From People to Entities:

Typed Search in the Enterprise and the Web

Der Fakultät für Elektrotechnik und Informatik der
Gottfried Wilhelm Leibniz Universität Hannover
zur Erlangung des akademischen Grades
Doktor der Naturwissenschaften (**Dr. rer. nat.**) genehmigte Dissertation

von MSc **Gianluca Demartini**

geboren am 06.12.1981, in Gorizia, Italien.

2011

Referent: Prof. Dr. Wolfgang Nejdl
Ko-Referent: Prof. Dr. Heribert Vollmer
Ko-Referent: Prof. Dr. Arjen P. de Vries
Tag der Promotion: 06. April 2011

ABSTRACT

The exponential growth of digital information available in Enterprises and on the Web creates the need for search tools that can respond to the most sophisticated informational needs. Retrieving relevant documents is not enough anymore and finding entities rather than just textual resources provides great support to the final user both on the Web and in Enterprises. Many user tasks would be simplified if Search Engines would support typed search, and return entities instead of just Web pages. For example, an executive who tries to solve a problem needs to find people in the company who are knowledgeable about a certain topic. Aggregation of information spread over different documents is a key aspect in this process.

Finding experts is a problem mostly considered in the Enterprise setting where teams for new projects need to be built and problems need to be solved by the right persons. In the first part of the thesis, we propose a model for expert finding based on the well consolidated vector space model for Information Retrieval and investigate its effectiveness.

We can define Entity Retrieval by generalizing the expert finding problem to any entity. In Entity Retrieval the goal is to rank entities according to their relevance to a query (e.g., "Countries where I can pay in Euro"); the set of entities to be ranked is assumed to be loosely defined by a generic category, given in the query itself (e.g., countries), or by some example entities (e.g., Italy, Germany, France). In the second part of the thesis, we investigate different methods based on Semantic Web and Natural Language Processing techniques for solving these tasks both in Wikipedia and, generally, on the Web. Evaluation is a critical aspect of Information Retrieval. We contributed to the field of Information Retrieval evaluation by organizing an evaluation initiative for Entity Retrieval.

Opinions and other relevant information about entities can be provided by different sources in different contexts. News articles report about events where entities are involved. In such setting the temporal dimension is critical as news stories develop over time and new entities appear in the story and others are not relevant anymore. In the third part of this thesis, we study the problem of Entity Retrieval for news applications and the importance of the news trail history (i.e., past related articles) to determine the relevant entities in current articles. We also study opinion evolution about entities. In the last years, the blogosphere has become a vital part of the Web, covering a variety of different points of view and opinions on political and event-related topics such as immigration, election campaigns, or economic developments. We propose a method for automatically extracting public opinion about specific entities from the blogosphere.

In summary, we develop methods to find entities that satisfy the user's need aggregating knowledge from different sources and we study how entity relevance and opinions evolve over time.

Keywords: *Information Retrieval, Expert Search, Entity Retrieval, Wikipedia*

ZUSAMMENFASSUNG

Das exponentielle Wachstum der Menge digitaler Informationen in Unternehmen und im Web macht die Entwicklung von innovativen Suchtechniken zur Deckung von immer anspruchsvolleren Informationsbedürfnissen notwendig. Konventionelle, text-basierte Dokumentsuche ist oft unzureichend; in vielen Fällen ist eine gezielte Suche nach Entities, sowohl im Web als auch in Unternehmen, erforderlich. Die Untersttzung von typisierter Suche durch Suchmaschinen kann dem Benutzer viele Aufgaben erleichtern. So kann eine Führungskraft, beispielsweise, die ein spezifisches Problem lösen mchte, Personen im Unternehmen finden, die mit der entsprechenden Thematik vertraut sind. Der Aggregierung von ber mehrere Dokumente verteilten Informationen kommt eine Schlüsselrolle in diesem Prozess zu.

Das Finden von Experten spielt hautschlich im Kontext von Unternehmen eine Rolle, wo Teams für neue Projekte zusammengestellt und geeignete Personen für bestimmte Problemstellungen identifiziert werden sollen. Im ersten Teil dieser Arbeit, stellen wir ein Modell für die Suche nach Experten vor, das auf einem fundierten Vektorraummodell für das Information Retrieval basiert, und wir untersuchen dessen Effektivität.

Entity-Suche kann als eine Generalisierung der Suche nach Experten auf beliebige Entities definiert werden. Ziel der Entity-Suche ist es, als Antwort auf eine Query (z.B. “Länder, in denen ich mit dem Euro bezahlen kann”) eine gerankte Liste von Entities zu erzeugen; es wird angenommen, dass die Menge der zu rankenden Entities grob durch eine generische Kategorie definiert ist, die durch die Query selbst (z.B. “Länder”) oder durch eine Menge von Beispiel-Entities (z.B. Italien, Deutschland, Frankreich) vorgegeben sein kann. Im zweiten Teil dieser Arbeit untersuchen wir verschiedene Methoden basierend auf Semantic Web Techniken und Natural Language Processing, um diese Aufgaben sowohl in Wikipedia als auch, allgemeiner, im Web zu lösen. Ein entscheidender Aspekt im Information Retrieval ist die Evaluation. Dazu haben wir im Rahmen einer von uns organisierten Evaluationsinitiative im Bereich der Entity-Suche beigetragen.

Meinungen und andere relevante Informationen ber Entities können aus unterschiedlichen Quellen und Kontexten stammen. Nachrichtenartikel, etwa, berichten ber Ereignisse in die Entities involviert sind. In diesem Zusammenhang ist die zeitliche Dimension von entscheidender Bedeutung, da sich Nachrichten im Laufe der Zeit entwickeln, neue Entities auftreten, und andere an Relevanz verlieren. Im dritten Teil dieser Arbeit untersuchen wir das Problem der Entity Suche für Nachrichtenanwendungen - insbesondere die Verwendung der Historie der Nachrichtenmeldungen (d.h. verwandte Artikel aus der Vergangenheit) zur Identifikation von Entities in aktuellen Artikeln. Weiterhin untersuchen wir die Evolution von Meinungen ber Entities. In den letzten Jahren wurde die Blogosphäre zu einem wichtigen Bestandteil des Webs, der eine Vielzahl von verschiedenen Blickwinkeln und Meinungen zu politischen und Ereignis-orientierten Themen wie Immigration, Wahlkampagnen, oder wirtschaftliche Entwicklungen bietet. Wir stellen eine Methode für die automatische Extrahierung von Politischen Meinungen über spezifische Entities aus der Blogosphäre vor.

Zusammengefasst entwickeln wir Methoden zum Finden von Entities, um somit den Informationsbedarf von Nutzern durch Aggregierung von Wissen aus verschieden Quellen zu decken, und wir untersuchen die zeitliche Entwicklung der Relevanz von Entities und der Meinungen zu Entities.

Schlagwörter: *Information Retrieval, Expert Search, Entity Retrieval, Wikipedia*

FOREWORD

The algorithms presented in this thesis have been published at various conferences or journals, as follows.

In Chapter 3 we describe contributions included in:

- *A Vector Space Model for Ranking Entities and Its Application to Expert Search*. **Gianluca Demartini**, Julien Gaugaz, and Wolfgang Nejdl. In: 31st European Conference on Information Retrieval (ECIR 2009), Toulouse, France, April 2009. [67]

Chapter 4 presenting the approaches to rank entities in Wikipedia is built upon the work published in:

- *Why Finding Entities in Wikipedia is Difficult, Sometimes*. **Gianluca Demartini**, Claudiu S. Firan, Tereza Iofciu, Ralf Krestel, and Wolfgang Nejdl. In: "Information Retrieval", Special Issue on Focused Retrieval and Result Aggregation, 2010. [65]

- *Overview of the INEX 2008 Entity Ranking Track*. **Gianluca Demartini**, Arjen P. de Vries, Tereza Iofciu, and Jianhan Zhu. In: 7th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2008 Dagstuhl Castle, Germany, December 2008. [60]

- *Overview of the INEX 2009 Entity Ranking Track*. **Gianluca Demartini**, Tereza Iofciu, and Arjen P. de Vries. In: 8th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2009 Brisbane, Australia, December 2009. [68]

Finally, in Chapter 5 we structure the presentation around the following papers:

- *Entity Summarization of News Articles*. **Gianluca Demartini**, Malik Muhammad Saad Missen, Roi Blanco, and Hugo Zaragoza. In: 33rd Annual ACM SIGIR Conference (SIGIR 2010), Geneva, Switzerland, July 2010. [69]

- *TAER: Time-Aware Entity Retrieval - Exploiting the Past to find Relevant Entities in News Articles*. **Gianluca Demartini**, Malik Muhammad Saad Missen, Roi Blanco, and Hugo Zaragoza. In: The 19th ACM International Conference on Information and Knowledge Management (CIKM 2010), Toronto, Canada, October 2010. [70]

- *Analyzing Political Trends in the Blogosphere*. **Gianluca Demartini**, Stefan Siersdorfer, Sergiu Chelaru, and Wolfgang Nejdl. In: Fifth International AAAI Conference on Weblogs and Social Media (ICWSM 2011), Barcelona, Spain, July 2011. [73]

During the Ph.D. work, I have also published a number of additional papers about Entity Retrieval, and Desktop Search. These papers are not touched in this thesis due to space limitation, but the complete list of publications follows:

- *Visual Interfaces for Stimulating Exploratory Search.* Ralf Krestel, **Gianluca Demartini** and Eelco Herder. In: ACM/IEEE Joint Conference on Digital Libraries (JCDL 2011), Ottawa, Canada, June 2011. [105]
- *ReFER: effective Relevance Feedback for Entity Ranking*. Tereza Iofciu, **Gianluca Demartini**, Nick Craswell, and Arjen P. de Vries. In: 33rd European Conference on Information Retrieval (ECIR 2011), Dublin, Ireland, April 2011. [97]
- *ARES: A Retrieval Engine based on Sentiments - Sentiment-based Search Result Annotation and Diversification*. **Gianluca Demartini**. In: 33rd European Conference on Information Retrieval (ECIR 2011 - Demo), Dublin, Ireland, April 2011. [58]
- *The Missing Links: Discovering Hidden Same-as Links among a Billion of Triples*. George Papadakis, **Gianluca Demartini**, Philipp Kärger and Peter Fankhauser. In: The 12th International Conference on Information Integration and Web-based Applications & Services (iiWAS2010), Paris, France, November 2010. [134]
- *Ranking Entities Using Web Search Query Logs*. Bodo Billerbeck, **Gianluca Demartini**, Claudiu S. Firan, Tereza Iofciu, and Ralf Krestel. In: 14th European Conference on Research and Advanced Technology for Digital Libraries (ECDL 2010), Glasgow, Scotland, September 2010. [30]
- *Exploiting Click-Through Data for Entity Retrieval*. Bodo Billerbeck, **Gianluca Demartini**, Claudiu S. Firan, Tereza Iofciu, and Ralf Krestel. In: 33rd Annual ACM SIGIR Conference (SIGIR 2010 poster session), Geneva, Switzerland, July 2010. [29]
- *Report on INEX 2009*. T. Beckers, P. Bellot, **G. Demartini**, L. Denoyer, C.M. De Vries, A. Doucet, K.N. Fachry, N. Fuhr, P. Gallinari, S. Geva, W.-C. Huang, T. Iofciu, J. Kamps, G. Kazai, M. Koolen, S. Kutty, M. Landoni, M. Lehtonen, V. Moriceau, R. Nayak, R. Nordlie, N. Pharo, E. San Juan, R. Schenkel, X. Tannier, M. Theobald, J.A. Thom, A. Trotman and A.P. de Vries. In: SIGIR Forum, June 2010, Volume 44 Number 1, pp 38-56. [26]

- *Dear Search Engine: What's your opinion about...? - Sentiment Analysis for Semantic Enrichment of Web Search Results*. **Gianluca Demartini** and Stefan Siersdorfer. In: Semantic Search 2010 Workshop located at the 19th Int. World Wide Web Conference WWW2010, Raleigh, NC, USA, April 2010. [72]

- *Leveraging Personal Metadata for Desktop Search – The Beagle++ System*. Enrico Minack, Raluca Paiu, Stefania Costache, **Gianluca Demartini**, Julien Gaugaz, Ekaterini Ioannou, Paul A. Chirita, and Wolfgang Nejdl. In: Journal of Web Semantics, Volume 8, Issue 1, Pages 37-54, March 2010. [125]

- *Current Approaches to Search Result Diversification*. Enrico Minack, **Gianluca Demartini**, and Wolfgang Nejdl. In: "1st International Workshop on Living Web: Making Web Diversity a true asset" at ISWC 2009, Washington DC, USA, October 2009. [124]

