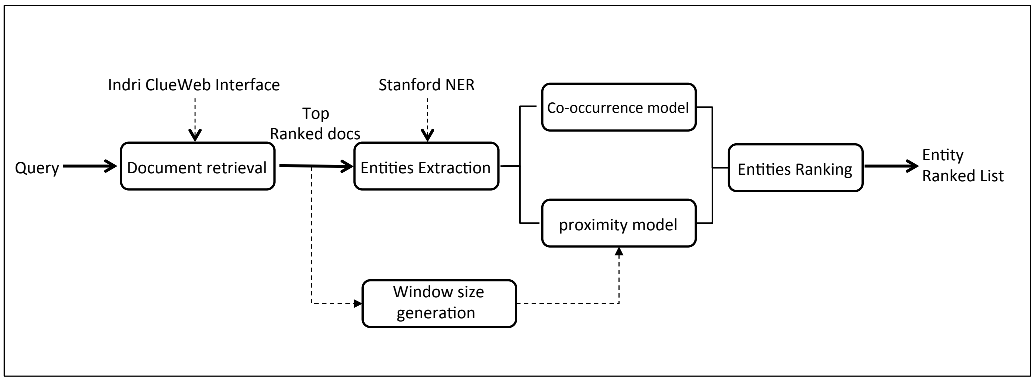


Figure 5.7: General system architecture. The boxes represent process steps. The dotted lines illustrate the external tools that plugged into the system.

The pipeline began with a TREC query. We analysed the narrative part of this query by applying the *OpenNLP* part-of-speech tagger and extracted the keywords and terms mentioned in this part for use at the query expansion stage. We retrieved 50 documents for each query using the online interface provided by the Lemur Project[4]. This number of documents was chosen, following the suggestion of Vechtomova and Robertson [2012], as it is a reasonably manageable number of documents for subsequent in-depth analysis that still generates a sufficient amount of text from which to extract entities. Next, we employed the Stanford Parser to identify entities of the same target type for each retrieved document. The list of entities on each page was considered to be part of the candidate list for this query. We retained a master list of candidate entities, which was the aggregate of all lists for the current query. Finally, the system re-ranked the master list and returned the entities.

When working with the Clueweb Category B dataset, we used the Stanford Named Entity Tagger to recognise entities [Finkel et al., 2005]. This tagger recognises three main entity types, namely PERSON, ORGANISATION, and LOCATION, using a linear chain Conditional Random Field (CRF) sequence model. The classifier was trained on data from CoNLL, MUC-6, MUC-7, and ACE named entity corpora, resulting in a fairly robust model across domains. However, this tagger does not recognise "product" entities as the annotated training corpora do not contain product entities. Thus, for product entities, we selected the entities tagged by the Stanford NER Tagger to be either a person or an organisation. This clearly produces some noise by including non-relevant entity types in the candidate list.

The Category B subset of the ClueWeb09 dataset[5] includes approximately 50 million documents. The testbed includes 20 topics with their relevance judgements. To compare our system against a competitive baseline, we used the proximity-based model, Equation 3.5, with a fixed window size. We also com-

[4]`http://boston.lti.cs.cmu.edu/Services/clueweb09_batch/`
[5]`http://lemurproject.org/clueweb09/`

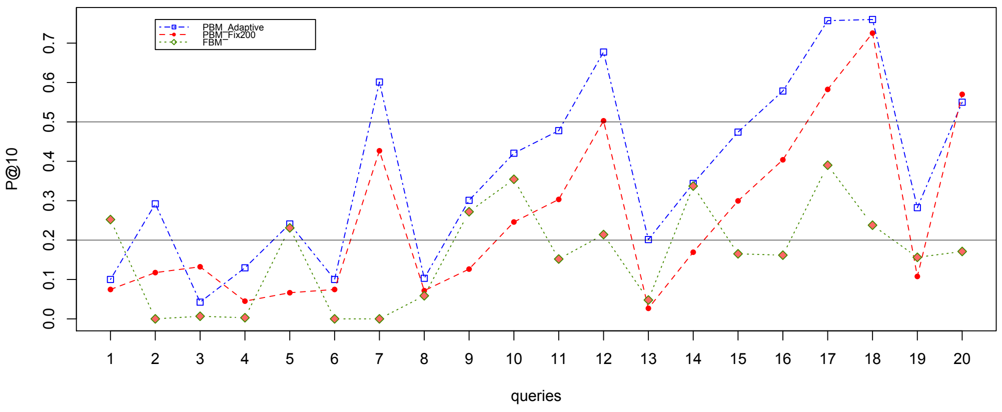


Figure 5.8: P@10 of our method, *PBM_Adaptive*, compared to two baselines (i.e., *PBM_Fix*200 and *FBM*) for each topic.

pared the results for our approach with those obtained using the modification of our method, PBM_Adaptive, to consider each document feature separately, as discussed in Section 5.2.2. Finally, we implemented the proposed method (*PBM_Adaptive*), by using the combination of document features which produced the best result during the training process.

When training, we divided the 20 queries into five groups to provide a 5-fold cross validation. We selected a number N of α combinations for the four document features considered. Four discrete α values (between 0 and 1) that summed to 1 were chosen for each combination.

We evaluated the performance of the systems based on three measures: Mean Average Precision (MAP), Mean Reciprocal Rank (MRR), and Precision at 10 (P@10). MAP was used as the primary metric. A summary of the results is shown in Table 5.10.

The results showed that the adaptive window size method made a modest MAP improvement over the baseline. It was also noticeable, when looking at the MAP measure, that the baseline performed better than the adaptive window size calculated on the basis of only one document feature. The MRR and $P@10$

Table 5.10: Summarised results for the adaptive-window method compared to the baselines. The highest scores for each evaluation matrix are typeset in boldface. We measured the statistical significance using a paired t-test and use △ to indicate $p < 0.05$

	PBM_Fix200	$Single\ Feature$				$PBM\_Adaptive$
		PBM_α_l	PBM_α_c	PBM_α_v	PBM_α_r	
MAP	0.2501	0.2418	0.2164	0.2292	0.2219	**0.2619**
MRR	0.7083	0.7395	0.6510	0.7135	0.6718	**0.7500**△
P@10	0.2438	0.2594	0.2347	0.3177	0.2574	**0.3715**△

measurements showed a more significant improvement over the baseline, which may indicate that the adaptive window size method returned reliable results at the top of the retrieved list.

Figure 5.8 enables a comparison between the P@10 measure[6] values for every topic for the adaptive window size method, *PBM_Adaptive*, with those obtained for the two baselines, i.e., the frequency-based baseline, *PBM*, (Equation 3.2) and the proximity-based baseline with a fixed window size of 200, *PBM_Fix200*. We can see that the *PBM_Adaptive* method outperformed the two baselines in 15 topics out of 20. It is interesting to note that *PBM_Fix200* performed particularly well for three topics (7th, 18th and 20th).

In order to obtain a more detailed evaluation of the methods, we compared the MAP and MRR for each entity type separately (see Figure 5.9). This figure shows diverse results based on the entity type. We noticed that, although the entities of the "product" type generally had the lowest results, applying the adaptive window size method enhanced this score by approximately 46% which is much more than for other entity types (e.g., organisation 6%, person 22%).

[6]We selected the P@10 measure so that we could see the results for each topic individually as well as to be able to compare the results with other work.

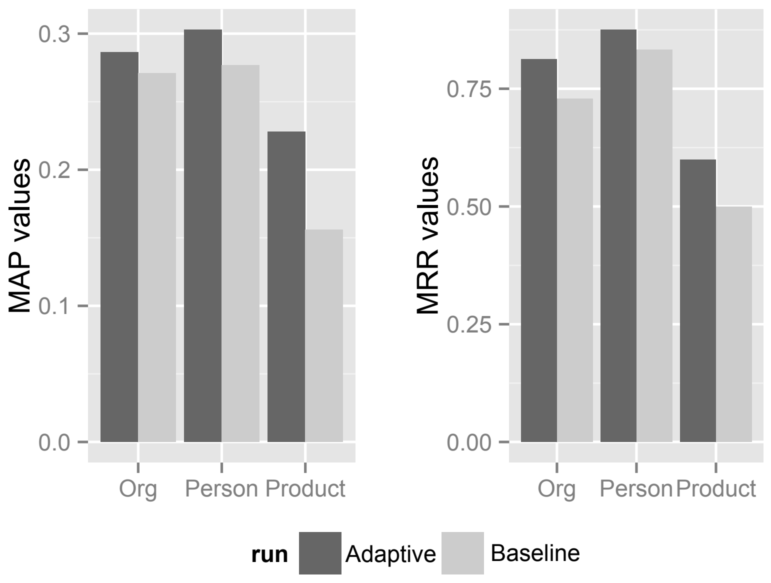


Figure 5.9: MAP and MRR results for the adaptive window size method, $PBM\_Adaptive$, and the baseline, PBM_Fix200, grouped by entity type (i.e., organisation, person, product).

5.4 Exploring Query Expansion and Alternative Document Retrieval Algorithms

Query expansion techniques have proved to have an impact on performance for many retrieval tasks. In this section, we discuss our research on query expansion in the entity finding domain. Our experiment examines a number of methods for query formulation including the thesaurus-based, relevance feedback, and exploitation of NLP structure expansion algorithms. We have incorporated a query expansion component into our entity finding pipeline. We use the CERC collection (section 2.3.1.2) to evaluate a variation of our approach that incorporates this component.

We begin this section with a discussion of the query expansion techniques compared in this experiment. Three main sources have been used for query expansion: the original query, the narrative part, and the retrieved documents, as detailed in Table 5.11.

Table 5.11: Query expansion methods used in this experiment.

Tag	Query expansion method	Data sources
QE1	Thesaurus-based	Query term
QE2	Stop word removal	Narrative Part
QE3	Collocations	Narrative Part
QE4	Relevance feedback	Retrieved documents

The use of different query expansion algorithms may produce disparate influences on the overall performance of expert finding systems. In some cases, applying query expansion can result in a worse performance, as observed in [Wu et al., 2008]. One of the main problems that may have an impact on query expansion for expert finding is *topical drift*, as discussed by Macdonald and Ounis [2007].