- *An Architecture for Finding Entities on the Web*. **Gianluca Demartini**, Claudiu S. Firan, Mihai Georgescu, Tereza Iofciu, Ralf Krestel, and Wolfgang Nejdl. In: 7th Latin American Web Congress (LA-WEB 2009), Mrida, Yucatan, Mxico, November 2009. 3rd Best Paper Award. [62]

- *Time based Tag Recommendation using Direct and Extended Users Sets*. Tereza Iofciu and **Gianluca Demartini**. In: ECML PKDD Discovery Challenge 2009, Bled, Slovenia, September 2009. [96]

- *Report on INEX 2008*. **Gianluca Demartini**, Ludovic Denoyer, Antoine Doucet, Khairun Nisa Fachry, Patrick Gallinari, Shlomo Geva, Wei-Che Huang, Tereza Iofciu, Jaap Kamps, Gabriella Kazai, Marijn Koolen, Monica Landoni, Ragnar Nordlie, Nils Pharo, Ralf Schenkel, Martin Theobald, Andrew Trotman, Arjen P. de Vries, Alan Woodley, Jianhan Zhu. In: SIGIR Forum, June 2009, Volume 43 Number 1, pp 22-28. [61]

- *How to Trace and Revise Identities*. Julien Gaugaz, Jakub Zakrzewski, **Gianluca Demartini**, and Wolfgang Nejdl. In: 6th Annual European Semantic Web Conference (ESWC2009), Heraklion, Greece, June 2009. [87]

- *L3S at INEX 2008: Retrieving Entities using Structured Information*. Nick Craswell, **Gianluca Demartini**, Julien Gaugaz, and Tereza Iofciu. In: 7th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2008 Dagstuhl Castle, Germany, December 2008. [50]

- *Social Recommendations of Content and Metadata*. Rodolfo Stecher, **Gianluca Demartini**, and Claudia Niedere. In: 10th International Conference on Information Integration and Web-based Applications & Services (iiWAS2008), Linz, Austria, November 2008. [155]

- *Finding Experts on the Semantic Desktop*. **Gianluca Demartini** and Claudia Niedere. In: Personal Identification and Collaborations: Knowledge Mediation and Extraction (PICKME 2008) Workshop at ISWC 2008, Karlsruhe, Germany, October, 2008. [71]

- *A Model for Ranking Entities and Its Application to Wikipedia*. **Gianluca Demartini**, Claudiu S. Firan, Tereza Iofciu, Ralf Krestel, and Wolfgang Nejdl. In: 6th Latin American Web Congress (LA-WEB 2008), Vila Velha, Espirito Santo, Brasil, October 2008. [64]

- *Semantically Enhanced Entity Ranking*. **Gianluca Demartini**, Claudiu S. Firan, Tereza Iofciu, and Wolfgang Nejdl. In: The Ninth International Conference on Web Information Systems Engineering (WISE 2008), Auckland, New Zealand, September 2008. [66]

- *Evaluating Relation Retrieval for Entities and Experts*. Jianhan Zhu, Arjen P. de Vries, **Gianluca Demartini**, and Tereza Iofciu. In: Future Challenges in Expertise Retrieval (fCHER 2008), SIGIR 2008 Workshop, Singapore, July, 2008. [173]

- *Entity Identifiers for Lineage Preservation*. Julien Gaugaz and **Gianluca Demartini**. In: 1st international workshop on Identity and Reference on the Semantic Web (IRSW2008) hosted by the 5th European Semantic Web Conference ESWC-08, Tenerife, Spain, June, 2008. [86]

- *Comparing People in the Enterprise*. **Gianluca Demartini**. In: 10th International Conference on Enterprise Information Systems, ICEIS 2008, Barcelona, Spain, June, 2008. [56]

- *How Many Experts? - A New Task for Enterprise Search Evaluation*. **Gianluca Demartini**. In: Workshop on Novel Methodologies for Evaluation in Information Retrieval at the 30th European Conference on IR Research, ECIR 2008, Glasgow, Scotland, April 2008. [57]

- *Ranking Categories for Web Search*. **Gianluca Demartini**, Paul-Alexandru Chirita, Ingo Brunkhorst, and Wolfgang Nejdl. In: Advances in Information Retrieval, 30th European Conference on IR Research, ECIR 2008, Glasgow, Scotland, April, 2008. [59]

- *Evaluating Personal Information Management Using an Activity Logs Enriched Desktop Dataset*. Sergey Chernov, **Gianluca Demartini**, Eelco Herder, Michal Kopycki, and Wolfgang Nejdl. In: "Personal Information Management Workshop at CHI 2008" PIM 2008, Florence, Italy, April 2008. [43]

- *L3S at INEX 2007: Query Expansion for Entity Ranking Using a Highly Accurate Ontology*. **Gianluca Demartini**, Claudiu S.Firan, and Tereza Iofciu.

In: "Focused Access to XML Documents", 6th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2007 Dagstuhl Castle, Germany, December, 2007. Selected Papers. [63]

- *Finding Experts Using Wikipedia*. **Gianluca Demartini**. In: "Finding Experts on the Web with Semantics Workshop at ISWC 2007 + ASWC 2007" FEWS 2007, Busan, South Korea, November 2007. [54]

- *Leveraging Semantic Technologies for Enterprise Search*. **Gianluca Demartini**. In: "First Ph.D. Workshop in CIKM" PIKM 2007, Lisboa, Portugal, November 2007. [55]

- *Building a Desktop Search Test-bed*. Sergey Chernov, Pavel Serdyukov, Paul-Alexandru Chirita, **Gianluca Demartini**, and Wolfgang Nejdl. In: Advances in Information Retrieval, 29th European Conference on IR Research, ECIR 2007, Rome, Italy, April, 2007. LNCS 4425, pp. 686-690. [44]

- *L3S Research Center at TREC 2006 Enterprise Track*. Sergey Chernov, **Gianluca Demartini**, and Julien Gaugaz. In: The Fifteen Text REtrieval Conference Proceedings (TREC 2006) [42]

Contents

List of Figures

1 Motivation and Overview

1.1 Challenges

The World Wide Web today is a global phenomenon that affects not only business but also the life of billions of people everywhere in the world. Thus, the Web is a reflection of our society as well as of the World economy. In the last years, the Web has become a huge repository of data and the estimated number of static Web pages is in the tens of billions. Because of this, new disciplines, such as "Web Science", were born to face the new challenges (e.g., "understand what the Web is, engineer its future and ensure its social benefit"[1]) such environment poses to the research communities.

At the current status of the Web, the main entry points for people to the vast amount of Web content are Search Engines. They allow people to navigate to specific Web pages, to discover content and information about new topics, and to get things done such as, for example, accessing on-line stores. Although such functionalities are critical for an effective usage of the Web, the result a Search Engine presents to a user, given a keyword query, is a simple list of ten links to retrieved Web pages.

The current architecture of Web Search Engines consists of the following steps: crawling, indexing, retrieval, and ranking. In order to go beyond the current state of the art, next generation Web Search Engines will need, among others, to perform a deeper analysis of available content for presenting the user with a more structured search result that, more than a list of links, allows to better answer the information need [12]. Instead of merely exploiting the syntactic structure of the Web and its documents, it is now possible to leverage semantic information about Web resources. Thus, Web search stops to be about documents rather, it is an interface for finding Web-mediated solutions of user goals about any type of entity.

Therefore, current research challenges include the extraction of information buried within Web pages that can be aggregated and presented to the end user. One example

[1] http://www.webscience.org/

is the query "New York restaurants" which is not aiming at a ranked list of Websites any of which could provide information about a restaurant in New York; rather, the user would be best satisfied by a list of entities with additional information such as the average price and, possibly, a map displaying the results.

Semantic technologies are now in a state able to significantly contribute to IR problems. Thus, general-use ontologies can be exploited to match entity types and well-structured repositories such as Wikipedia are a source of easily processable information for dealing with more complex user queries. For going beyond current Web search, the next step is to rank, rather than documents, information units of varying type, complexity, and structure, that is, *entities*. Being able to retrieve entities rather than documents would allow current Search Engines to answer more complex user queries such as "Countries where I can pay in Euro" or "Italian Nobel Prize winners".

In the commercial setting, new prototypes going in this direction are being developed. Examples of such systems include Powerset, Yahoo! Correlator, and Google Squared. Powerset was the first example of an entity search system based on Wikipedia. This start-up company, later acquired by Microsoft, provided a system for searching and browsing information contained in Wikipedia. By typing "Albert Einstein" the user is presented with a page summarizing relevant information about this entity: date of birth, date of death, etc., also including verbs and objects related to the requested entity (e.g., "published - theory"). Yahoo! Correlator[2] provides a new way to search the Wikipedia and finds entities related to the requested one. For example, by searching for "Albert Einstein" we can find as related entity "Max Born", a colleague of him. Google Squared[3] is a tool that allows to retrieve a list of entities relevant to a user query. For example, if we search for "physicists", the system shows us a table containing people who did research on physics. This is the most complete system as, more than just extracting entities and related information, it is able to aggregate them in order to respond to specific user queries requesting a list of relevant entities.

1.2 Contributions

In this thesis we provide a twofold contribution to the Information Retrieval field. First, we identify methods for addressing the problem of Entity Retrieval (ER) proposing and evaluating different models for both the Enterprise and the Wikipedia setting also creating reusable test collections for evaluating effectiveness of ER systems. In the Enterprise setting we focus on the problem of expert finding, that is, finding people who are knowledgeable about the requested topic. In the Wikipedia setting we target general typed search where retrieved entities are represented by their Wikipedia pages.

Moreover, we consider additional dimensions of ER, namely, time and opinions. We study the problem of ER over time in the news setting. We exploit the time dimension

[2] http://correlator.sandbox.yahoo.com
[3] http://www.google.com/squared

and, in particular, the history of a news trail, in order to improve effectiveness of finding relevant entities about the user query which are mentioned in a news article. We also address the difficult problem of mining public opinion from the Web and, in particular, from the blogosphere. We propose and evaluate methods for predicting public opinion about candidates of a political election. We leverage sentiments expressed in blog postings and, by means of aggregation, predict the general feeling about the candidates over time.

The thesis is organized around the different settings where the ER problem can be addressed. Chapter 2 introduces general notions in the context of IR, Semantic Web, and describes some of the general characteristics of Wikipedia which are important for understanding the following chapters.

The next three chapters start with an introduction to the area, followed by a review of related work specific to that particular domain also comparing it to our contributions. Then, proposed models and algorithms are first defined and explained, and then experimentally evaluated. We conclude each chapter with a discussion on the advantages and disadvantages of each proposed algorithm, as well as on the possible further steps.

Chapter 3 addresses the problem of finding entities in an Enterprise setting. Specifically we target the expert finding task where the user is looking for a person who is expert about the topic described by the issued query. In order to deal with such problem we propose a general model for finding entities and we show how this can be applied to different entity search scenarios. We then apply it to the expert finding problem and evaluate it on standard test collections.

In Chapter 4 we continue to address the ER task by focusing on the Wikipedia setting. We first present our efforts on creating standard and reusable test collections for evaluating effectiveness of ER systems in the Wikipedia setting. We then propose a model based on a structured representation of entities and several algorithms based on Link Analysis, Natural Language Processing, and Named Entity Recognition. We conclude the chapter by a discussion about which user queries are easy and which are difficult to answer for current ER systems.

Chapter 5 makes an additional step ahead by addressing different dimensions of the ER problem. We first focus on the time dimension and on ER in the news setting. We propose methods for retrieving the most important entities in a news article given a user query. We exploit both features from the current news article as well as from past related articles experimentally showing how evidence from the history is critical for improving the effectiveness of such newly defined task. Next, we look at the problem of mining opinions about entities. Here, we focus on the political domain by estimating public opinion about electoral candidates. We adopt both sentiment-based lexicon as well as text classification techniques together with aggregation methods in order to predict opinion trends in the blogosphere.

Chapter 6 concludes this thesis enumerating the contributions we brought to the Information Retrieval field, while also discussing future directions and open challenges associated to the discussed topics.

2 Technical Basis

In order to perform the challenging task of retrieving entities, several approaches from different research areas need to be considered and applied. In this chapter we present a brief overview of different techniques used in this thesis. In detail, we provide an overview of Web Information Retrieval and Enterprise Search with particular attention to the different search tasks considered in such settings. We then present relevant concepts from the Semantic Web field such as ontologies. We define the problem of clustering and classification addressed in the Machine Learning area also mentioning algorithms for them. Finally, we describe Wikipedia as a research dataset which is widely used throughout the thesis.

2.1 Information Retrieval

The term "Information Retrieval" (IR) is classically defined as finding unstructured documents that satisfy an information need from within large collections [121]. In the recent years advances have been made moving from the unstructured scenario (i.e., textual documents) to a more structured representation of documents as well as to other types of material to be found (e.g., pictures, videos, etc.).

2.1.1 Web Search

When we consider an IR system on the Web we have to deal with several differences with traditional document collections. Such differences include, just to mention a few, scale, link structure, evolution, diversity of content, spam, etc.

A Web Search Engine is a system designed to search for information on the Web. Results of a query to a Search Engine are usually presented as a list of Web pages, images, or other types of file. In this thesis we propose methods for including into the result page a list of entities rather than a list of documents.

A key aspect to be taken into account by Web Search Engines is the user information

need. In detail, different users of Web Search Engines may have in mind different search tasks. Broder [35] proposed a taxonomy of such tasks classifying Web queries in:

- **Navigational**. The immediate intent is to reach a particular site.
- **Informational**. The intent is to acquire some information assumed to be present on one or more Web pages.
- **Transactional**. The intent is to perform some Web-mediated activity.

In this thesis we focus on ER and we propose in Section 3.1.2 a taxonomy describing different types of entity-related queries.

Another important aspect of Web Search is the use of the link structure present among Web Pages. A lot of work has been analysing and exploiting such structure where the most famous work is the PageRank algorithm [131] that contributed to the initial success of the Google Search Engine. The idea of the algorithm is to leverage the number of incoming links to compute the "authority" score of a Web page. Such scores are then propagated to other pages via the outgoing links. The computation is usually done by a random walk algorithm on the Web graph and used as a query-independent document weight. In this thesis we exploit the link structure in Wikipedia in order to improve effectiveness of ER (see Section 4.6).

2.1.2 Enterprise Search

Searching within an enterprise, and finding what you need, is not an easy task as it is on the Web. However, the expectations of corporate workers are rising due to the effectiveness of their satisfactory Web Search experience. An Enterprise Search (ES) systems should make accessible the collective intelligence, translating it into a huge competitive advantage for the enterprise increasing innovation, productivity and agility. Tools for ES are becoming more and more important due to the necessity of handling large amount of data in the context of the enterprise.

A good definition of what ES includes has been provided by Hawking in [92]:

- search of the organisation's external Website;
- search of the organisation's internal Website (its Intranet);
- search of other electronic text held by the organisation in the form of email, database records, documents on fileshares and the like.

This definition shows that there are different aspects to take into account such as Intranet search; search in a P2P network, given that each employee shares some file on her desktop; search in both structured (databases) and unstructured (documents) data; search for not only files (people search).

Some work has been already done in the field of ES but several open issues are still asking to be solved. The sub problem of Intranet search has been analysed in [82] where they show the differences between ES and Web search and how to improve the performances of Intranet search, and in [172] where they propose methods for improving the effectiveness of ES on navigational queries using dictionaries and geographically tagging Intranet pages. A list of the most common ES users' needs is presented in [110] and it is compared with the one for the Web proposed by Broder [35]. One of such tasks is expert finding. In this thesis (see Chapter 3) we propose to consider experts as well as documents as retrievable units in a vector space.

2.2 Semantic Web

The Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries.[1] Thus, the Semantic Web mainly deals with structured data on the Web and studies techniques for representing and reasoning over such data. This allows a more formal representation of knowledge that the unstructured text is not able to provide. Examples of such structured representation are ontologies. In general, Semantic Web technologies enable Search systems to answer more complex user queries.

2.2.1 Ontologies

An ontology is a formal specification of a shared conceptualization [90]. Thus, many domain-specific ontologies exists and are being reused. For example, the FOAF ontology allows to describe statements about persons and their relations. The Dublin Core ontology is basic vocabulary to describe documents.

Recently an effort of automatically creating general-purpose ontologies has been made. One of the first such ontologies was YAGO: a semantic knowledge base containing more than 2 million entities and 20 million statements. It is automatically extracted from Wikipedia and uses WordNet to structure information. In the context of this thesis we exploit the YAGO ontology for improving the quality of ER in Wikipedia by refining entity types in the hierarchy provided within YAGO (see Section 4.6).

2.2.2 Semantic Search

Systems that apply Semantic Web technologies for improving IR effectiveness are usually grouped under the name of Semantic Search. They could either be used to better understand the user intent when only a keyword query is provided or to perform structured search over

[1] http://www.w3.org/2001/sw/

the (structured) Web of data. In this thesis we exploit the first type of such techniques when performing query expansion using lexical resources like Wordnet (see Section 4.7).

Wordnet [84] is a resource that groups words into sets of synonyms expressing the same concept. Moreover, several types of relations between concepts are provided. In this way related words can be found. For example, the following relations between names are described:

- hypernyms: Y is a hypernym of X if every X is a (kind of) Y (canine is a hypernym of dog, because every dog is a member of the larger category of canines);
- hyponyms: Y is a hyponym of X if every Y is a (kind of) X (dog is a hyponym of canine); coordinate terms: Y is a coordinate term of X if X and Y share a hypernym (wolf is a coordinate term of dog, and dog is a coordinate term of wolf)
- holonym: Y is a holonym of X if X is a part of Y (building is a holonym of window);
- meronym: Y is a meronym of X if Y is a part of X (window is a meronym of building).

2.3 Machine Learning

An important field of research for IR algorithms is that of Machine Learning. This is usually defined as the design of algorithms that can adapt their behaviour based on available example data. In detail, we can distinguish two main classes of machine learning algorithms: supervised and unsupervised.

2.3.1 Classification

Supervised machine learning algorithms, also know as classification algorithms, exploit labelled examples to learn how to separate unseen instances in two or more classes.

One of the best performing model is Supporting Vector Machines (SVMs). Given a set of training examples, each marked as belonging to one of two categories, an SVM algorithm builds a model that predicts whether a new example falls into one category or the other. More formally, a support vector machine constructs a hyperplane in a high dimensional space, which can be used for classification. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training datapoints of any class. In this thesis we exploit linear SVMs to automatically classify blog postings as carrier of positive and negative opinions for predicting the public opinion about political candidates (see Section 5.4).

2.3.2 Clustering

Clustering techniques are the most important type of unsupervised machine learning algorithms. The goal of clustering algorithms is also to group items in separate sets (i.e., clusters) according to some measure of similarity. The main difference with classification algorithms is that no initial training is provided.

One of the most famous clustering algorithm is k-means. It attempts to find the centres of natural clusters in the data by iteratively refining the solution. The first step assigns each instance to the cluster with the closest mean. The second step calculates the new mean to be the centroid of the instances in the cluster.

2.4 Wikipedia

Wikipedia is a multilingual, Web-based, free-content encyclopedia project based on an openly-editable model.[2] Such model made possible for Wikipedia to increase the chances that factual errors or misleading statements are promptly corrected. This makes most Wikipedia articles being balanced, neutral, and encyclopedic, containing notable verifiable knowledge. For all these reasons Wikipedia is a great source of knowledge freely available in textual format. Of course, it is important to use Wikipedia carefully, since individual articles may vary in quality and maturity.

In this thesis we use Wikipedia as a repository of entity descriptions on top of which we run ER queries for which our system will return a ranked list of entities represented by their Wikipedia page (see Chapter 4).

[2] http://en.wikipedia.org/wiki/Wikipedia:About

3 Retrieving Entities in the Enterprise

3.1 Introduction

3.1.1 Motivation

Finding entities on the Web is a new search task which goes beyond the classic document search. While for informational search tasks (see [35] for a classification) document search can give satisfying results for the user, different approaches should be followed when the user is looking for specific entities. For example, when the user wants to find a list of "European female politicians" it is easy for a classical Search Engine to return documents about politics in Europe. It is left to the user to extract the information about the requested entities from the provided results. Our goal in this thesis is to develop a system that can find entities and not just documents on the Web (see Figure 3.1).

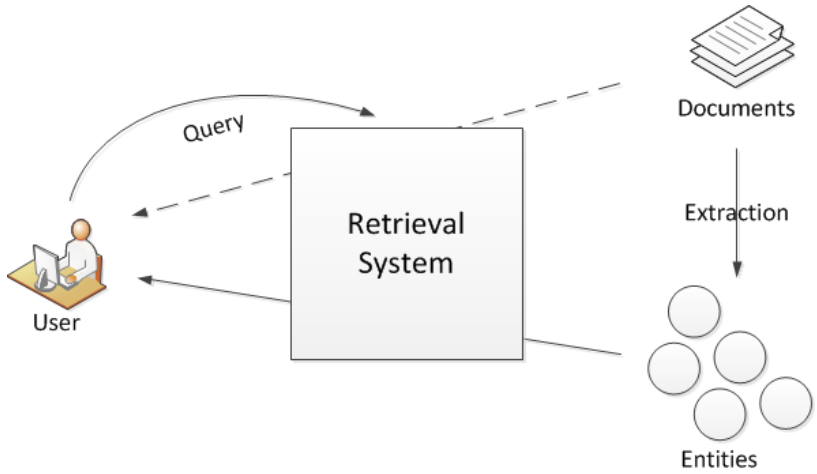


Figure 3.1 Entity vs Document Retrieval

Being able to find entities on the Web can become a new important feature of current

Search Engines. It can allow users to find more than just Web pages, but also people, phone numbers, books, movies, cars, etc. Searching for entities in a collection of documents is not an easy task. Currently, we can see the Web as a set of interlinked pages of different types, e.g., describing tasks, answering questions or describing people. Therefore, in order to find entities, it is necessary to do a preprocessing step of identifying entities in the documents. Moreover, we need to build descriptions of those entities to enable Search Engines to rank and find them given a user query. Applying classical IR methodologies for finding entities can lead to low effectiveness as seen in previous approaches [25, 40, 135]. This is because Entity Retrieval, that is, finding entities relevant to a query, is a task different than document search. An example of an ER query is "Airports in Germany" where a relevant result is, e.g., "Frankfurt-Hahn Airport". Airports not in Germany or entities other than airports would not be relevant to the given query. It is crucial to rely on consolidated information extraction technologies if we do not want to start with an already high error that the ranking algorithms can only increase.

3.1.2 A Taxonomy of Entity-Related Search Tasks

With the current size of the Web and the variety of data it contains, traditional Search Engines are restricted to simple information needs. Despite of this, complex and long queries are becoming more common and need, usually, a lot of effort on the user side in order to be satisfied. In such category we can find several entity related search tasks:

Expert Finding. In the Enterprise context often people need to search for experts. We can define expert finding as the task of finding a person (employee, associate, or anyone else in the enterprise) who has a particular knowledge on a given topic. Being able to search for people that are experts about a given topic (e.g., "C++ compilers") allows to easily build a working team or to find a person who can solve a certain problem.

Entity Retrieval. Finding entities of different types is a challenging search task which goes beyond classic document retrieval as well as beyond single-type entity retrieval such as, for example, the popular task of expert finding [14]. The motivation for the ER task is that many user queries are not looking for documents to learn about a topic, but really seek a list of specific entities: countries, actors, songs, etc. Examples of such informational needs include 'Formula 1 drivers that won the Monaco Grand Prix', 'Female singer and songwriter born in Canada', 'Swiss cantons where they speak German', and 'Coldplay band members'. The query 'countries where I can pay in Euro' is answered by current Web Search Engine with a list of pages on the topic 'Euro zone', or ways to pay in Euros, but not with a list of country names as the user is asking for. Note that while a single query refers to a single entity type, a system must be able to answer queries for different entity types (differently from an expert finding system where the response is always of type person). A commercial prototype performing this task is Google Squared[1].

[1] http://www.google.com/squared

Entity List Completion. A task similar to Entity Retrieval is Entity List Completion. In this case the user, more than just providing a keyword query, inputs to the retrieval system some examples of relevant entities. For example, for the query 'countries where I can pay in Euro' the user would also select 'Germany', 'Italy', and 'Spain'. The system task is defined as returning all the relevant entities not yet provided by the user.

Question Answering. It must also be mentioned how Entity Retrieval task relates with Question Answering (QA). Common queries in the QA context usually are of type Who, When, Where, Why, How Many [1]. That is, they expect a precise answer as, for example, a number or a name instead of a list of entities. ER queries have considerable similarities with QA "list" questions where the user is looking for a list of items as a result (e.g., "What companies has AARP endorsed?"). In the evaluation benchmarks, QA queries usually consist of sets of questions about a particular topic: this might let the system approach the problem in a different way, e.g., by mining documents retrieved with a keyword query or by exploiting the answer of previous questions on the same topic (e.g., "What does AARP stand for?"). In conclusion, there are similarities between ER and QA queries. In particular for list QA queries we can imagine ER technologies described in this chapter exploited, among other things, by QA systems to perform better on this particular type of queries.