The following query expansion methods were evaluated in this experiment:

Relevance feedback

In relevance feedback, users are involved in the retrieval process by giving feedback on the relevance of the documents returned, thereby helping to revise result sets. However, in some scenarios, as in our applications, user involvement tends to be somewhat limited. In such cases, we resort to pseudo-relevance feedback, which automates the process of relevance feedback by assuming that the top k ranked documents[7] are relevant. Using this assumption, one can revise the initial result set. In this experiment, we used the Rocchio algorithm [Rocchio, 1971] for

[7]We have used the default value of $k = 10$. It may be possible to further improve the performance of this method by varying k, but we have left this for future work.

relevance feedback. We have incorporated the algorithm implementation for the Lucene search engine[8] into our approach.

Thesaurus expansion

When implementing thesaurus expansion, we used WordNet[9] to automatically extract synonyms. Using this method, we normalised each query by removing the stop words and punctuation. Next, we expanded each term in the query with synonyms and related words.

Stop word removal

This approach is very common in TREC runs [Duan et al., 2008; SanJuan et al., 2008]. After deleting all of the stop words, we used the query's narrative field (see Figure 2.6) to expand the query.

Collocations

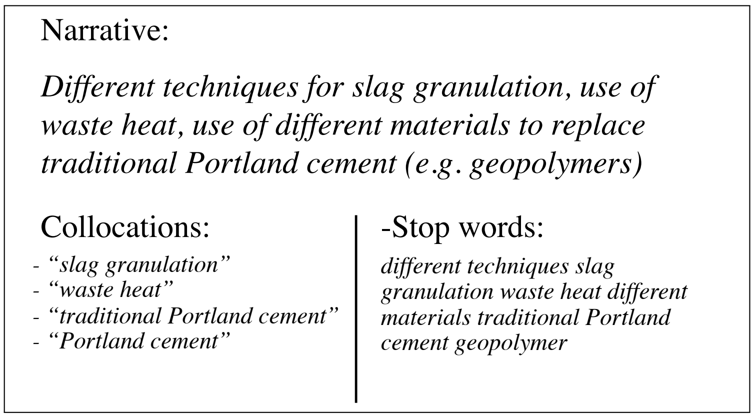
According to Manning and Schütze [1999], "a collocation is an expression consisting of two or more words that correspond to some conventional way of saying things". In this work, we used tag patterns to represent collocations. Table 5.12 outlines the tag patterns (suggested by Justeson and Katz [1995]) that were used in this experiment. Figure 5.10 gives an example of a query expansion using collocation patterns and stop word removal.

We evaluated our expert finding system after integration with different query expansion and document retrieval methods. The results were calculated using two standard metrics: the mean average precision (MAP) and the mean reciprocal rank (MRR). Details of the performance of the expert finding system with respect to these metrics are provided in Table 5.13. We tested three document retrieval algorithms: the Vector Space Model (VSM), Language Modelling (LM), and BM25. The results showed that using different query expansion algorithms appears to have an influence on the overall performance. For instance, with $QE1$,

[8]`http://lucene-qe.sourceforge.net`
[9]`http://wordnet.princeton.edu`

Table 5.12: Parts of speech tag patterns as suggested by Justeson and Katz [1995] where A is an adjective, N is a lexical noun (i.e. not a pronoun) and P is a preposition.

Tag Pattern	Example
A N	linear function
N N	regression coefficients
A A N	Gaussian random variable
A N N	cumulative distribution function
N A N	mean squared error
N N N	class probability function
N P N	degrees of freedom


Narrative:

Different techniques for slag granulation, use of waste heat, use of different materials to replace traditional Portland cement (e.g. geopolymers)

Collocations:

- *"slag granulation"*
- *"waste heat"*
- *"traditional Portland cement"*
- *"Portland cement"*

-Stop words:

different techniques slag granulation waste heat different materials traditional Portland cement geopolymer

Figure 5.10: Comparison between using collocation patterns and removing stop words in the narrative part of the query CE-017.

the performance declined by approximately 4% to 10% when compared to the baseline, $QE0$, in which no query expansion was used. Of the query expansion techniques used in this experiment, $QE3$ proved to have the greatest impact with an improvement of up to 14.8% over the baseline. On the other hand, it was clear that the document retrieval algorithms had a slight effect, which may indicate the stability of the retrieval model. However, when we compared the three document retrieval algorithms, the improvement over the baseline for the results obtained using the language-modelling algorithm was marginally better than that for the other algorithms. The results also suggested that, generally, the percentage improvements in the MRR results were slightly higher than the percentage improvements in the MAP results. Our results seemed to be consistent for all test collections, with no clear pattern emerging as to which was more sensitive to the query expansion methods employed.

Table 5.13: Results for the implementation of different query expansion techniques in the expert finding system. The best scores are in boldface. The significance for each document retrieval algorithm is tested against the baseline using a two-tailed paired t-test. We use △ to indicate $p < 0.01$.

			TREC 2007		TREC 2008	
			MAP	MRR	MAP	MRR
VSM	$QE0$	(baseline)	0.5022	0.7313	0.4464	0.7956
	$QE1$		0.4804	0.7145	0.4351	0.7740
	$QE2$		0.5102	0.7741	0.4618	0.8434
	$QE3$		0.5242	0.7957△	0.4776△	0.8495△
	$QE4$		0.5137	0.7807△	0.4724	0.8411△
LM	$QE0$	(baseline)	0.5089	0.7336	0.4514	0.7976
	$QE1$		0.4907	0.7117	0.4441	0.7726
	$QE2$		0.5252	0.8071△	0.4639	0.8517△
	$QE3$		**0.5937**△	**0.8766**△	**0.5140**△	0.9228△
	$QE4$		0.5819△	0.8583△	0.5084	0.9137△
BM25	$QE0$	(baseline)	0.5210	0.7634	0.4573	0.8161
	$QE1$		0.4975	0.6810	0.4455	0.7553
	$QE2$		0.5535	0.7440	0.4763	0.8656△
	$QE3$		0.5662△	0.8157△	0.5022	**0.9376**△
	$QE4$		0.5605	0.7978	0.4962	0.9293△

The results in the above table show that the best performance was achieved using the collocation query expansion and language modelling document retrieval algorithms.

5.5 Summary

In this chapter we presented an experimental evaluation of our adaptive window size approach to the entity and expert finding tasks. We began by performing a pilot study to confirm the validity of our approach, Section 5.1. The W3C dataset and test collection were used in this study. Three document features (the document-length, number-of-candidates, and average-sentence-size) were used in the calculation of the size of the adaptive window for use with our model. The performance of our model was compared with the performance of a number of baseline models. The results of the pilot study showed that our model does improve on these baseline models. They also showed that the behaviour of our model changes when each document feature is used, in isolation, to calculate the size of the adaptive window. This finding motivated us to investigate the incorporation of additional document features into the calculation of the adaptive window size. Subsequent experiments incorporated a fourth feature, namely the readability-index, into this calculation.

We next evaluated the adaptive window size approach using a variety of data sets for expert finding in Section 5.2, and entity finding, in Section 5.3. The combination of these results shows us that the adaptive window size approach generalizes well across datasets.

Finally, in Section 5.4, we investigated further refinements that could be made to our approach, including query expansion techniques and different document retrieval algorithms. Of the different query expansion techniques evaluated, it was found that the collocations method resulted in the greatest improvement in performance. We also found that language modelling document retrieval algorithms produced better outcomes.

While this chapter discussed the results for each experiment independently, the next chapter will look more generally at the results across all experiments in order to investigate the overall improvement over the baseline afforded by our approach. Moreover, we shall reflect on the different document features used in our model and discuss the effects of using other languages on the performance of our approach. We shall conclude with a consideration of some of the limitations of our work.

6
Discussion

In this chapter we evaluate the performance of the adaptive window size method for entity finding on the overall retrieval process. As shown in the previous sections, the proposed method generally improves the entity finding system by combining different document features to generate a unique window size for each document. We compare the different methods used in this thesis, and discuss the implications and limitations of our research.

6.1 Method Comparison

An analysis of our findings shows that the proposed adaptive window size method performs better than methods incorporating fixed window sizes on the same test collection. In Table 6.1, we present the percentage change in the system performance obtained by applying our method, rather than a proximity based method with a fixed window size of 200 words. We calculate the percentage improvement using the Mean Average Precision (MAP) and Mean Reciprocal Rank (MRR) metrics. The MAP metric shows an average improvement across all datasets of about 11% with our proposed approach, whereas the MRR metric shows an average improvement for all datasets except the UvT, when searching for Dutch queries, of about 8.5%.

Table 6.1: Percentage change between the PBM_Adaptive method and the baseline, PBM_Fix200 method, for the Mean Average Precision (MAP) and Mean Reciprocal Rank (MRR) measures.

		MAP	MRR
W3C	2005	+17.24%	+6.55%
	2006	+10.26%	+8.66%
CERC	2007	+9.92%	+18.66%
	2008	+11.19%	+1.78%
UvT	English	+13.84%	+15.81%
	Dutch	+13.63%	−0.4%
TU-GT1	English	+23.1%	+18.15%
	Dutch	+16.91%	+11.71%
TU-GT2	English	+14.49%	+10.6%
	Dutch	+8.3%	+4.94%
TU-GT3	English	+15.38%	+13.62%
	Dutch	+4.79%	+2.32%
TU-GT4	English	+7.56%	+7.01%
	Dutch	+9.07%	+7.87%
Essex	NER	+55.99%	+66.52%
	Predefined	+29.30%	+26.93%
Entity		+4.72%	+5.89%

6.2 Document features

We evaluate the use of four document features to determine the size of the adaptive window, namely: *document_length*, *number_of_candidates*, *average_sentence_size*, and *readability_index*. A comparison of these features suggests that the *number_of_candidates* has the strongest impact on the window size, and so it has been given an increased weight in our formula for the adaptive window size. This means that documents with a high density of candidate evidence are associated with larger proximity windows. There are a number of exceptions, such as when the accuracy of the identification of candidate experts is unclear, for instance, when we do not have a pre-defined list of candidates as was the case when working with the ClueWeb09 collection, or when dealing with a different language. These exceptions will be discussed in further detail later in this chapter. By contrast, the *readability_index* feature generally has a low impact. We found that tuning the window size based on this feature provides only very limited improvement in performance.