Related Entities. Another related task is finding entities similar or related to other entities. In this case the user might have in mind a search query consisting of an example entity. For a given entity, such as "New York", one would expect to find as associated entities places to visit in New York (e.g., "Empire State Building", "Statue of Liberty"), connected historical events (e.g., "September 11, 2001") or famous people (e.g., "Rudy Giuliani"), etc. in a faceted-search fashion. The associated entities can be presented to the user as a lists or grouped by type and other properties (e.g., date). For a query "Albert Einstein", the system may return related entities like, for example, "Germany", "Nobel prize", "physics", "Lieserl Einstein", etc. This task is different from ER as the result set may contain entities of different types. Here the system provides the user with a browsing opportunity rather than with a list of retrieved entities as for ER. A commercial prototype performing this task is Yahoo! Correlator[2].

Navigational vs Informational Entity Retrieval Queries. An additional distinction can be made in the context of the Entity Retrieval task. In some cases the user might be looking for (the identifier of) one specific entity. In this case the user query will provide a structured or unstructured entity description to the system which is required to retrieve the correct entity identifier. One example query of such search task is "Albert Einstein". This task can be mapped to *navigational* queries in the context of Web search.

[2] http://correlator.sandbox.yahoo.net

The other kind of search task, which is the one we address in this thesis, is defined as a user looking for a list of entities as answer to her query. The user provides the system with some properties of the required entities (e.g., "Nobel Prize winners in physics") looking for the names (or labels) of relevant entities. This task can be mapped to *informational* queries in the context of Web search.

Expert Finding vs People Search. Another distinction must be made between expert finding and Entity Retrieval queries dealing with people. In the case of expert finding the user is looking for a list of persons whose expertise matches that described in the query (e.g., "C++ compilers"). On the other hand, in the case of general people search, the query describes properties that retrieved entities should have in order to be relevant (e.g., "Nobel Prize winners"). While similar techniques might be used by retrieval systems (e.g., create profiles for indexed entities) some differentiation must be made at query processing time. For such reasons, in this thesis we present two different models for dealing with the expert finding setting in Chapter 3 and dealing with general Entity Retrieval in Chapter 4 respectively.

Figure 3.2 relates all the different search tasks described in this section.

3.1.3 Contributions

In this chapter we aim at creating a system for finding entities which is independent of the considered entity type. It is not easy to design a system that can find entities because usually different types of entities require different types of search approaches. It is easier, on the other hand, to develop and provide the users with solutions for specific entity types (e.g., people, chemical entities, phone numbers, events).

In the past, systems focusing on specific entity types have been proposed. As an example, there is a lot of work done in the area of Expert Finding. Applying classical IR methodologies for finding entities can lead to low effectiveness as seen in previous approaches [25, 40]. This shows again how entity search is a different task than document search.

In detail, we propose a general model for finding entities and we show how this can be applied to different entity search scenarios. We extend the classical Vector Space Model representing entities in it as a weighted profile, a linear combination of text documents, resulting in entity vectors in the same vector space as documents. We first index a document collection as vectors in the space. Then, we extract entities mentioned in the documents and we create a weighted profile for each entity thus building a vector representation of it that can be added to the same space where the documents lay. We experimentally show how customized techniques can be integrated in the search process and we evaluate the prototype system on the TREC Expert Search scenario as proof of concept.

The main contributions of this chapter are: 1) A general model for Entity Retrieval (Section 3.3); and 2) Application of the model to the enterprise setting in order to find

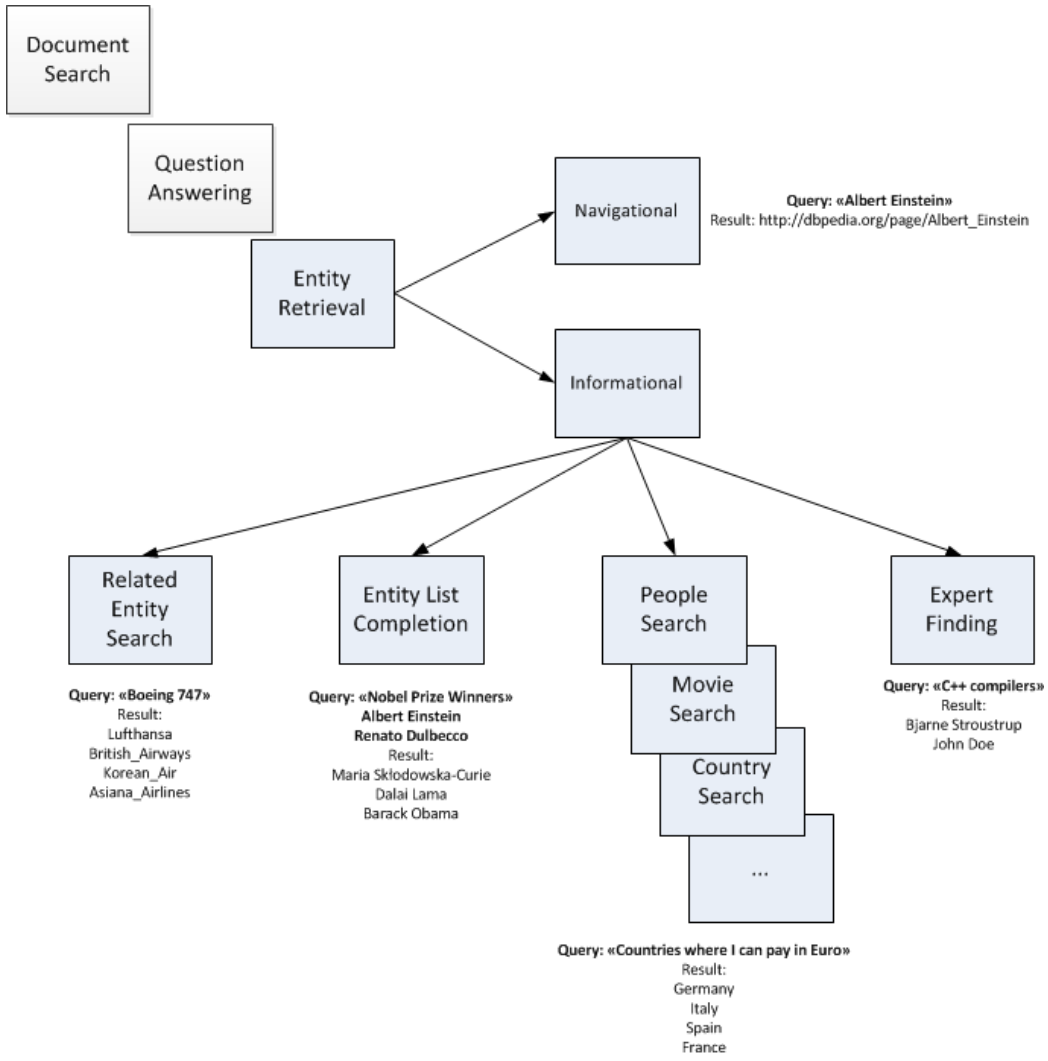


Figure 3.2 A taxonomy of Entity-related search tasks.

experts (Section 3.4).

The chapter is structured as follows: In Section 3.2 we present the related work in the area of Entity Retrieval and Expert Finding. In Section 3.3 we define the formal model for Entity Retrieval together with its possible extensions for including several evidences of relevance. In Section 3.4 we apply the model to the scenario of Expert Finding evaluating its effectiveness on a standard testbed. Finally, in Section 3.5 we conclude the chapter also describing possible future work.

3.2 Related Work

3.2.1 Entity Search

Finding entities instead of documents is a recent topic in the field of IR. The earliest approaches [25, 39, 40, 41] mainly focus on scaling efficiently with the dimension of the dataset rather than on the search effectiveness. The goal of this chapter is to focus on the precision in the Entity Retrieval task providing a system that can find entities incorporating several types of relevance evidence.

A formal model for finding entities has been presented in [64]. This model represents entities as a set of $(<attribute>, <value>)$ pairs together with the entity type information. In this chapter we propose a model for the Entity Retrieval process where the entity representation is a simpler vector representation based on the original vector space model [144]. A framework for modelling the IR process has been presented in [143] where the authors present a matrix-based framework for modelling possible search tasks. The model we propose is focused only on Entity Retrieval while being more intuitive.

Approaches for finding entities have also been developed on the Wikipedia dataset. For example, Pehcevski et al. [135] use the link information for improving the effectiveness of Entity Retrieval in Wikipedia. In [66] the authors improve the effectiveness leveraging on a highly accurate ontology for refining the search on the Wikipedia category hierarchy. Similarly to the model we present in this chapter, in [64] the authors present a model for Entity Retrieval making the development of algorithms possible also in domains different from Wikipedia. Also independent on the dataset, in [38] the author presents HubRank, a PageRank like proximity search system for entity-relation graphs. Another work which can be a foundation for an effective Entity Retrieval is the automatic identification of instances and classes in the Wikipedia category hierarchy [174]. Knowing which categories describe instances can help the system in finding entities relevant to the query.

3.2.2 Expert Finding

Many more systems and models have been proposed for expert finding. An interesting system related to our approach is the Enterprise PeopleFinder [122] also known as P@noptic Expert [49]. This system first builds a candidate profile attaching all documents related to that candidate in one big document giving different weights to the documents based on their type (homepages, for example, are more important than other Web pages). The system uses document terms as topics of expertise and candidate name matching (i.e., whether a name appears into the document or not) for connecting documents and experts. Differently from our model, relationships between candidates and documents are binary, i.e., a given document is either related to a candidate or not. Moreover, our model can incorporate several features as evidences of expertise.

A similar model for Expert Finding proposed in [116, 117] views expert finding as a voting problem. The documents associated to a candidate are viewed as votes for this

candidate's expertise and relevance feedback techniques are considered. In this model the relationships between candidates and documents are only binary and not continuous. The CombSUM approach first retrieves documents similar to the query. The model we propose first places experts in the space and then retrieves them. This also allows our model to retrieve a mix of documents and experts together.

In [118] the authors compare the effect of different evidences on the effectiveness of expert finding. They analyse the effect of considering candidate homepages, which can be easily included in our model using a document dependent extension. They also study the effect of proximity of query terms with the candidate name. This can be included in our model representing widows of text around the candidate name as separate vectors in the space. Additionally, they study the impact of URL length and inlinks which can be integrated with document dependent extensions. Finally, they study how clustering candidate profile can solve the problem of topic drift. In our model the projection similarity allows multiple expertise topics to be represented in the candidate profile without interference caused by topic drift.

Language models-based approaches [17, 18, 83] are the most popular techniques for ranking experts. These approaches either first create expert profiles using the document collection and then rank such profiles [20, 137, 149], or first retrieve relevant documents and then extract the relevant experts from them [136]. Other approaches also use graph based techniques [150], learning methods [95], or aim at exploiting evidence from the Web [148] or other Enterprise data [21].

3.3 A Vector Space Model for Ranking Entities

The model proposed in this section builds on the well-known vector space model, and represents entities as vectors in the same space together with documents and queries about them, allowing to retrieve relevant documents as well as entities with the same query. This allows us to re-use many techniques developed for IR to improve entity search effectiveness. The basic model is simple and easy to extend, using not only documents as relevance evidence, but also prior knowledge, such as time or the link structure.

3.3.1 Ranking Entities: Problem Definition

We assume a collection of documents $D = d_1, \ldots, d_m$ and a list of entities extracted from the documents $E = e_1, \ldots, e_n$. An entity is defined as a concept perceived by humans as a self-contained whole. Typical examples include persons, organizations, locations, and commercial products. Additionally, we have a set of topics[3], extracted from the collection of documents or predefined, $T = t_1, \ldots, t_l$ which will represent the dimensions of the space, and a query q. The task is then to retrieve a list of entities from E ranked by degree of

[3] Like terms or concepts, not to confound with queries.

relevance to q.

In this thesis we consider topics as dimensions of the space and entities as vectors in the space which are a linear combination of topics. A different approach may be to consider entities, instead as vectors in the space, as its dimensions. In such setting documents would then be represented as a linear combination of entities.

3.3.2 Formal Definition of the Basic Model

The model builds a multi-dimensional vector space S with inner product. We define on S a basis representing l topics, $T = \vec{t_1}, \ldots, \vec{t_l}$.

We further represent a set of m documents $d_i \in D$, with $1 \leq i \leq m$ as linear combination of the basis vectors, as usually done in the standard vector space model. We then have $\vec{d_i} = d_{1,i}\vec{t_1} + \ldots + d_{l,i}\vec{t_l}$, where the $d_{k,i}$'s are coefficients measuring how a topic vector $\vec{t_k}$ belongs to the document d_i. These coefficients can be, for example, TF×IDF ones if we represent the topic basis vectors $\vec{t_k}$ by terms.

Our model differs from the standard IR vector space model by also representing entities $e_j \in E$, $1 \leq j \leq n$ as vectors in this space. In order to do so, we define a function $f : D \times E \to \mathbb{R} : (d_i, e_j) \mapsto r_{i,j}$ which, for each entity $e_j \in E$ assigns a weight $r_{i,j}$ to each document d_i. This weight is defined as representing how much the entity e_j is relevant to the subject addressed by document d_i. Another view is to see the document d_i as evidence of the relevance of entity e_j. For example, the entity "Bob Dylan" will mostly appear in documents about music thus making it relevant to queries about musicians. In order to estimate the probability that an entity e_j appears in a document d_i (and use this as the weight $r_{i,j}$.) we can use information extraction techniques.

Given the set of document-vectors D and the function f, we can find the coordinates of an entity e_j in the vector space generated by the chosen topic basis T:

$$\vec{e_j} = \sum_{i=1}^{m} r_{i,j}\vec{d_i} = \sum_{i=1}^{m}\left(r_{i,j}\sum_{k=1}^{l} d_{k,i}\vec{t_k}\right) = \sum_{k=1}^{l}\left(\sum_{i=1}^{m} d_{k,i}r_{i,j}\right)\vec{t_k}$$

or in matrix form

$$\boxed{E = D \times R} \tag{3.1}$$

where

$$E = [\vec{e_1}|\ldots|\vec{e_n}],\ D = [\vec{d_1}|\ldots|\vec{d_m}]$$
$$R = \left[r_{i,j}\right] \text{ with } 1 \leq i \leq m, 1 \leq j \leq n$$

and n is the number of entities, m is the number of documents, $\vec{d_i}$ is the vector of the ith document, $\vec{e_j}$ is the vector of the jth entity, and $r_{i,j}$ is the relationship weight between the document d_i and the entity e_j. In summary, an entity representation can be seen as a linear combination of several documents vectors opportunely weighted based on $r_{i,j}$ scores as well as a point in the vector space directly.

Finally, the query is represented as a vector containing the topics for which we need appropriate entities or documents, i.e., $\vec{q} = q_1\vec{t_1} + \ldots + q_n\vec{t_n}$.

We then use a distance or similarity measure in order to rank entities and documents based on increasing distance (decreasing similarity) to the query vector. A default measure would be the well known cosine similarity $sim(\vec{q},\vec{v}) = \frac{\vec{q}\cdot\vec{v}}{\|\vec{q}\|\|\vec{v}\|}$ where "$\cdot$" denotes the dot-product of the two vectors and $\vec{v} \in \{d_i, e_j\}$.

Now that we have represented both entities and documents in the same vector space, we can retrieve both entities and documents relevant to a query. This is an advantage over pure document or entity search systems. We can even query for documents most representative of the domain of an entity, using an entity vector as query.

Duplicates in the document collections. Note that duplicates in the collection of documents can be problematic. If the same document is used several times for building the profile of an entity, the coordinates of the entity in the topics of the document will be artificially boosted. However, if two documents are just similar, but not duplicates, then the boosting of the entity's coordinates on the topics of the documents is desirable, since the appearance of the entity in the same context but different documents is an evidence of relevance. Known techniques such as fingerprinting can be used for duplicate detection.

3.3.3 Extensions of the Model

Different types of refinement can be applied in order to include more evidences of relevance. We will discuss three types of extensions: document dependent ones, entity and topic dependent ones, and entity dependent extensions.

Document dependent extensions. For these extensions, we add weights for each document and use a diagonal matrix to add them in our model: $E = D \times (diag(\vec{x}) \times R)$ where $diag(\vec{x})$ is a $m \times m$ diagonal matrix with values x_i. Each value x_i represents the weight assigned to the document d_i. This weight can, for example, represent time aspects, assigning for each document a weight proportional to the creation date of the document, valuing newer documents higher than old ones with the assumption that older documents might contain outdated evidence with respect to entity relevance.

Entity and Topic Dependent Extensions. The second kind of refinement takes *entity and topic dependent* aspects into account, relating the weight to each entity and to each (topic) dimension of the space. This can be used to model the entity type information. For example, we can add more weight for all entities of a certain type (e.g., movies) on the respective dimension of the space. To represent this kind of refinement we can compute $E' = E \circ W$ where W is a matrix of the same dimension of E and $\circ$ indicates the Hadamard product between matrices. Each element $w_{j,k}$ of W indicates the weight for the dimension k of the entity e_j.

Entity Dependent Extension. Finally, we could refine the results using relevance evidence weights depending only on the entity. We can for example use a *cost function*, to prefer popular (more linked, using for example PageRank) entities, or to prefer newer entities as in the case of movies. This can be represented as a transformation $E'' = E' \times diag(\vec{cf})$ where $diag(\vec{cf})$ is a $l \times l$ diagonal matrix with values cf_k, each value representing the cost associated with the entity e_j.

3.3.4 Vector Space Basis

Different ways of representing topics are possible. The following paragraphs illustrate three different ways of building and selecting the reference topics used as basis of the vector space (T).

Terms (TF×IDF). This is the classical approach, where the dimensions of the space are given by all the terms in the collection and each document is represented as a vector containing the TF×IDF values of all terms in the document. The coordinates of a document are then the TF×IDF values related to the corresponding term basis [121]. Using this approach each entity is represented as a vector containing a value for each term in the document collection representing its relevance on the topic represented by that term.

Terms & LSA. Another relevant technique is to use latent semantic analysis (LSA) [53] which aims at solving the problem of synonymy and polysemy. LSA is able to manipulate and significantly reduce the dimensions of the space and to improve performance compared to TF×IDF without dimension reduction by 30% in the case of document retrieval [77].

Lexical compounds. Finally, we investigated the use of lexical compounds for keyword extraction. As Anick and Tipirneni have shown in [9], we can extract from a set of documents a list of "themes", as key concepts of that set. For example "business process model" is thus recognized as a topic and can be used to represent entities on that topic. In [45, 9] the authors have presented very good results of this technique for query expansion tasks. In our context we used this method in order to extract the key properties of an entity from the set of documents related to it, and used this evidence to place the entity in the vector space as described above.

3.4 Applying the Model to Expert Finding

In this section we describe a possible application of the presented model, namely, Expert Finding. We want to show how the proposed model is applicable to several entity search contexts, independently on the types of documents and entities considered. For this reason we decided to use the Expert Search collection developed in TREC [48]. When using this

collection we can consider people as a special type of entities. Thus, using the proposed model, we can find entities as well as documents supporting their relevance.

In this section we also describe the experiments we conducted. Those aimed at 1) showing that we achieve reasonable results on the expert finding task with the generic model we described, and 2) showing how we can use some of the many known techniques in IR in order to influence the behaviour of the system. We will focus on document dependent extensions as appropriate relevance assessments and data are still missing for entity and topic dependent extensions.

3.4.1 The TRECent 2006 Collection

The Enterprise Search Track (TRECent) [48] started in TREC 2005 as a pilot track and since 2006 to 2008 took place with a complete set of relevance assessments. In TRECent 2006, participants used the W3C test collection [4], consisting of a document collection with about 330,000 HTML pages—mailing list entries, people home pages, CVS logs, W3C Web pages and wiki pages—the relevance assessments from the human evaluators, a list of topics, and a list of expert entities which contain the name and the e-mail address of persons to be considered as experts candidates for the topics. We used the 2006 topics composed of title, description, and narrative parts, representing a query as concatenation of these constituents. In 2007 and 2008 TRECent used the CSIRO collection [13]. We decided to use the W3C collection because the list of considered entities (i.e., the candidate experts) was officially provided by the organizers. This allows us to focus on the aggregation of expertise evidence rather than on the identification of candidates.

3.4.2 Projection Similarity

Long documents are usually not more relevant to a query than small ones. For this reason, in document retrieval, to avoid favouring long over small documents, measures like cosine similarity are used, which normalizes document length. If we want to retrieve expert entities, though, we do prefer to retrieve a person having more expertise on the topic expressed by the query. We will therefore define a measure we call *projection similarity*, which models the amount of expertise in the query topic by an *orthogonal projection* of the candidate vector onto the query vector, defined as $projSim(\vec{q},\vec{v}) = cos\theta \, \|\vec{v}\| = \frac{\vec{q}\cdot\vec{v}}{\|\vec{q}\|}$, where $\vec{v}$ is an entity vector, $\vec{q}$ is the query vector and "$\cdot$" denotes the dot-product of two vectors. Projection similarity favours long vectors over shorter ones, as we multiply the length of the vector $\vec{v}$ with the cosine of the angle between $\vec{v}$ and $\vec{q}$. So the longer the expert entity vector the higher the similarity score.

[4] http://research.microsoft.com/users/nickcr/w3c-summary.html

Table 3.1 Expert entities extraction effectiveness. The high number of document per candidate is mainly due to the large proportion of emails in the collection.

collection	%cand	%rel_cand	#avg	%docs
2006	71.38%	97.89%	1246	40.89%

3.4.3 Experimental Setup

We implemented an entity search system which incorporates the model defined in this chapter and applied it to the expert finding task as a possible application of entity search. As output, our system produces a run file in TRECent format that allows us to use the official `trec_eval` utility program to compute the common evaluation measures in order to compare the effectiveness of our system with TRECent participants.

Vector space bases and pruning methods. We compared three different indexing methods—term based, LSA based, and lexical compounds based—to investigate different possibilities of defining topics of expertise as described in Section 3.3.4.

Furthermore, we tested different pruning schemes on the term basis. We used all the terms occurring in the document and then pruned the space dimensions considering only the first k basis vectors ordered by document frequency where k is the rank where we reach 80% of the total document frequency in the document collection. We also tried to remove noise from the collection considering only terms consisting of letters.

Relationship weights. To compute the relationship matrix R (see Section 3.3.2) we used a version of the collection which contains tags marking occurrences of candidates' names and e-mails[5].

To associate documents with candidates, we set the relationship weight to one when a candidate name, email or username as defined in the annotated W3C corpus appears in a document. Alternatively, we also experimented weighing differently candidates occurring as author of an email than candidates occurring only in the body of the email. We intend to investigate further weighing alternatives in the future. Table 3.1 shows how effectively we can identify the candidates in this tagged collection, where[6] *%cand* is the percentage of candidates identified in the collection, *%rel_cand* is the percentage of identified candidates which are relevant, *#avg* is the average number of documents associated (that is, $r_{i,j} > 0$) with a candidate, and *%docs* is the percentage of document d_i with at least one relationship $r_{i,j} > 0$.

Document dependent extensions. To experiment and evaluate the document dependent extensions discussed in Section 3.3.3 we used the PageRank values of the documents.

[5] http://ir.nist.gov/w3c/contrib/W3Ctagged.html

[6] The notation reflects the one used in [17].

These values are computed using the link structure of the HTML document collection[7]. We note that, even though PageRank has been shown not to work very well in the enterprise scenario, the collection we used is based on a public Web crawl so PageRank should be suited to identify authoritative documents. As described in Section 3.3.3, we used a document dependent extension multiplying the values in the relationship matrix with the PageRank value of each document.

Similarity functions. We investigated the performances of two different similarity measures comparing query and retrievable vectors: cosine similarity and the projection similarity presented in Section 3.4.2. Our hypothesis was that projection similarity can improve expert retrieval effectiveness.

3.4.4 Experimental Results

All experiments were performed on the TRECEnt 2006 Expert Search topics and the W3C test collection. To compare performances of the different variants we tested, we ran the system varying only one variable at a time, keeping the others fixed. As *reference* run we use the run with the following properties: 1) terms as a basis 2) pruning method described in Section 3.4.3, considering only terms containing letters 3) projection similarity measure 4) weights (1/1) representing the occurrence of the candidates in either author or text fields, and 5) PageRank values of the documents *not* used. As we considered this as our baseline, all mean average precision (MAP) values presented below are followed by the p-value of the t-test between the various runs and the reference run, noted in parentheses. We considered a difference to be statistically significant when $p \leq 0.05$.

In figure 3.3(a) we show all runs submitted to TRECent 2006 ordered by MAP, as well as our system. Notice that our best run has a MAP at around two third of the best run at TRECent 2006. This is sufficient, considering the role of model validation of our experiments. Several options are possible to improve our score, choosing among the plethora of available existing techniques for the vector space model (e.g., BM25 instead of $TF \times IDF$). Standard person name recognition techniques together with string similarities could also significantly improve results.