Our analysis also showed that the *document_length* feature has only a weak impact on performance with the W3C, CERC and UvT datasets. However, it shows a stronger effect on search performance with ClueWeb09. This could reflect the nature of the test collections, as ClueWeb09 contains documents of greatly varying lengths, whereas the other collections are more homogeneous with respect to the length of their documents. The final feature, *average_sentence_size*, has an average impact on all collections.

It has been established through previous experiments that the best results are obtained when these features are combined. In fact, in some cases, actually applying the features individually may produce worse results than the fixed window baseline. This can be seen in Table 5.10.

6.3 Language

In previous experiments, we observed that the influence of the document features was affected by a number of factors, with one of the most significant being the language of the collection. For instance, in the UvT experiment, variations between the results for the English and Dutch queries could be seen. In Figure 6.1, we display the MAP values for various combinations of α. The coloured bars on the x-axis represent the combinations of α values used in each trial. The contribution of each α value is depicted using a different colour. The percentage contribution of each α value used in the combination is indicated by the proportion of the label made up of that colour. Where there are fewer than four coloured areas in a label, the missing coloured areas correspond to α values which are set to zero in the corresponding α combination. The y-axis shows the MAP value obtained when the experiment was run with the window size determined using the α combination represented by the x-label. For instance, the best MAP value for the English collection was 0.264 which was obtained with the α combination of $\alpha_l = 0.1$, $\alpha_c = 0.6$, $\alpha_v = 0.0$, and $\alpha_r = 0.3$. It can be seen that the *number_of_candidates* and *document_length* have the strongest impacts on performance when searching for English queries, whereas *average_sentence_size* and *document_length* are the most influential for Dutch queries.

Our analysis of the UvT experiment (Section 5.2.2) showed that our method performed better when applied to English queries. We have also noticed disparate effects for the document features, depending on the language of the documents and queries. Although the application of our method resulted in an improvement over the baselines, this improvement was smaller than that observed for previous datasets, Sections 5.2.1 and 5.1. One of the reasons for this could lie in the nature of the dataset. A large proportion of the UvT dataset consists of highly

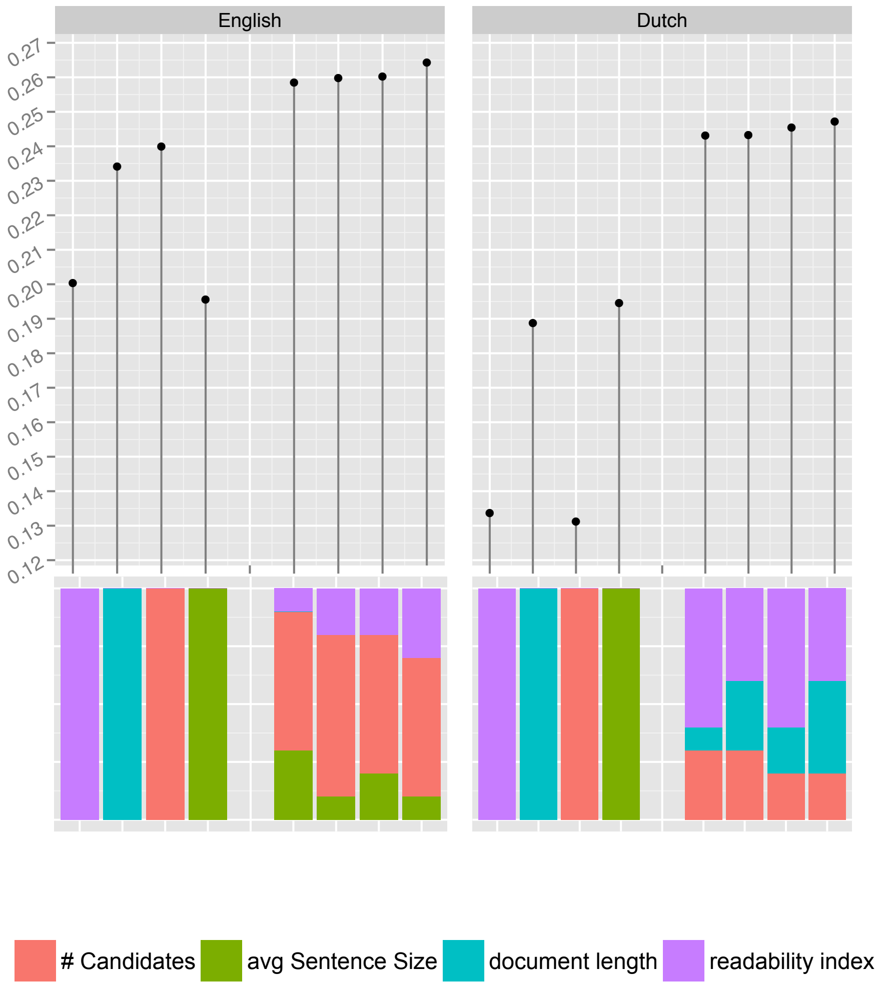


Figure 6.1: The MAP results for the adaptive window size method using a number of feature-combinations and single features for English and Dutch queries on the UvT collection.

structured documents with rich metadata that our method did not utilise. Alternatively, this could have been due to the lack of language-specific normalisation prior to processing the documents. Many other issues arise when dealing with different languages, but we leave these for future work and further experiments.

6.4 Limitations

Our findings are subject to a number of limitations. One such limitation is that our approach does not take into account the wealth of additional information and other characteristics that individual documents may offer. Many expert finding models have benefited from considering special document aspects such as document structure, or candidate aspects such as social networks existing between candidates. We chose not to incorporate such aspects into our model as we aimed to design a general approach. For instance, to incorporate document structure, we would need to consider only a specific type of document. Moreover, it would be hard to generalise our approach from expert-finding to entity-finding if we constructed and incorporated the candidate's social network into the model. However, our approach may not be as accurate as other approaches which consider these features.

The second limitation is that our approach may not be as efficient as approaches which use passage retrieval to identify specific parts of documents relevant to a query. The document retrieval process used in this thesis (see section 2.1) employs a similarity measure which is applied to whole documents to estimate how relevant each (whole) document is to the user's query and returns a list of relevant documents as an answer to this query. However, other retrieval strategies such as passage retrieval [Kaszkiel and Zobel, 2001] exist for answering the user's query and these may be more efficient, both in terms of system performance and in satisfying the user's information need. In passage retrieval, each document in the collection is regarded as a set of passages. Each one of these passages is a contiguous block of text. Instead of computing the similarity for each document as a whole, the similarity is computed for each passage. User queries are then answered with a list of passages, rather than a list of documents. Passage retrieval has been shown to be effective in IR situations, particularly when documents are long or include multiple changes of topic. Our approach is not as effective in these situations as it returns only a list of whole documents that gives the user little indication of where the desired information may be found in each document.

The techniques used for the evaluation of our approach are another limitation of the work presented in this thesis. Our systems were evaluated using only the Cranfield approach [Voorhees, 2002] to offline evaluation. In this approach, a set of pre-judged queries is used to determine the effectiveness of a retrieval approach on a given test collection. Perhaps a more effective approach is to use online evaluation. In online evaluation, typical users evaluate the retrieval system by using it in a controlled setting. This approach was rarely used until recently as it was expensive and difficult to complete correctly. The main obstacle lay in the difficulty of gathering a sufficiently large and representative sample of actual users of the retrieval system. The situation has changed with the development of crowdsourcing methods that facilitate the gathering of such large representative samples from online communities. As online evaluation is now a more feasible approach, some questions have been raised about the validity of offline evaluation for analysing retrieval approaches [Peters et al., 2012].

A number of other limitations of the work presented in this thesis are apparent. Some of these have arisen from the datasets that have been used for our experiments. Generally, we have used general-purpose test collections to evaluate our approach. Consequently, every experiment that we have performed can only be generalised within the context of this data. The results for each experiment may only provide a limited insight into the performance of the system on data that goes beyond the specific collection tested. In addition, we have only considered a limited number of types of entities in our experiments. It is unclear whether our approach will provide the same level of performance improvement for other types of entities. Further limitations are caused by the way in which we have set the parameters for the experiments. In our experiments, we set our parameters to discrete values which are cut off at step points. This is only one way of exploring the search space: genetic algorithms and regression analysis, for instance, may provide better alternatives for parameter selection.

6.5 Summary

In this chapter, we considered some general issues in relation to our adaptive window size approach. We first discussed the method's overall performance across the different datasets used in this thesis, thereby analysing the overall improvement over the baseline performance achieved when applying the adaptive window size approach. We then discussed the document features which we have used to calculate the window size for the main component of our adaptive window size approach. Next, we considered the generalisation of our approach to other languages and demonstrated how this changes the relative importance of different document features in determining the optimal adaptive window size. Finally, we considered some the limitations of our work. In the next chapter, we shall elaborate on the overall conclusions that may be drawn from our work, revisit our research questions, and discuss some directions for future research.

7 Conclusion

In this chapter, the key findings and conclusions from our research are summarised. We review the approaches developed during the research and the observations made while answering the research questions. Finally, we discuss possible future directions for research and the potential to extend our results.

The main motivation for this research was to develop an adaptive, well-founded, general-purpose approach to entity finding. Entity finding applications consider a number of different types of entities such as products, organisations, and locations. In this thesis, we mainly focus on expert finding, but we also demonstrate in Section 5.3 that our proposed method could also be applied to other types of entities.