Vector space bases. Our experiments show that using lexical compounds as topics of expertise is not significantly better than using all terms in the documents (see table 3.2). We also see that applying LSA on the term space virtually kills the effectiveness of our system. One reason for this could be that we performed LSA on the already pruned version because of resource limitations. We also tried to prune the lexical compounds basis in the same way we pruned the term basis (see Section 3.4.3) and we observed that, contrary to the term basis, performance does not significantly change (p =0.8353) considering only the top k lexical compounds basis vectors ordered by document frequency.

[7] http://ir.nist.gov/w3c/contrib

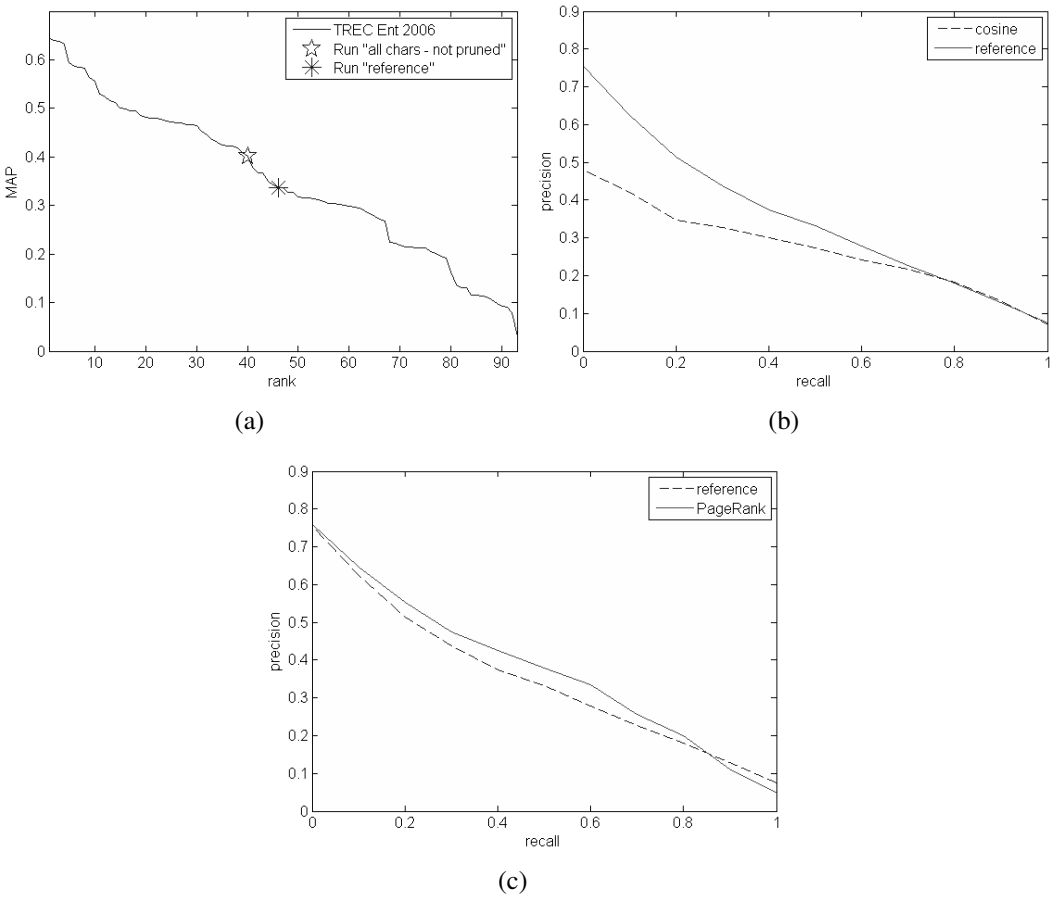


Figure 3.3 (a) Retrieval effectiveness (MAP values) of the 2006 participant at TRECent (b) Precision/Recall curves varying the vector similarity function (c) Precision/Recall curves using PageRank weights for the documents.

Pruning methods. Performing the experiments on the term based indexing of the collection, we see that performance is maximized when we consider both digits and letters and when we do not prune the basis considering only the most frequent terms (see table 3.3). As expected from document retrieval, removing terms decreases retrieval effectiveness while increasing indexing efficiency. In the reference run we used the smallest basis to save time.

Relationship weights. In these experiments we used different weights to represent the occurrence of a candidate's name or address in the author field or in the body of an email. For these experiments we considered only the emails; this is sound as using only the mailing list part of the W3C collection on the 2006 queries performance was not significantly different. We observed that when not considering candidate occurrences in the text (i.e., text weight is zero) effectiveness decreases. Including text occurrences with 10% of the weight of an author occurrence still yields performance lower than the reference run (which used 1/1 weights for author/text). Other combinations of author/text occurrence weights did not significantly change effectiveness compared to the reference run (see table 3.4). These re-

Table 3.2 Retrieval effectiveness (MAP and p-value) using different vector space dimensions.

Dimension	Term	LSA	LexComp	LexComp Pruned
MAP (p-value)	0.3370	0.0894 (p =0.0)	0.3586 (p =0.5927)	0.3625 (p =0.5374)

Table 3.3 Retrieval effectiveness (MAP and p-value) varying the pruning techniques.

	Pruned	Not Pruned
Only Letters	0.3370	0.3854 (p =0.0091)
All Chars	0.3716 (p =0.0112)	0.4024 (p =0.0035)

sults imply that candidate occurrences in the text are also very important and can not be ignored.

Table 3.4 Retrieval effectiveness (MAP and p-value) using different text weights.

Author/Text weights	1/0	1/0.1	1/0.25	1/0.5	1/0.75	1/1
MAP	0.2246	0.3149	0.3306	0.3378	0.3365	0.3370
p-value	0.0	0.0183	0.1559	0.6803	0.5528	1

Document dependent extensions. We experimented with the use of a document dependent feature in order to refine the candidates position in the vector space. The MAP values show that using the PageRank values of the documents as weights for the candidate placement slightly improves performance: MAP of 0.3435 with PageRank compared to 0.3370 without PageRank with a statistical significance of $p = 0.0515$. The difference at early precision values is very small, though, see figure 3.3(c).

Similarity functions. While it is possible to use cosine similarity, this does not take information about the vector lengths into account. Using *projection similarity* retrieval effectiveness improves substantially growing from a MAP of 0.2502 for cosine similarity to 0.3370 for projection similarity, with a statistically significant difference ($p = 0.0020$). Figure 3.3(b) also shows that, especially for early precision values, the improvement is substantial. These results confirm our intuition described in Section 3.4.2.

3.5 Discussion

In this chapter we addressed the ER problem and presented a general model for ranking entities. We described in detail a possible instantiation of the model and a set of techniques for the Expert Finding task. The experimental evaluation has shown that by combining our approaches we achieve an improvement of 35% in terms of MAP and of 53% in terms of P@10 over our baseline. The overall effectiveness is still rather low but this can be explained by the non optimal selection of features included in our relationship weights. Most important, the prototype application we developed shows the flexibility of the proposed ER model and how it is possible to apply it to any kind of scenario where documents about entities are available. Another possible application can be, for example, an entity search system on desktop collections.

As continuation of this work, based on the proposed model, we will design ER algorithms for the entire Web of Entities. The first step will be the identification of entities in Web pages. After this we will build entity representations which can be indexed as Web pages, allowing the end user to query for entities. Moreover, we will perform an effectiveness comparison between the developed system and other entity search systems.

As a first step toward this direction, in the next chapter we present methods for ranking entities in the Wikipedia setting. As done by the INEX Entity Ranking track, we will consider each Wikipedia page as an entity profile that systems can retrieve, which is obviously not the typical setting in other datasets, where one document refers to more than one entity, and one entity is mentioned in more than one document.

4 Retrieving Entities in Wikipedia

4.1 Introduction

4.1.1 Motivation

In the previous chapter we presented an ER model and its application to the Enterprise setting. We described a general model for ER and instantiated it to the problem of Expert Finding, that is, find people who are knowledgeable about the topic described in the user query.

We now focus on the ER task performed on the Web. Example user queries in this context may be "Countries where I can pay in Euro" or "Movies starring Harrison Ford" (see Figure 4.1). User needs may differ and refer to any entity type. For such reasons, new

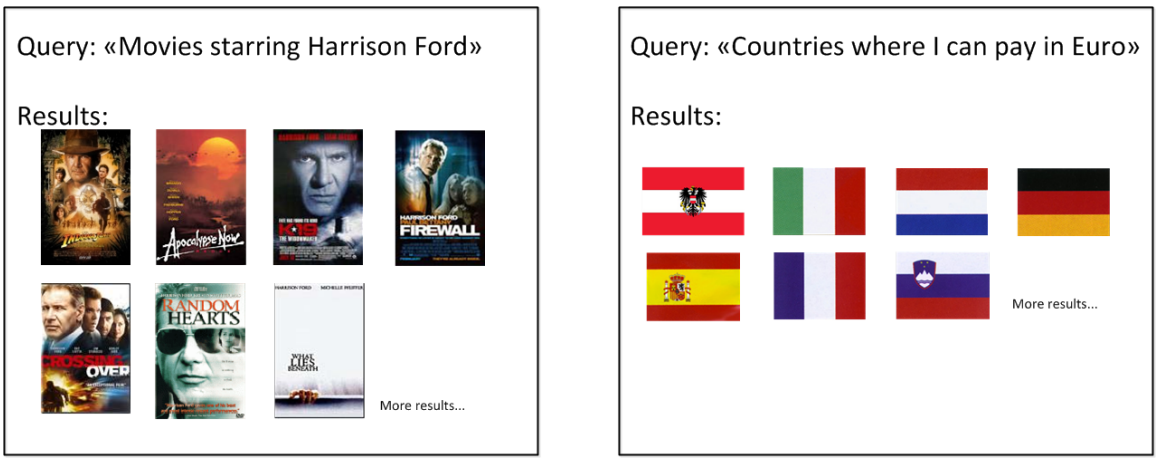


Figure 4.1 Examples of ER queries dealing with different entity types.

challenges arise when we look at such more difficult task as compared to Expert Finding (cfr. Section 3.1.2).

Wikipedia is the result of a collaborative effort of creating a open encyclopedia. Anyone on the Web can obtain write access to it and modify any of its articles. Over time, the quality and coverage of the Wikipedia content has increased. For such reasons, this is a very good document collection on top of which to perform the ER task. Queries like the examples used so far can be perfectly answered by searching for information in Wikipedia.

If we look at the Wikipedia setting, ER concerns triples of type <query, category, entity>. The category (i.e., the entity type), specifies the type of 'objects' to be retrieved. The query is a free text description that attempts to capture the information need. The Entity field specifies example instances of the entity type. Expert finding uses the semantic notion of 'people' as its category, where the query would specify 'expertise on T' for finding experts on topic T. While document retrieval and expert finding represent common information needs, and therefore would require specific technologies to be developed, the ER task on Wikipedia challenges researchers to develop generic ranking methods that apply to entities irrespective of their type: e.g., actors, restaurants, museums, countries, etc.

4.1.2 Contributions

In this chapter we focus on the ER task and we first propose a general model for finding entities of a given type, we relate it with the model proposed in the previous chapter, and show how this can be applied to different entity search scenarios. We generalize this search task and identify its main actors so that we can optimize solutions for different search contexts such as, for example, the Wikipedia corpus. We also present results for list completion task when starting from given entities. Building on top of the designed model, we developed search algorithms based on Link Analysis, Natural Language Processing (NLP), and Named Entity Recognition (NER) for finding entities in the Wikipedia corpus. Moreover, we experimentally evaluate the developed techniques using a standard testbed for ER. We show that these algorithms improve significantly over the baseline and that the proposed approaches – incorporating Link Analysis, NLP and NER methods – can be beneficially used for ER in Wikipedia. We evaluated our algorithms for Entity Retrieval only on Wikipedia as they are designed for this specific context and can not be directly applied to the Web at large. It will be a future step to extend the approach to the entire Web of Entities. We also perform an analysis of which ER queries are difficult and which are easy for state-of-the-art systems.

The main contributions of this chapter are thus a formal Entity Retrieval model along with a collection of tested methods designed for the Wikipedia scenario. Additionally, we present a per-topic analysis of the ER systems performance and an analysis of the different kinds of entity queries.

The chapter is structured as follows. In Section 4.2 we describe related work in the area of ER on Wikipedia. In Section 4.3 we describe our effort on creating test collections for evaluating effectiveness of ER systems on the Wikipedia. Then, we describe and evaluate our system for such task. In Section 4.4, we first define a model for ER (entities, queries, and the ER system) and compare it with the model proposed in the previous chapter. In

Section 4.5 we present an ER system we use in our experiments including a standard benchmark data set. In Sections 4.6 and 4.7 we depict all the different approaches we use and their effectiveness evaluations. Section 4.8 analyzes the queries and discusses the results. We finally conclude the chapter and present future improvements in Section 4.9.