The entity finding task was addressed by using a state-of-the-art two stage model. The first stage of this model comprised the retrieval of documents relevant to the user's query. In the second stage, the most relevant entities were extracted from these documents. The association between the candidate entities and the query terms was defined, based on both the frequency of the candidate evidence in the relevant documents and the proximity between the query terms and candidate evidence in these documents.

In Part I of this thesis, an adaptive window size model was introduced. This model was used to enhance the proximity association between candidate entities and queries. The model utilised a number of document features in order to select the window sizes for proximity searches in each document. The document features considered were the average sentence size, number of entities in the document, document length, and a readability index.

Part II of the thesis focused on the thorough evaluation of the adaptive window size model on a variety of datasets. In particular, the W3C, CERC, UvT/TU, and Essex datasets, and the ClueWeb09 entity finding collection were used. Our approach was tested using two languages (i.e., English and Dutch) and in two different applications, namely, Expert and Entity Finding. The impact of different document features on the window size differed, depending upon the nature of the test collection employed. Experimental results showed that our proposed method not only out-performed baseline methods across the aforementioned datasets, but it also demonstrated the potential to use more generalised input data arising from varying environments, languages, and applications.

7.1 Answers to Research Questions

We can now return to our original research questions.

RQ1: Can the state-of-the-art in entity finding be pushed forward by employing document features to determine the size of the window in proximity-based approaches?

> We have been able to answer this question positively by comparing the results of applying a state-of-the-art entity finding model with a fixed window size to those obtained with the adaptive window approach.
>
> Our experiments showed that the use of the adaptive window method provided a significant improvement over the baselines. We initially used three document features to generate the adaptive window. These were the document length, number of candidates, and the average sentence size. This led to an investigation of the use of a fourth feature, namely, a readability index, (Sections 5.2 and 5.3). We found that individual features, when considered in isolation, did not give as much of an improvement as a combination of features. The application of the combination of the four features considered in this thesis allowed for the confirmation of our original findings, and provided a significant improvement over the current baselines. This,

in turn, supported an affirmative answer to this research question.

RQ2: Will an adaptive-window-size approach to entity finding be robust across different types of named entities and different types of document collections?

The experimental results presented in Chapter 5 demonstrate the consistency of the behaviour of the developed models across document collections. We evaluated our approach on document collections of different types. For instance, we tested our model on an enterprise collection (Section 5.2.1), and in academic environments (Sections 5.2.2 and 5.2.3), as well as on an internet collection of a much larger scale (Section 5.3).

In addition to evaluating our model on a variety of collection types, we tested our proposed method on documents in different languages. In particular, we investigated the use of a dataset that was bilingual in English and Dutch (Section 5.2.2). The results obtained clearly showed that our approach has the potential to be applied to document collections and queries in different languages.

Although most of the experimental evaluation was carried out on the task of finding people, we also evaluated our approach on other named entities, including organisation and product entities (Section 5.3).

The results of our investigations suggest that our approach generalizes effectively across different environments, document types, languages, and entity types.

7.2 Future work

Although we have successfully addressed our research questions, a number of questions remain unanswered and new questions continue to emerge. Here, we discuss several avenues for future research that have arisen during the course of our research.

Our work involved the development of an adaptive window size approach to entity finding in which three document features were considered for the calculation of the window size for each document. Based on our preliminary evaluation of this system, we decided to investigate the inclusion of an additional document feature into our model. It would be interesting to seek other document features that may have an impact on the window size. In particular, we plan to consider features related to the types of data being considered and to assess their effects on the window size. For example, data obtained from social media might have specific features that could be incorporated into the window size calculation. Moreover, it might be interesting to consider a group of features that could be employed to measure the level of formality expressed in each document.

Although we evaluated our method on an English-Dutch bilingual dataset (Section 5.2.2), we did not implement any special language-dependent processing for the documents. It would be interesting to determine whether incorporating specific preprocessing steps for the Dutch language (e.g. Dutch stop words, stemming, etc.) would yield more informative results. We plan to further investigate the adaptive window-size approach on document collections in other languages such as Arabic and to determine whether the approach provides similar performance enhancements. This investigation will enable us to identify whether there are other language-specific document features that can be incorporated into our model in order to enhance the performance of its application to document collections in other languages.

Our current work has a number of limitations (Section 6.4). Each of these limitations could give rise to a direction for future work. For instance, we could incorporate more user-specific data when implementing our entity search system. To do this we could incorporate social aspects into our approach to see whether this would improve our findings to a degree that justifies the complications injected into the system by adding such extra parameters. We could address another limitation in future work by incorporating a passage retrieval component into the underlying search engine in place of the current document retrieval component. Passage retrieval has already been implemented and has proven very successful for similar information retrieval tasks such as Question Answering [Dumais et al., 2002; Tellex et al., 2003]. Finally, we could perform a more rigorous evaluation of our approach by running a set of user-oriented (online) evaluations. According to Peters et al. [2012], system-oriented (offline) IR evaluation is not sufficient by itself, and user-oriented evaluations are equally important when producing an effective operational system. This is because user-oriented evaluations would consider factors other than system performance alone.

References

Ackerman, M. S. and Halverson, C. (2004). Sharing expertise: The next step for knowledge management. In Wulf, V. and Huysman, M., editors, *Social Capital and Information Technology*. MIT Press, Cambridge MA. 24

Ackerman, M. S. and McDonald, D. W. (1996). Answer Garden 2: merging organizational memory with collaborative help. In *CSCW '96: Proceedings of the 1996 ACM Conference on Computer Supported Cooperative Work*, pages 97–105, Boston, MA. ACM. 23, 26

Ackerman, M. S., Pipek, V., and Wulf, V. (2003). *Sharing expertise: Beyond knowledge management*. MIT press, Cambridge MA. 23

Alarfaj, F., Kruschwitz, U., Hunter, D., and Fox, C. (2012). Finding the Right Supervisor: Expert-finding in a University Domain. In *NAACL HLT '12: Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Student Research Workshop*, pages 1–6. Association for Computational Linguistics. 46

Allan, J., Aslam, J., Belkin, N., Buckley, C., Callan, J., Croft, B., Dumais, S., Fuhr, N., Harman, D., Harper, D. J., Hiemstra, D., Hofmann, T., Hovy, E., Kraaij, W., Lafferty, John amd Lavrenko, V., Lewis, D., Liddy, L., Manmatha, R., McCallum, A., Ponte, J., Prager, J., Radev, D., Resnik, P., Robertson, S., Rosenfeld, R., Roukos, Salim Sanderson, M., Schwartz, R., Singhal, A., Smeaton, A., Turtle, H., Voorhees, E., Weischedel, R., Xu, J., and Zhai, C. (2003). Challenges in information retrieval and language modeling: report of a workshop held at the center for intelligent information retrieval, university of massachusetts amherst, september 2002. *ACM SIGIR Forum*, 37(1):31–47. 22

Armstrong, S. J. (2004). The impact of supervisors' cognitive styles on the quality of research supervision in management education. *British Journal of Educational Psychology*, 74(4):599–616. 108

Aslam, J. A., Yilmaz, E., and Pavlu, V. (2005). A geometric interpretation of r-precision and its correlation with average precision. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 573–574, Salvador, Brazil. ACM. 54

Aslay, Ç., O'Hare, N., Aiello, L. M., and Jaimes, A. (2013). Competition-based networks for expert finding. In *SIGIR '13: Proceedings of the 36th International ACM SIGIR conference on research and development in Information Retrieval*, pages 1033–1036, New York, NY, USA. ACM. 24

Azzopardi, L., Balog, K., and de Rijke, M. (2005). Language Modeling Approaches for Enterprise Tasks. In *Proceedings of the 14th Text REtrieval Conference (TREC-14)*, volume 500-266 of *NIST Special Publication*, Gaithersburg, Maryland. 37

Baeza-Yates, R. and Ribeiro-Neto, B. (2011). *Modern Information Retrieval - the concepts and technology behind search*. Addison Wesley, Harlow, UK, 2nd edition edition. 13, 14, 18, 53, 87

Bailey, P., Craswell, N., Soboroff, I., and de Vries, A. P. (2007a). The CSIRO enterprise search test collection. *ACM SIGIR Forum*, 41(2):42–45. xiixii, 42, 164

Bailey, P., de Vries, A. P., Craswell, N., and Soboroff, I. (2007b). Overview of the TREC 2007 Enterprise Track. In *Proceedings of the 16th Text REtrieval Conference (TREC-16)*, volume 500-272 of *NIST Special Publication*, Gaithersburg, Maryland. 22, 56, 100

Balog, K., Azzopardi, L., and de Rijke, M. (2006). Formal models for expert finding in enterprise corpora. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 43–50, Seattle, USA. ACM. 23, 27, 29, 106

Balog, K., Azzopardi, L., and de Rijke, M. (2009a). A language modeling framework for expert finding. *Information Processing and Management*, 45(1):1–19. 73

Balog, K., Bogers, T., Azzopardi, L., de Rijke, M., and van den Bosch, A. (2007). Broad expertise retrieval in sparse data environments. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 551–558, Amsterdam. ACM. 44

Balog, K. and de Rijke, M. (2008a). Combining candidate and document models for expert search. In *Proceedings of the 17th Text REtrieval Conference (TREC-17)*, volume 500-277 of *NIST Special Publication*, Gaithersburg, Maryland. 58, 103

Balog, K. and de Rijke, M. (2008b). Non-local evidence for expert finding. In *CIKM '08: Proceedings of the 17th ACM conference on Information and knowledge management*, pages 489–498, New York, USA. ACM. 60

Balog, K., de Vries, A. P., Serdyukov, P., Thomas, P., and Westerveld, T. (2009b). Overview of the TREC 2009 Entity Track. In *Proceedings of the 18th Text REtrieval Conference (TREC-18)*, volume 500-278 of *NIST Special Publication*, Gaithersburg, Maryland. 51, 100

Balog, K., Serdyukov, P., and Vries, A. P. d. (2010). Overview of the TREC 2010 Entity Track. In *Proceedings of the 19th Text REtrieval Conference (TREC-19)*, volume 500-294 of *NIST Special Publication*, Gaithersburg, Maryland. 22

Balog, K., Thomas, P., Craswell, N., Soboroff, I., Bailey, P., and de Vries, A. P. (2008). Overview of the TREC 2008 Enterprise Track. In *Proceedings of the 17th Text REtrieval Conference (TREC-17)*, volume 500-277 of *NIST Special Publication*, Gaithersburg, Maryland. xiixii, 22, 56, 100, 165

Bao, S., Duan, H., Zhou, Q., Xiong, M., Cao, Y., and Yu, Y. (2008). A probabilistic model for fine-grained expert search. In *ACL'08: Proceedings of the 46rd Annual Meeting on Association for Computational Linguistics*, pages 914–922, Columbus, Ohio. Association for Computational Linguistics. 61, 92

Bao, S., Duan, H., Zhou, Q., Xiong, M., Yu, Y., and Cao, Y. (2006). Research on Expert Search at Enterprise Track of TREC 2006. In *Proceedings of the 15th Text REtrieval Conference (TREC-15)*, volume 500-272 of *NIST Special Publication*, Gaithersburg, Maryland. 37, 98

Becerra-Fernandez, I. (2000a). Facilitating the online search of experts at NASA using Expert Seeker People-Finder. In *Third International Conference on Practical Aspects of Knowledge Management.* 23

Becerra-Fernandez, I. (2000b). The role of artificial intelligence technologies in the implementation of people-finder knowledge management systems. *Knowledge-Based Systems*, 13(5):315–320. 24

Berendsen, R., Rijke, M. d., Balog, K., Bogers, T., and Bosch, A. v. d. (2013). On the assessment of expertise profiles. *Journal of the American Society for Information Science and Technology*, 64(10):2024–2044. xixi, 45, 100

Berger, A. and Lafferty, J. (1999). Information retrieval as statistical translation. In *SIGIR '99: Proceedings of the 22ed International ACM SIGIR conference on research and development in Information Retrieval*, pages 222–229, Berkeley, USA. ACM. 20

Blanco, R., Cambazoglu, B. B., Mika, P., and Torzec, N. (2013). Entity recommendations in web search. In *Proceedings of the 12th International Semantic Web Conference*, volume 8219, pages 33–48. Springer, Sydney, Australia. 4

Borgatti, S. P. and Cross, R. (2003). A relational view of information seeking and learning in social networks. *Management Science*, 49(4):432–445. 22, 35

Brown, P. F., Cocke, J., Pietra, S. A. D., Pietra, V. J. D., Jelinek, F., Lafferty, J. D., Mercer, R. L., and Roossin, P. S. (1990). A statistical approach to machine translation. *Computational linguistics*, 16(2):79–85. 20

Buckley, C. and Voorhees, E. M. (2000). Evaluating evaluation measure stability. In *SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 33–40, Athens, Greece. ACM. 54

Buckley, C. and Voorhees, E. M. (2004). Retrieval evaluation with incomplete information. In *SIGIR '04: Proceedings of the 27th annual international ACM*

SIGIR conference on Research and development in information retrieval, pages 25–32, Sheffield, United Kingdom. ACM. 55

Bush, V. (1945). As we may think. *The Atlantic Monthly*, 176(1):101–108. 15

Büttcher, S., Clarke, C. L., and Cormack, G. V. (2010). *Information retrieval: Implementing and evaluating search engines*. MIT Press. 18

Cao, Y., Liu, J., Bao, S., and Li, H. (2005). Research on Expert Search at Enterprise Track of TREC 2005. In *Proceedings of the 14th Text REtrieval Conference (TREC-14)*, volume 500-266 of *NIST Special Publication*, Gaithersburg, Maryland. 60, 69

Cheng, Z., Caverlee, J., Barthwal, H., and Bachani, V. (2014). Finding local experts on twitter. In *WWW'14*, pages 241–242. 24

Cleverdon, C. (1967). The cranfield tests on index language devices. *Aslib proceedings*, 19(6):173–194. 16, 38

Cleverdon, C. W. (1991). The significance of the cranfield tests on index languages. In *SIGIR '91: Proceedings of the 14th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3–12, Chicago, IL, USA. ACM. 38

Cleverdon, C. W. and Keen, M. (1966). Factors affecting the performance of indexing systems. *ASLIB, Cranfield Research Project. Bedford*, 2:3759. xixi, 52

Clough, P. and Sanderson, M. (2013). Evaluating the performance of information retrieval systems using test collections. *Information Research*, 18(2). 39

Consens, M. P. and Baeza-Yates, R. (2005). Database and information retrieval techniques for XML. In *Proceedings of the 10th Asian Computing Science Conference*, pages 22–27. Springer, Kunming, China. 14

Craswell, N., de Vries, A. P., and Soboroff, I. (2005). Overview of the TREC 2005 Enterprise Track. In *Proceedings of the 14th Text REtrieval Conference (TREC-14)*, volume 500-266 of *NIST Special Publication*, Gaithersburg, Maryland. xiixii, 23, 41, 56, 95, 163

Craswell, N., Hawking, D., Vercoustre, A., and Wilkins, P. (2001). P@noptic expert: Searching for experts not just for documents. In *Ausweb Poster Proceedings, Queensland, Australia.* 26

Croft, B. and Lafferty, J. (2003). *Language modeling for information retrieval.* Springer Netherlands. 19

Croft, W. B., Metzler, D., and Strohman, T. (2015). *Search engines: Information retrieval in practice.* 13, 17, 20, 87, 88

Davenport, T. H. and Prusak, L. (2000). *Working knowledge: How organizations manage what they know.* Harvard Business Press, 2ed edition edition. 24

De Kretser, O. and Moffat, A. (1999). Effective document presentation with a locality-based similarity heuristic. In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 113–120, Berkeley, USA. ACM. 61

Deerwester, S. C., Dumais, S. T., Landauer, T. K., Furnas, G. W., and Harshman, R. A. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science and Technology*, 41(6):391–407. 16

Demartini, G., Iofciu, T., and de Vries, A. P. (2010). Overview of the INEX 2009 entity ranking track. In *INEX '09: Proceedings of the 8th International Workshop of the Initiative for the Evaluation of XML Retrieval*, volume 6203, pages 254–264. Springer Berlin Heidelberg, Brisbane, Australia. 22

Duan, H., Zhou, Q., Lu, Z., Jin, O., Bao, S., Cao, Y., and Yu, Y. (2008). Research on Enterprise Track of TREC 2007 at SJTU APEX Lab. In *Proceedings of the 16th Text REtrieval Conference (TREC-16)*, volume 500-274 of *NIST Special Publication*, Gaithersburg, Maryland. 103, 119

Dumais, S., Banko, M., Brill, E., Lin, J., and Ng, A. (2002). Web question answering: Is more always better? In *SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 291–298, Tampere, Finland. ACM. 137

Fang, H. and Zhai, C. (2007). Probabilistic models for expert finding. In *Advances in Information Retrieval, Proceedings of the 29th European Conference on IR Research (ECIR-2007)*, volume 4425 of *Lecture Notes in Computer Science*, pages 418–430. Springer, Rome, Italy. 23, 27, 28, 37

Fang, S. (2012). A survey of teacher-student style mismatches. *Higher Education of Social Science*, 3(1):5–12. 108, 109

Fang, Y. and Si, L. (2015). Related entity finding by unified probabilistic models. *World Wide Web*, 18(3):521–543. 4, 113

Fang, Y., Si, L., and Mathur, A. P. (2011). Discriminative probabilistic models for expert search in heterogeneous information sources. *Information Retrieval*, 14(2):158–177. 104

Finkel, J. R., Grenager, T., and Manning, C. (2005). Incorporating non-local information into information extraction systems by Gibbs sampling. In *ACL*

'05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, pages 363–370, Stroudsburg, PA, USA. Association for Computational Linguistics. 109, 114

Forst, J. F., Tombros, A., and Rölleke, T. (2007). Solving the Enterprise TREC Task with Probabilistic Data Models. In *Proceedings of the 16th Text REtrieval Conference (TREC-16)*, volume 500-274 of *NIST Special Publication*, Gaithersburg, Maryland. 74

Frakes, W. B. (1992). *Introduction to information storage and retrieval systems.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA. 14

Frischmuth, P., Klímek, J., Auer, S., Tramp, S., Unbehauen, J., Holzweißig, K., and Marquardt, C.-M. (2012). Linked data in enterprise information integration. *Semnatic Web.* 5

Fu, Y., Xiang, R., Liu, Y., Zhang, M., and Ma, S. (2007a). A CDD-based formal model for expert finding. In *CIKM '07: Proceedings of the 16th ACM conference on Conference on information and knowledge management*, pages 881–884, New York, USA. ACM. 59

Fu, Y., Xue, Y., Zhu, T., Liu, Y., Zhang, M., and Ma, S. (2007b). THUIR at TREC 2007: Enterprise Track. In *Proceedings of the 16th Text REtrieval Conference (TREC-16)*, volume 500-274 of *NIST Special Publication*, Gaithersburg, Maryland. 58, 103

Fu, Y., Yu, W., Li, Y., Liu, Y., Zhang, M., and Ma, S. (2005). THUIR at TREC 2005: Enterprise Track. In *Proceedings of the 14th Text REtrieval Conference (TREC-14)*, volume 500-266 of *NIST Special Publication*, Gaithersburg, Maryland. 60

Ganesan, K. and Zhai, C. (2012). Opinion-based entity ranking. *Information Retrieval*, 15(2):116–150. 5

George, C. A., Bright, A., Hurlbert, T., Linke, E. C., St. Clair, G., and Stein, J. (2006). Scholarly use of information: graduate students' information seeking behaviour. *Information Research*, 11(4). 108