4.2 Related Work

Finding entities on the Web is a recent topic in the IR field. The first proposed approaches [25, 39, 40, 41] mainly focus on scaling efficiently on Web dimension datasets but not on the effectiveness of search, as addressed in this chapter.

A formal model for entities has been presented in [132]. This entity representation is, similarly to our proposal, based on $(<attribute>, <value>)$ pairs and on a "Category of reference" that describes the entity type which can be taken from an ontology. In this chapter we propose a model for the entire ER process where the entity representation is just a sub-part. A framework for modelling the IR process has been presented in [143] where the authors present a matrix-based framework for modelling possible search tasks. The model we propose is focused on ER; it is less formal but more intuitive.

Approaches for finding entities have also been developed in the Wikipedia context. Previous approaches to rank entities in Wikipedia exploited the link structure between Wikipedia pages [135, 160] or its category structure [157] also by using graph based algorithms [142, 158], clustering approaches [3], or probabilistic models [19]. Compared to these methods, we start first designing a model for ER making the development of algorithms possible also in domains different from Wikipedia and we exploit semantic and NLP techniques to improve effectiveness. Our next step will be to apply the algorithms, evaluated on the Wikipedia corpus, on the entire Web, as done in [25, 39, 40, 41], aiming to find the best compromise between efficiency and effectiveness of search.

With respect to previous approaches we based our algorithms on a structured representation of entities at indexing level – we used a structured index built using NLP techniques. For this reason, relevant to our work are projects aiming at extracting and annotating entities and structure in Wikipedia. For example, versions of Wikipedia annotated with state of the art NLP tools are available [101, 145].

Another relevant work is [170] which also aims at retrieving entities in Wikipedia but without the assumption that an entity is represented by a Wikipedia page as done in INEX-XER. They rather annotate and retrieve any passage of a Wikipedia article that could represent an entity. Our structured index allows such kind of retrieval as well. A foundation for an effective ER can also be the automatic identification of instances and classes in the Wikipedia category hierarchy [174]. Knowing which categories describe instances can help the ER system in finding entities relevant to the query because not all the articles in Wikipedia are entity descriptions.

More recently, in the TREC 2009 Entity Track [15], the task of finding related entities given one entity as query (e.g., "Airlines that currently use Boeing 747 planes") was inves-

tigated. Results shown that the use of Wikipedia as evidence of relevance is very important to effectively solve this task [147, 103].

An important related area of research is entity identity on the Web. It is crucial for the ER task being able to uniquely and globally identify entities on the Web so that the Search Engine can return a list of identifiers to the user who can afterwords navigate in the entity descriptions. A strong discussion already started in the Web research community [31, 34] and solutions for entity identity resolution on the Web have been proposed [32]. Our solution for finding entities relies on these infrastructures able to globally identify entities on the Web.

With respect to our final analysis of easy and difficult topics, a related area is that of query difficulty prediction [36]. In particular, in [159] they study how to automatically predict the difficulty of an ER query in the Wikipedia context. They also study how to adapt their system variables accordingly in order to improve effectiveness. Our findings about what characterizes a difficult or easy topic are consistent with the features they use for classifying topics. For example, they use the number of articles attached to categories, the number of categories attached to the entities, query length, etc. Compared to this work we perform a more detailed analysis of which properties make a query difficult or not for systems to answer. On the other hand, we did not make our system adaptive to different topics even though we have shown how different techniques among the proposed ones work better for different topics.

4.3 Evaluation Testbed for Entity Retrieval

In this section we describe our effort in creating reusable test collections for evaluating and comparing ER systems in the Wikipedia setting. In the remaining of this chapter we will use such test collection for evaluating and comparing the designed search algorithms.

From 2007 to 2009, the Initiative for the Evaluation of XML retrieval (INEX) has organized a yearly XML Entity Ranking track (INEX-XER) to provide a forum where researchers may compare and evaluate techniques for engines that return lists of entities. I co-organized the INEX-XEX 2008 and 2009 editions. In entity ranking (XER) and entity list completion (LC), the goal is to evaluate how well systems can rank entities in response to a query; the set of entities to be ranked is assumed to be loosely defined by a generic category, implied in the query itself (for XER), or by some example entities (for LC). For evaluation purpose we adopted a stratified sampling strategy for creating the assessment pools, using xinfAP as the official evaluation metric [169]. In 2009 we adopted the new Wikipedia document collection containing annotations with the general goal of understanding how such semantic annotations can be exploited for improving Entity Retrieval.

In this section we describe the INEX-XER track running both the ER and the LC tasks, using manually defined topics over two different INEX Wikipedia collections. For evaluation purpose we adopted a stratified sampling strategy for creating the assessment pools, using xinfAP as the official evaluation metric [169].

The remainder of the section is organized as follows. In Section 4.3.1 we present details about the collections used in the track and the two different search tasks. In Section 4.3.2 we present an experimental comparison of different sampling strategies for creating assessment pools. In Sections 4.3.3 and 4.3.4 we summarize the evaluation results computed on the final set of topics for both the XER and LC tasks for INEX-XER 2008 and 2009 respectively. As in 2009 we used a selection of topics from the past INEX-XER campaigns, in Section 4.3.5 we provide an initial comparison of the new test collection with the previous ones. Finally, in Section 4.3.6, we discuss the results.

4.3.1 INEX-XER Setup

Data

The INEX-XER 2007 and 2008 track used the Wikipedia XML collection [75], where we exploit the category metadata about the pages to define the entity types. Participants are challenged to exploit fully Wikipedia's rich text, structure and link information.

Category information about the pages loosely defines the entity sets. The entities in such a set are assumed to loosely correspond to those Wikipedia pages that are labelled with this category (or perhaps a sub-category of the given category). Obviously, this is not perfect as many Wikipedia articles are assigned to categories in an inconsistent fashion. There are no strict restrictions in Wikipedia for category assignment, thus categories can represent the type of an entity as well as a label for it. Retrieval methods should handle the situation that the category assignments to Wikipedia pages are not always consistent, and also far from complete. The intended challenge for participants is therefore to exploit the rich information from text, structure, links and annotations to perform the Entity Retrieval tasks.

The INEX-XER 2009 track used the new Wikipedia 2009 XML data based on a dump of the Wikipedia taken on 8 October 2008 and annotated with the 2008-w40-2 version of YAGO [156], as described in [145]. The Wikipedia pages and links are annotated with concepts from the WordNet thesaurus. Additionally to the category information already available in the previous collection, such semantic annotations could be exploited to find relevant entities.

Tasks

The INEX-XER track consisted of two tasks, i.e., entity ranking (with categories), and entity list completion (with examples). Entity list completion is a special case of entity ranking where a few examples of relevant entities are provided instead of the category information as relevance feedback information.

Entity Ranking. The motivation for the entity ranking (XER) task is to return entities that satisfy a topic described in natural language text. Given preferred categories, rele-

vant entities are assumed to loosely correspond to those Wikipedia pages that are labelled with these preferred categories (or perhaps sub-categories of these preferred categories). Retrieval methods need to handle the situation where the category assignments to Wikipedia pages are not always consistent or complete. For example, given a preferred category 'art museums and galleries', an article about a particular museum such as the 'Van Gogh Museum' may not be labelled by 'art museums and galleries' at all, or, be labelled by a sub-category like 'art museums and galleries in the Netherlands'. Therefore, when searching for "art museums in Amsterdam", correct answers may belong to other categories close to this category in the Wikipedia category graph, or may not have been categorized at all by the Wikipedia contributors. The category 'art museums and galleries' is only an indication of what is expected, not a strict constraint.

List Completion. List completion (LC) is a sub-task of entity ranking which considers relevance feedback information. Instead of knowing the desired category (entity type), the topic specifies a number of correct entities (instances) together with the free-text context description. Results consist again of a list of entities (Wikipedia pages). If we provide the system with the topic text and a number of entity examples, the task of list completion refers to the problem of completing the partial list of answers. As an example, when ranking 'Countries' with topic text 'European countries where I can pay with Euros', and entity examples such as 'France', 'Germany', 'Spain', then 'Netherlands' would be a correct completion, but 'United Kingdom' would not.

Topics

At INEX-XER 2008, participants from eleven institutions have created a small number of (partial) entity lists with corresponding topic text. Candidate entities correspond to the names of articles that loosely belong to categories (for example may be subcategory) in the Wikipedia XML corpus. As a general guideline, the topic title should be type explanatory, i.e., a human assessor should be able to understand from the title what type of entities should be retrieved.

At INEX-XER 2009, based on the topics from the previous two INEX-XER years, we have set up a collection of 60 INEX-XER topics for both the XER and LC tasks, with 25 from 2007 and 35 topics form 2008. The <categories> part is provided exclusively for the Entity Ranking Task. The <entities> part is given to be used exclusively for the List Completion Task.

The INEX-XER 2008 Test Collection

The test collection created during INEX-XER 2008 consists of 35 topics and their assessments in an adapted trec_eval format (adding strata information) for the xinfAP evaluation script. Topics 101-149 are genuine INEX-XER topics, in that the participants created these topics specifically for the track, and (almost all) topics have been assessed by the original

topic authors. From the originally proposed topics, topics with less than 7 relevant entities and topics with more than 74 relevant entities have been excluded from the test collection (because they would be unstable or incomplete, respectively). Three more topics were dropped, one on request of the topic assessor and two due to unfinished assessments, resulting in a final INEX-XER 2008 test collection consisting of 35 topics with assessments. The judging pools have been created using a stratified sampling strategy (see Section 4.3.2). We used the following strata and sampling rates for the pool construction of INEX-XER 2008:

- $[1,8]$ 100%
- $[9,31]$ 70%
- $[32,100]$ 30%

where $[i,j]$ indicates the interval of retrieved results considered (i.e., from rank i to rank j) followed by the sampling rate for the interval.

The INEX-XER 2008 Test Collection is available for download at http://www.l3s.de/~demartini/XER08/.

The INEX-XER 2009 Test Collection

The test collection created during INEX-XER 2009 consists of 55 topics and their assessments on the new annotated Wikipedia collection. The judging pools for the topics have been based on all submitted runs, using a stratified sampling strategy [169]. As we aimed at performing relevance judgements on 60 topics (as compared to 49 in 2008), we adopted a less aggressive sampling strategy that would make the judging effort per topic lower. We used the following strata and sampling rates for the pool construction of INEX-XER 2009:

- $[1,8]$ 100%
- $[9,31]$ 70%
- $[32,50]$ 30%
- $[51,100]$ 10%

where $[i,j]$ indicates the interval of retrieved results considered (i.e., from rank i to rank j) followed by the sampling rate for the interval. The resulting pools contained on average 312 entities per topic (as compared to 400 in 2008 and 490 in 2007).

All 60 topics have been re-assessed by INEX-XER 2009 participants on the new collection. As in 2008, from the originally proposed ones, topics with less than 7 relevant entities (that is, 104, and 90) and topics with more than 74 relevant entities (that is, 78, 112, and 85) have been excluded [60]. The final set consists of 55 genuine XER topics with assessments.

Out of the 55 XER topics, 3 topics have been excluded for the LC task (i.e., 143, 126, and 132). The reason is that example entities for these topics were not relevant as the underlying Wikipedia collection has changed. After this selection, 52 List Completion topics are part of the final set and are considered in the evaluation.

The INEX-XER 2009 Test Collection is available for download at http://www.l3s.de/~demartini/XER09/.

Not-an-entity annotations

An additional difference from the relevance judgements performed during INEX-XER 2008 is the possibility for the assessor to mark a retrieved result as not being an entity. This choice is intended for those Wikipedia pages that do not represent an entity and, thus, would be irrelevant to any XER query. Examples include "list-of" or "disambiguation" pages.

Differentiating between a non-relevant and not-an-entity result does not influence the evaluation of INEX-XER systems as both judgements are considered a wrong result for XER tasks.

4.3.2 Investigation on Sampling strategies

In INEX-XER 2008, we decided to use sampling strategies for generating pools of documents for relevance assessments. The two main reasons for sampling are to reduce the judging effort and to include into the pools also documents from higher ranks.

The first aspect we have to analyse is how the comparative performances of systems is affected while we perform less relevance judgements. We used the 2007 INEX-XER collection simulating the situation of performing less relevance judgements. We compared three different sampling strategies, that is, a uniform random sampling from the top 50 documents retrieved by the IRSs, a sampling based on the relevance distribution among the different ranks, and a stratified sampling with strata manually defined by looking at the distribution of relevance from the previous year.