Ghosh, S., Sharma, N., Benevenuto, F., Ganguly, N., and Gummadi, K. (2012). Cognos: crowdsourcing search for topic experts in microblogs. In *SIGIR '12: Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 575–590, Portland, Oregon, USA. ACM. 24

Goker, A. and Davies, J. (2009). *Information retrieval: Searching in the 21st century*. John Wiley & Sons. 18

Gollapalli, S. D., Mitra, P., and Giles, C. L. (2011). Ranking authors in digital libraries. In *JCDL '11: Proceedings of the 11th Annual International ACM/IEEE Joint Conference on Digital Libraries*, pages 251–254, New York, USA. ACM. 100

Grossman, D. A. and Frieder, O. (2004). *Information retrieval: Algorithms and heuristics*, volume 15. Springer Publishers. 87

Guo, J., Xu, G., Cheng, X., and Li, H. (2009). Named entity recognition in query. In *SIGIR '09: Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 267–274, Boston, USA. ACM. 22

Han, S., He, D., Brusilovsky, P., and Yue, Z. (2013). Coauthor Prediction for Junior Researchers. In *Proceedings of the 6th International Conference on*

Social Computing, Behavioral-Cultural Modeling and Prediction, volume 7812, pages 274–283, Washington, DC, USA. Springer. 22

Hansen, M. T., Nohria, N., and Tierney, T. (1999). What's your strategy for managing knowledge? *Harvard Business Review*, 77(2). 23

Harman, D. (1992). Overview of the first text retrieval conference. In *Proceedings of the 1st Text REtrieval Conference (TREC-1)*, volume 500-207 of *NIST Special Publication*, Gaithersburg, Maryland. 16

Harman, D. (2011). Information retrieval evaluation. *Synthesis Lectures on Information Concepts, Retrieval, and Services*, 3(2):1–119. 38

Hawking, D. (2004). Challenges in enterprise search. In *ADC '04: Proceedings of the 15th Australasian database conference*, pages 15–24, Darlinghurst, Australia. Australian Computer Society, Inc. 5

Hawking, D. and Robertson, S. (2003). On collection size and retrieval effectiveness. *Information Retrieval*, 6(1):99–105. 53

He, B., Macdonald, C., Ounis, I., Peng, J., Santos, R. L., and Santos, R. (2008). University of Glasgow at TREC 2008: Experiments in blog, enterprise, and relevance feedback tracks with Terrier. In *Proceedings of the 17th Text REtrieval Conference (TREC-17)*, volume 500-277 of *NIST Special Publication*, Gaithersburg, Maryland. 103

Hecht, B., Teevan, J., Morris, M. R., and Liebling, D. J. (2012). SearchBuddies: Bringing Search Engines into the Conversation. In *Proceedings of the 6th International AAAI Conference On Weblogs and Social Media*, pages 138–145, Dublin, Ireland. 24

Hertzum, M. (2014). Expertise seeking: A review. *Information Processing and Management*, 50(5):775 – 795. 22, 36

Hertzum, M. and Pejtersen, A. M. (2000). The information-seeking practices of engineers: searching for documents as well as for people. *Information Processing and Management*, 36(5):761–778. 4, 22

Hiemstra, D. (2001). *Using language models for information retrieval*. PhD thesis, University of Twente. 21

Hofmann, K., Balog, K., Bogers, T., and De Rijke, M. (2010). Contextual factors for finding similar experts. *Journal of the American Society for Information Science and Technology*, 61(5):994–1014. 36, 100

Jelinek, F. (1997). *Statistical methods for speech recognition*. MIT press. 20

Jelinek, F. and Mercer, R. L. (1980). Interpolated estimation of markov source parameters from sparse data. In *Pattern recognition in practice*, pages 381–402. NorthHolland publishing company. 21

Jiang, J., Lu, W., and Zhao, H. (2008). CSIR at TREC 2008 Expert Search Task: Modeling Expert Evidence in Expert Search. In *Proceedings of the 17th Text REtrieval Conference (TREC-17)*, volume 500-277 of *NIST Special Publication*, Gaithersburg, Maryland. 37

Jurafsky, D. and Martin, J. H. (2009). *SPEECH and LANGUAGE PROCESSING*. Prentice Hall, 2ed edition edition. 88

Justeson, J. S. and Katz, S. M. (1995). Technical terminology: some linguistic properties and an algorithm for identification in text. *Natural language engineering*, 1(01):9–27. xiixii, 119, 120

Kantor, P. B. and Voorhees, E. M. (2000). The trec-5 confusion track: Comparing retrieval methods for scanned text. *Information Retrieval*, 2(2-3):165–176. 56

Kaptein, R., Serdyukov, P., de Vries, A., and Kamps, J. (2010). Entity ranking using wikipedia as a pivot. In *CIKM '10: : Proceedings of the 19nd ACM international conference on Conference on information and knowledge management*, pages 69–78, New York, USA. ACM. 113

Karimzadehgan, M., Zhai, C., and Belford, G. (2008). Multi-aspect expertise matching for review assignment. In *CIKM '08: Proceedings of the 17th ACM conference on Information and knowledge management*, pages 1113–1122, New York, USA. ACM. 22

Kaszkiel, M. and Zobel, J. (2001). Effective ranking with arbitrary passages. *Journal of the American Society for Information Science and Technology*, 52(4):344–364. 129

Kowalski, G. J. and Maybury, M. T. (2002). *Information Storage and Retrieval Systems: Theory and Implementation.* Springer US. 14

Krulwich, B. and Burkey, C. (1995). ContactFinder: Extracting indications of expertise and answering questions with referrals. In *Working Notes of the 1995 Fall Symposium on Intelligent Knowledge Navigation and Retrieval*, pages 85–91. 26

Krulwich, B. and Burkey, C. (1996). The contactfinder agent: answering bulletin board questions with referrals. In *Proceedings of the thirteenth National Conference on Artificial Intelligence*, pages 10–15, Portland, Oregon. 23, 26

Kumar, R. and Tomkins, A. (2010). A Characterization of Online Browsing Behavior. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pages 561–570. ACM. 22

Lavrenko, V. and Croft, W. (2001). Relevance based language models. In *SIGIR '01: Proceedings of the 24th International ACM SIGIR conference on research and development in Information Retrieval*, pages 120–127, New Orleans, USA. ACM. 28

Lelei, D. E. K. (2013). Supporting lifelong learning: Recommending personalized sources of assistance to graduate students. In Lane, H., Yacef, K., Mostow, J., and Pavlik, P., editors, *Artificial Intelligence in Education*, volume 7926 of *Lecture Notes in Computer Science*, pages 912–915. Springer Berlin Heidelberg. 108

Lu, C., Bing, L., and Lam, W. (2013). Structured positional entity language model for enterprise entity retrieval. In *CIKM '13: Proceedings of the 22nd ACM international conference on Conference on information and knowledge management*, pages 129–138, New York, USA. ACM. 63

Luhn, H. P. (1957). A statistical approach to mechanized encoding and searching of literary information. *IBM Journal of Research and Development*, 1(4):309–317. 15

Luhn, H. P. (1958). The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2(2):159–165. 15

Lv, Y. and Zhai, C. (2009). Positional language models for information retrieval. In *SIGIR '09: Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 299–306, Boston, USA. ACM. 62

Macdonald, C., Hannah, D., and Ounis, I. (2008). High quality expertise evidence for expert search. In *Advances in Information Retrieval, Proceedings of the 30th European Conference on IR Research (ECIR-2008)*, volume 4956 of *Lecture*

Notes in Computer Science, pages 283–295, Glasgow, UK. Springer. 33, 34, 59, 60

Macdonald, C. and Ounis, I. (2006). Voting for candidates: adapting data fusion techniques for an expert search task. In *CIKM '06: Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 387–396, New York, USA. ACM. ixix, 23, 34

Macdonald, C. and Ounis, I. (2007). Expertise drift and query expansion in expert search. In *CIKM '07: Proceedings of the 16th ACM conference on Conference on information and knowledge management*, pages 341–350, New York, USA. ACM. 118

Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Introduction to information retrieval*, volume 1. Cambridge University Press, Cambridge. 17, 38, 55, 91

Manning, C. D. and Schütze, H. (1999). *Foundations of statistical natural language processing*. MIT press. 119

Maron, M., Kuhns, J., and Ray, L. (1959). Probabilistic indexing. a statistical technique for document identification and retrieval. Technical report, DTIC Document. 15

Marz, N. and Warren, J. (2015). *Big Data: Principles and Best Practices of Scalable Realtime Data Systems*. Manning Publications Co., Greenwich, CT, USA, 1st edition edition. 4

Mattox, D., Maybury, M. T., and Morey, D. (1999). Enterprise Expert and Knowledge Discovery. In *Proceedings of the 8th International Conference on Human-Computer Interaction*, pages 303–307, Hillsdale, USA. L. Erlbaum Associates Inc. 23

McAlpine, L. and Norton, J. (2006). Reframing our approach to doctoral programs: an integrative framework for action and research. *Higher Education Research and Development*, 25(1):3–17. 108

McCampbell, A. S., Clare, L. M., and Gitters, S. H. (1999). Knowledge management: the new challenge for the 21st century. *Journal of Knowledge Management*, 3(3):172–179. 23

McDonald, D. and Ackerman, M. (1998). Just talk to me: a field study of expertise location. In *CSCW '98: Proceedings of the 1998 ACM Conference on Computer Supported Cooperative Work*, pages 315–324. ACM. 35

McDonald, D. W. (2001). Evaluating expertise recommendations. In *GROUP '01: Proceedings of the 2001 International ACM Conference on Supporting Group Work*, pages 214–223. ACM. 25

McDonald, D. W. and Ackerman, M. S. (2000). Expertise Recommender: A Flexible Recommendation System and Architecture. In *CSCW '00: Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work*, pages 231–240. ACM. 24

Miao, J., Huang, J. X., and Ye, Z. (2012). Proximity-based rocchio's model for pseudo relevance. In *SIGIR '12: Proceedings of the 35th International ACM SIGIR conference on research and development in Information Retrieval*, pages 535–544, Portland, Oregon. ACM. 78

Mockus, A. and Herbsleb, J. (2002). Expertise browser: a quantitative approach to identifying expertise. In *Proceedings of the 24th International Conference on Software Engineering*, pages 503–512. ACM. 26

Peters, C., Braschler, M., and Clough, P. (2012). *Multilingual Information Retrieval*. Springer Berlin Heidelberg. 130, 137

Petkova, D. and Croft, W. B. (2007). Proximity-based document representation for named entity retrieval. In *CIKM '07: Proceedings of the 16th ACM conference on Conference on information and knowledge management*, pages 731–740, New York, USA. ACM. 27, 28, 37, 59, 60, 63

Petkova, D. and Croft, W. B. (2008). Hierarchical language models for expert finding in enterprise corpora. *International Journal on Artificial Intelligence Tools*, 17(01):5–18. 27, 28, 59

Pikrakis, A., Bitsikas, T., Sfakianakis, S., Hatzopoulos, M., DeRoure, D., Hall, W., Reich, S., Hill, G., and Stairmand, M. (1998). MEMOIR-Software Agents for Finding Similar Users by Trails. In *PAAM '98: The third International Conference and Exhibition on the Practical Application of Intelligent Agents and Multi-Agents*, pages 453–466, London, UK. 26

Ponte, J. M. and Croft, W. B. (1998). A language modeling approach to information retrieval. In *SIGIR '98: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 275–281, Melbourne, Australia. ACM. 20

Porter, M. F. (1980). An algorithm for suffix stripping. *Program*, 14(3):130–137. 16, 88

Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77:257–286. 20

Robertson, S. E., Walker, S., Hancock-Beaulieu, M., and Gatford, M. (1992a). Okapi at TREC-2. In *Proceedings of the 2ed Text REtrieval Conference (TREC-2)*, volume 500-215 of *NIST Special Publication*, pages 21–34, Gaithersburg, Maryland. 18

Robertson, S. E., Walker, S., Hancock-Beaulieu, M., Gull, A., and Lau, M. (1992b). Okapi at TREC. In *Proceedings of the 1st Text REtrieval Conference (TREC-1)*, volume 500-207 of *NIST Special Publication*, pages 21–30, Gaithersburg, Maryland. 18

Robertson, S. E., Walker, S., Jones, S., Hancock-Beaulieu, M. M., Gatford, M., et al. (1994). Okapi at TREC-3. In *Proceedings of the 3ed Text REtrieval Conference (TREC-3)*, volume 500-225 of *NIST Special Publication*, pages 109–128, Gaithersburg, Maryland. xiiixiii, 16, 18, 19

Rocchio, J. J. (1965). Relevance feedback in information retrieval. Technical Report ISR-9, The Computation Laboratory of Harvard University, Cambridge. Massachusetts. 16

Rocchio, J. J. (1971). *Relevance feedback in information retrieval.* Prentice-Hall, Englewood Cliffs NJ. 118

Rode, H., Serdyukov, P., and Hiemstra, D. (2008). Combining document-and paragraph-based entity ranking. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 851–852, Singapore, Singapore. ACM. 28

Ru, Z., Chen, Y., Xu, W., and Guo, J. (2005). TREC 2005 Enterprise Track Experiments at BUPT. In *Proceedings of the 14th Text REtrieval Conference (TREC-14)*, volume 500-266 of *NIST Special Publication*, Gaithersburg, Maryland. 74

Salton, G. (1968). *Automatic Information Organization and Retrieval.* McGraw-Hill, New York. 16

Salton, G. and McGill, M. J. (1986). *Introduction to Modern Information Retrieval.* McGraw-Hill, Inc., New York, NY, USA. 17

Salton, G., Wong, A., and Yang, C.-S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620. 16

Salton, G. and Yang, C.-S. (1973). On the specification of term values in automatic indexing. *Journal of Documentation*, 29(4):351–372. 16

Sanderson, M. (2010). Test collection based evaluation of information retrieval systems. *Foundations and Trends in Information Retrieval*, 4(4):247–375. 38, 52

Sanderson, M. and Croft, W. B. (2012). The history of information retrieval research. *Proceedings of the IEEE*, 100(Special Centennial Issue):1444–1451. 16, 17

Sanderson, M. and Zobel, J. (2005). Information retrieval system evaluation: effort, sensitivity, and reliability. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 162–169, Salvador, Brazil. ACM. 55

SanJuan, E., Flavier, N., Ibekwe-SanJuan, F., and Bellot, P. (2008). Universities of Avignon Lyon III at TREC 2008: Enterprise track. In *Proceedings of the 17th Text REtrieval Conference (TREC-17)*, volume 500-277 of *NIST Special Publication*, Gaithersburg, Maryland. 119

Santos, R. L. T., Macdonald, C., and Ounis, I. (2010). Voting for related entities. In *RIAO '10: Proceedings of the 9th RIAO Conference Adaptivity, Personalization and Fusion of Heterogeneous Information*, pages 1–8. 113

Serdyukov, P., Rode, H., and Hiemstra, D. (2008). Modeling multi-step relevance propagation for expert finding. In *CIKM '08: Proceedings of the 17th ACM conference on Information and knowledge management*, pages 1133–1142, New York, USA. ACM. 23, 28

Shami, N., Ehrlich, K., and Millen, D. (2008). Pick Me!: Link Selection in Expertise Search Results. In *CHI '08: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1089–1092. ACM. 35

Shen, H., Wang, L., Bi, W., Liu, Y., Cheng, X., and Cheng, X. (2008). Research on enterprise track of TREC 2008. In *Proceedings of the 17th Text REtrieval Conference (TREC-17)*, volume 500-277 of *NIST Special Publication*, Gaithersburg, Maryland. 74, 103

Singhal, A. (2001). Modern information retrieval: A brief overview. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 24(4):35–43. 15, 16, 53

Smirnova, E. and Balog, K. (2011). A user-oriented model for expert finding. In *Proceedings of 33rd European Conference on Information Retrieval (ECIR-2011)*, volume 5478 of *Lecture Notes in Computer Science*, pages 580–592. Springer, Dublin, Ireland. 36

Soboroff, I., de Vries, A. P., and Craswell, N. (2006). Overview of the TREC 2006 Enterprise Track. In *Proceedings of the 15th Text REtrieval Conference (TREC-15)*, volume 500-272 of *NIST Special Publication*, Gaithersburg, Maryland. xiixii, 41, 56, 95, 97, 164

Song, F. and Croft, W. B. (1999). A general language model for information retrieval. In *CIKM '99: Proceedings of the 8th international conference on Information and knowledge management*, pages 316–321, New York, USA. ACM. 20

Spärck Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28(1):11–21. 16

Spärck Jones, K., Robertson, S., Hiemstra, D., and Zaragoza, H. (2003). Language modeling and relevance. *The Springer International Series on Information Retrieval*, 13:57–71. 20

Streeter, L. A. and Lochbaum, K. E. (1988). An expert/expert-locating system based on automatic representation of semantic structure. In *Proceedings of the Fourth Conference on Artificial Intelligence Applications*, pages 345–350. IEEE. 23

Switzer, P. (1964). Vector images in document retrieval. In *Proceedings of the Statistical Association Methods for Mechanized Documentation*, pages 163–171, Washington, USA. 16

Tague-Sutcliffe, J. and Blustein, J. (1994). A statistical analysis of the TREC-3 data. In *Proceedings of the 3ed Text REtrieval Conference (TREC-3)*, volume 500-226 of *NIST Special Publication*, Gaithersburg, Maryland. 54

Tang, J., Wu, S., Sun, J., and Su, H. (2012). Cross-domain Collaboration Recommendation. In *KDD '12: Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1285–1293. ACM. 22

Tellex, S., Katz, B., Lin, J., Fernandes, A., and Marton, G. (2003). Quantitative evaluation of passage retrieval algorithms for question answering. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 41–47, Toronto, Canada. ACM. 137

van Rijsbergen, C. J. (1979). *Information Retrieval.* Butterworth-Heinemann, Newton, MA, USA, 2nd edition. 13, 88

Vechtomova, O. and Robertson, S. E. (2012). A domain-independent approach to finding related entities. *Information Processing and Management*, 48(4):654–670. 61, 114

Voorhees, E. and Harman, D. (1998). Overview of the seventh text retrieval conference (TREC-7). In *Proceedings of the 7th Text REtrieval Conference (TREC-7)*, volume 500-242 of *NIST Special Publication*, Gaithersburg, Maryland. 54

Voorhees, E. M. (1999). The TREC-8 Question Answering Track Report. In *Proceedings of the 8th Text REtrieval Conference (TREC-8)*, volume 500-246 of *NIST Special Publication*, pages 77–82, Gaithersburg, Maryland. 56

Voorhees, E. M. (2002). CLEF 2001: Second Workshop of the Cross-Language Evaluation Forum. In Peters, C., Braschler, M., Gonzalo, J., and Kluck, M., editors, *Evaluation of Cross-Language Information Retrieval Systems*, volume 2406 of *Lecture Notes in Computer Science*, pages 355–370. Springer. 38, 39, 130

Voorhees, E. M. and Harman, D. K. (2005). *TREC: Experiment and evaluation in information retrieval*, volume 1. MIT press Cambridge. 16, 38, 54

Wang, Z., Tang, C., Sun, X., Ouyang, H., Lan, R., Xu, W., Chen, G., and Guo, J. (2010). Pris at TREC 2010: Related Entity Finding task of Entity Track. In *Proceedings of the 19th Text REtrieval Conference (TREC-19)*, volume 500-294 of *NIST Special Publication*, Gaithersburg, Maryland. 113

Weerkamp, W., Balog, K., and Meij, E. (2009). A Generative Language Modeling Approach for Ranking Entities. In *INEX '08: Proceedings of the 7th International Workshop of the Initiative for the Evaluation of XML Retrieval*, volume

5631, pages 292–299, Dagstuhl Castle, Germany. Springer Berlin Heidelberg. 113

Williams, R. V. (2002). The use of punched cards in us libraries and documentation centers, 1936-1965. *IEEE Annals of the History of Computing*, 24(2):16–33. 15

Woudstra, L. and van den Hooff, B. (2008). Inside the source selection process: Selection criteria for human information sources. *Information Processing and Management*, 44(3):1267 – 1278. 35, 43

Wu, M., Scholer, F., and Garcia, S. (2008). Rmit university at TREC 2008: Enterprise track. In *Proceedings of the 17th Text REtrieval Conference (TREC-17)*, volume 500-277 of *NIST Special Publication*, Gaithersburg, Maryland. 118

Yang, L., Qiu, M., Gottipati, S., Zhu, F., Jiang, J., Sun, H., and Chen, Z. (2013). CQArank: jointly model topics and expertise in community question answering. In *CIKM '13: Proceedings of the 22nd ACM international conference on Conference on information and knowledge management*, pages 99–108, New York, USA. ACM. 24

Yao, J., Xu, J., and Niu, J. (2008). Using role determination and expert mining in the enterprise environment. In *Proceedings of the 17th Text REtrieval Conference (TREC-17)*, volume 500-277 of *NIST Special Publication*, Gaithersburg, Maryland. 27

Yilmaz, E. and Aslam, J. A. (2006). Estimating average precision with incomplete and imperfect judgments. In *CIKM '06: Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 102–111, New York, USA. ACM. 55

Yimam-Seid, D. and Kobsa, A. (2009). Expert-finding systems for organizations: Problem and domain analysis and the DEMOIR approach. *Journal of Organizational Computing and Electronic Commerce*, 13(1):1–24. 23

Zhai, C. (2008). Statistical language models for information retrieval a critical review. *Foundations and Trends in Information Retrieval*, 2(3):137–213. 19, 20

Zhao, J., Huang, J. X., and He, B. (2011). CRTER: using cross terms to enhance probabilistic information retrieval. In *SIGIR '11: Proceedings of the 34th International ACM SIGIR conference on research and development in Information Retrieval*, pages 155–164, Beijing, China. ACM. 61, 64

Zhu, J., Huang, X., Song, D., and Rüger, S. (2010). Integrating multiple document features in language models for expert finding. *Knowledge and Information Systems*, 23(1):29–54. 28

Zhu, J., Song, D., and Rüger, S. (2008). The Open University at TREC 2007 Enterprise Track. In *Proceedings of the 16th Text REtrieval Conference (TREC-16)*, volume 500-274 of *NIST Special Publication*, Gaithersburg, Maryland. 103

Zhu, J., Song, D., and Rüger, S. (2009). Integrating multiple windows and document features for expert finding. *Journal of the American Society for Information Science and Technology*, 60(4):694–715. 61

Zhu, J., Song, D., Rüger, S., Eisenstadt, M., and Motta, E. (2006). The Open University at TREC 2006 Enterprise Track Expert Search Task. In *Proceedings of the 15th Text REtrieval Conference (TREC-15)*, volume 500-272 of *NIST Special Publication*, Gaithersburg, Maryland. 28, 57, 60, 98

Zikopoulos, P., deRoos, D., Bienko, C., Buglio, R., and Andrews, M. (2015). *Big Data Beyond the Hype*. McGraw-Hill Education, New York, USA. 4

A
TREC Results

Table A.1: TREC 2005 Expert-Finding Results [Craswell et al., 2005]

Run	MAP	r-prec	bpref	P@5	P@10	P@20	P@30	P@100	P@1000	RR1
THUENT0505	**0.2749**	**0.3330**	0.4880	**0.4880**	**0.4520**	**0.3390**	**0.2800**	0.1142	0.0114	**0.7268**
MSRA054	0.2688	0.3192	**0.5685**	0.4080	0.3700	0.3190	0.2753	0.1306	**0.0131**	0.6244
MSRA055	0.2600	0.3089	0.5655	0.3920	0.3580	0.3150	0.2733	**0.1308**	**0.0131**	0.5832
CNDS04LC	0.2174	0.2631	0.4299	0.4120	0.3460	0.2820	0.2240	0.0942	0.0094	0.6068
uogES05CbiH	0.1851	0.2397	0.4662	0.3800	0.3160	0.2600	0.2133	0.1130	0.0113	0.5519
PRISEX3	0.1833	0.2269	0.4182	0.3440	0.3080	0.2530	0.2087	0.1026	0.0103	0.5614
uams05run1	0.1277	0.1811	0.3925	0.2720	0.2220	0.2000	0.1753	0.0944	0.0094	0.4380
DREXEXP1	0.1262	0.1743	0.3409	0.3120	0.2500	0.1760	0.1467	0.0720	0.0072	0.4635
LLEXemails	0.0960	0.1357	0.2985	0.2000	0.1860	0.1530	0.1213	0.0628	0.0063	0.4054
qmirex4	0.0959	0.1511	0.2730	0.2360	0.1880	0.1390	0.1233	0.0534	0.0053	0.4189

Table A.2: TREC 2006 Expert-Finding Results. The table is sorted according to MAP for the best run in each group as reported in [Soboroff et al., 2006]

Run	MAP	R-prec	bpref	P@5	P@10	P@20	MRR
kmiZhu1	**0.4421**	**0.4835**	**0.4986**	**0.6612**	**0.5633**	**0.4459**	**0.8369**
SJTU04	0.3943	0.4304	0.4581	0.5714	0.5204	0.4143	0.8132
SRCBEX5	0.3602	0.4092	0.4299	0.5551	0.4735	0.3969	0.7350
IBM06MA	0.3346	0.3829	0.4135	0.5878	0.4878	0.3602	0.7339
PRISEXB	0.3345	0.4203	0.4228	0.5429	0.4571	0.3847	0.6695
UvAprofiling	0.3016	0.3637	0.3743	0.4980	0.4265	0.3582	0.7177
FDUSF	0.2796	0.3148	0.3356	0.4653	0.4041	0.3204	0.6767
UMaTiDm	0.2740	0.3205	0.3350	0.4980	0.4102	0.3204	0.6344
THUPDDFBS	0.2573	0.3035	0.3155	0.4082	0.3673	0.3020	0.6117
DUTEX2	0.2290	0.2918	0.3028	0.4898	0.3898	0.3031	0.6703
qutbaseline	0.2110	0.2561	0.2527	0.4082	0.3531	0.2694	0.6115
ex3512	0.2031	0.2466	0.2724	0.3959	0.3286	0.2786	0.6481
UIUCe2	0.1650	0.2271	0.2582	0.3143	0.2898	0.2347	0.5874
ICTCSXRUN01	0.1648	0.2338	0.2497	0.2857	0.2347	0.2143	0.4245
UMDemailTLNR	0.1410	0.2015	0.1997	0.3388	0.2980	0.2357	0.5561
uwXSHUBS	0.1389	0.2028	0.1938	0.3551	0.2878	0.2449	0.5185
uogX06csnQE	0.1387	0.2046	0.2180	0.3061	0.2551	0.2071	0.5430
PITTPHFREQ	0.1117	0.1843	0.1744	0.3143	0.2857	0.2031	0.5085
allbasic	0.0996	0.1479	0.1409	0.3020	0.2429	0.1786	0.5233
sophiarun1	0.0934	0.1415	0.1322	0.3184	0.2449	0.1582	0.4646
SPlog	0.0781	0.1179	0.1470	0.2000	0.1694	0.1347	0.4265
l3s2	0.0714	0.0827	0.0820	0.3429	0.1755	0.0878	0.5840
body	0.0484	0.0809	0.1004	0.1224	0.1122	0.0918	0.2606

Table A.3: TREC 2007 Expert-Finding Results as reported in [Bailey et al., 2007a]

Group	Run	MAP	P@5	P@20
Tsinghua	THUIRMPDD4	0.4632	0.2280	0.0910
SJTU	SJTUEntES03	0.4427	0.2360	0.0910
OU	ouExTitle	0.4337	0.2520	0.0950
CAS	ExpertRun02	0.3689	0.2040	0.0790
CSIRO	CSIROesQnarr	0.3655	0.2240	0.0770
Wuhan	WHU10	0.3399	0.1960	0.0710
Glasgow	uogEXFeMNZcP	0.3138	0.2200	0.0800
UvA	uams07exbl	0.3090	0.2080	0.0790
DUT	DUTEXP1	0.2630	0.1400	0.0580
Fudan	FDUn7e3	0.1788	0.1440	0.0610
Beijing	PRISRR	0.1571	0.0920	0.0440
Twente	qorwnewlinks	0.1481	0.1080	0.0540
Peking	zslrun	0.0944	0.0600	0.0220
Hyberbad	AUTORUN	0.0939	0.0560	0.0330
UALR	UALR07Exp1	0.0200	0.0160	0.0130

Table A.4: First 20 results of TREC 2008 Expert-Finding task [Balog et al., 2008]

Run	Group	Type	Fields	MAP	MRR
UvA08ESweb	UAmsterdam	auto	q	0.4490	0.8721
ICTI3Sexp01	CAS	auto	q	0.4214	0.7241
uogTrEXfeNPC	UGlasgow	auto	q	0.4126	0.7611
FDURoleRes	Fudan	auto	qn	0.4114	0.7516
THUPDDlchrS	Tsinghua	auto	q	0.3846	0.7419
WHU08NOPHR	Wuhan	auto	q	0.3826	0.6770
utqurl	UTwente	auto	q	0.3728	0.7647
UCLex04	UC-London	auto	q	0.3476	0.6759
DERIrun3	NUI-Galway	auto	q	0.2619	0.6212
LiaIcExp08	UAvingon	manual	qn	0.2513	0.8545
pristask204	BUPT	manual	qn	0.0977	0.2343