For the experimental comparison of the three different sampling approaches we used the 24 INEX-XER topics from the 2007 collection. As only data from 2007 could be used at the time of the study (to design the INEX-XER 2008 track setup), we used the leave-one-out approach for simulating the approach of learning from past data. That is, we considered all the topics but one as previous year data. In these topics the relevance assessments and, therefore, the relevance distribution over the ranks is known. The relevance distribution computed on all the topics but one is used for generating a random sample (based on such probabilities) of documents from the runs. The systems' ranking on the remaining topic (the one left out and therefore not used for learning) is then computed and compared with the original ranking. This process is iterated over all topics and the average correlation

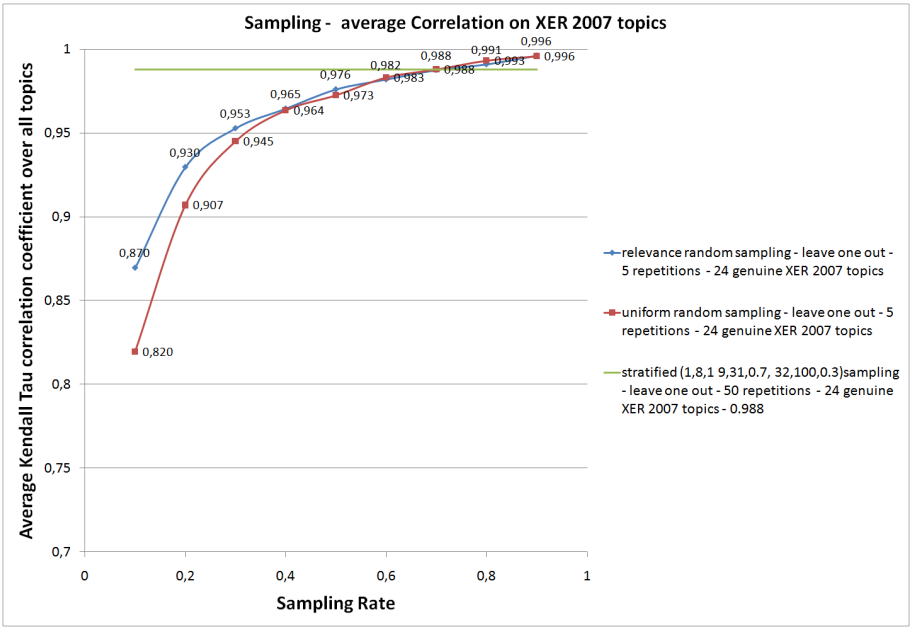


Figure 4.2 Correlation results between the original system ranking and the ranking derived using the strata-based sampling strategy.

value[1] is taken.

Uniform Random Sampling

The first approach we decided to investigate is a Uniform Random Sampling of retrieved documents which would allow to compute metrics such as infAP [168]. In order to do so, we first randomly selected some ranks at which to take documents from the runs. Then, we considered only the assessments on those documents for ranking the systems, assuming that the other entities were not judged (and therefore not relevant). Finally, we measured the correlation with the original system ranking using the 24 XER topics from 2007. Figure 4.2 presents the result. We conclude that the desirable high correlation is feasible as long as sufficiently many assessments are made.

Relevance based Random Sampling

In order to perform a sampling with a higher chance of selecting relevant documents into the pools, we used the distribution of relevance over ranks and learned from the 2007 data the probability of finding a relevant entity at each rank (up to 50 as the depth of 2007 pool) in the runs. We then sampled the 2008 runs using such probability distribution. The relevance distribution in 2007 for ranks up to 100 is displayed in Figure 4.3.

[1] We used Kendall's τ as measure for ranking correlation.

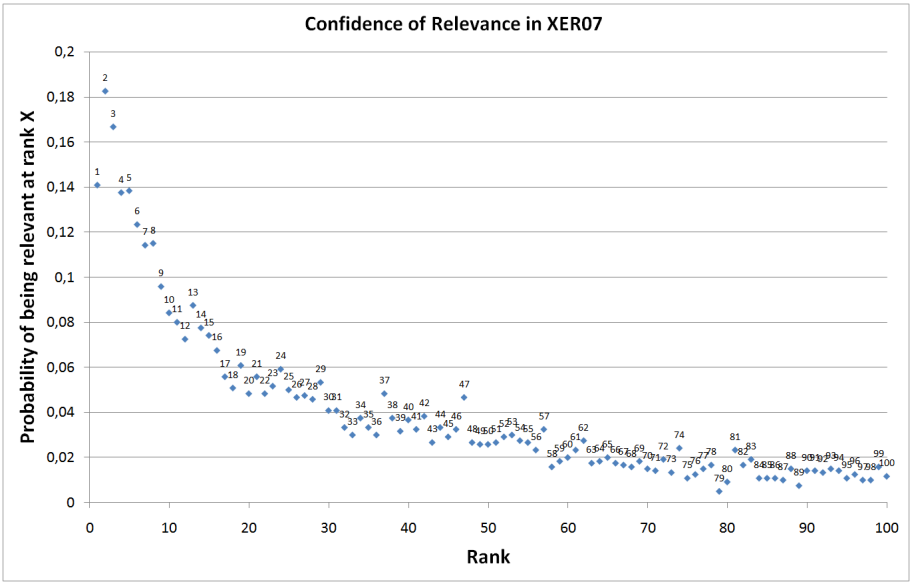


Figure 4.3 Distribution of relevance over rank of top 100 retrieved results in INEX-XER 2007

Figure 4.2 shows the correlation between the original system ranking and the ranking derived from either the relevance based or the uniform random sampling.

Stratified Sampling

A third option to performing a sampling in order to construct pools for relevance assessment is the stratified approach, which aims at including in the pools a big number of relevant results. The idea is to perform sampling within each stratum independently of the other. In this way it is possible to sample more documents in higher strata and less from strata which are down in the ranking. Using stratified sampling allows to compute *xinfAP* [169] as evaluation metric, which is a better estimate of AP in the case of incomplete assessments. There is then the need to optimally selecting the strata and the sampling percentage for each strata, which is an open problem.

Considering the results shown in Figure 4.3, we decided to use the following strata for the pool construction of INEX-XER 2008:

- $[1,8]$ 100%
- $[9,31]$ 70%
- $[32,100]$ 30%

This means that we include in the pool 45 documents from each run. In order to compare this approach with the ones presented above, we computed the correlation using the strata-based sampling strategy. The result is presented in Figure 4.2.

Stratified sampling with the selected parameters performs, in terms of IRS ranking correlation, as well as uniform and relevance based sampling at 70%. The two 70% sampling approaches make each run contribute 35 documents to the pool while the stratified approach, by going down to rank 100 in the runs, make each run contribute 45 documents. Given that we used the 2007 collection for the experimental comparison we should notice that relevance assessments have been performed up to rank 50. Therefore, several documents ranked from 51 to 100 may not have been assessed; so they are considered not relevant in the experiments, even if they could be. If we want to fairly compare the judgement effort of the three sampling approaches we have to count the number of documents the stratified sampling approach make the runs contribute up to rank 50, which corresponds to 30 documents. In other words, stratified sampling gives a slightly lower judging effort than the uniform random sampling and the relevance based sampling for the same correlation in IRS rankings.

4.3.3 INEX-XER 2008 Results

At INEX-XER 2008 six groups submitted 33 runs. The pools have been based on all submitted runs, using the stratified sampling strategy detailed in Section 4.3.1. The resulting pools contained on average 400 entities per topic. The evaluation results for the XER task are presented in Table 4.1, those for the LC task in Table 4.2, both reporting xinfAP. In the LC task, the example entities provided in the topics are considered not relevant as the system is supposed not to retrieve them. The runs which name ends with ".fixed" have been corrected by the organizers removing the example entities present in the topics.

Most participants used language modelling techniques as underlining infrastructure to build their Entity Retrieval engines. For both the XER and the LC task the best performing approach uses topic difficulty prediction by means of a four-class classification step [159]. They use features based on the INEX topics definition and on the Wikipedia document collection obtaining 24% improvement over the second best LC approach. Experimental investigation showed that Wikipedia categories helped for easy topics and the link structure helped most for difficult topics. As also shown in INEX-XER 2007, using score propagation techniques provided by PF/Tijah works in the context of ER [141]. The third best performing approach uses categories and links in Wikipedia [102]. They exploit distances between document categories and target categories as well as the link structure for propagating relevance information showing how category information leads to the biggest improvements.

For the LC tasks the same techniques performed well. Additionally, [102] also used relevance feedback techniques using example entities. In [99] they adapted language models created for expert finding to the LC task incorporating category information in the language model also trying to understand category terms in the query text.

As for the use of the stratified sampling techniques we performed an analysis of possible bias due to the order in which IRS have been considered when constructing the pool. A potential drawback of the stratified sampling approach is that the order in which runs are

Table 4.1 Evaluation results for ER runs at INEX-XER 2008.

Run	xinfAP
1_FMIT_ER_TC_nopred-cat-baseline-a1-b8:	0.341
1_cirquid_ER_TEC_idg.trec:	0.326
4_UAms_ER_TC_cats:	0.317
2_UAms_ER_TC_catlinksprop:	0.314
1_UAms_ER_TC_catlinks:	0.311
3_cirquid_ER_TEC.trec:	0.277
2_cirquid_ER_TC_idg.trec:	0.274
2_500_L3S08_ER_TDC:	0.265
1_L3S08_ER_TC_mandatoryRun:	0.256
3_UAms_ER_TC_overlap:	0.253
1_CSIR_ER_TC_mandatoryRun:	0.236
4_cirquid_ER_TC.trec:	0.235
4_UAms_ER_TC_cat-exp:	0.232
1_UAms_ER_TC_mixture:	0.222
3_UAms_ER_TC_base:	0.159
6_UAms_ER_T_baseline:	0.111

considered for contributing to the pool could influence the strata information in the evaluation process, as we select the strata of the entity according to the run in which it was first encountered. Clearly, the order of the runs does not influence the number of entities contributed from each run in the pool, as Figure 4.4 shows that each run contributes an equal number of entities to the pool set. This conclusion is not valid anymore if we consider the run contribution to each stratum. As in the pooling process we considered an entity being part of the stratum where it was first encountered, the first runs considered had a bigger contribution as for the strata information. It is possible to see such bias in Figure 4.5 where run `1_FMIT_ER_TC_nopred-cat-baseline-a1-b8` has the most prominent presence in each stratum as it was the first run considered while creating the pool. Further research will have to show if this materializes into an effect on the system's ranking.

4.3.4 INEX-XER 2009 Results

Five groups submitted 32 runs to the INEX-XER 2009 track. The evaluation results for the XER task are presented in Table 4.3, those for the LC task in Table 4.4, both reporting xinfAP [169]. We can see that best effectiveness values are higher than 0.5 xinfAP which shows improvement over past years. It must be noticed that high effectiveness values may be due to the adoption of topics already used in past campaigns.

Looking at the approaches taken by track participants, it is possible to notice that a common behaviour at INEX-XER 2009 was to identify entity mentions in the text of Wikipedia articles, passages, or queries. They then applied different techniques (e.g., detect entity

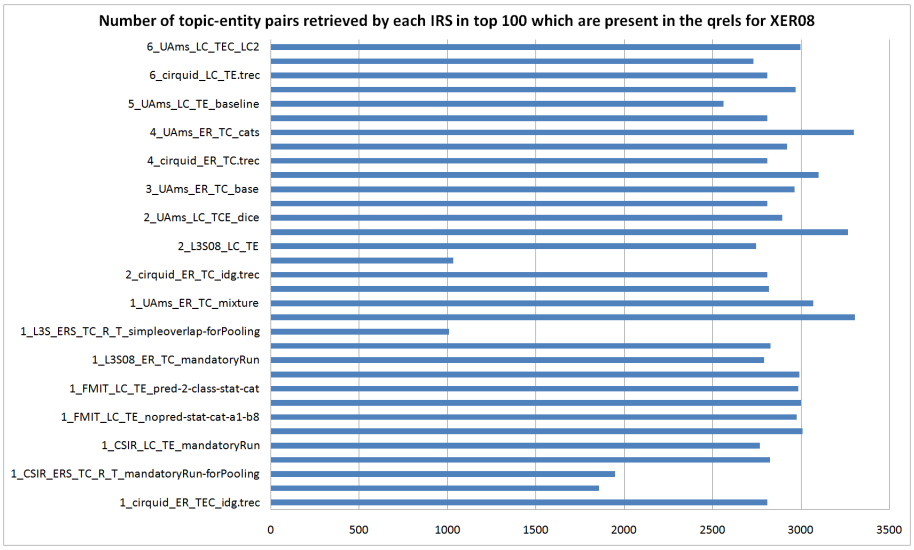


Figure 4.4 Number of retrieved entities which are present in the pool for INEX-XER 2008 submitted runs.

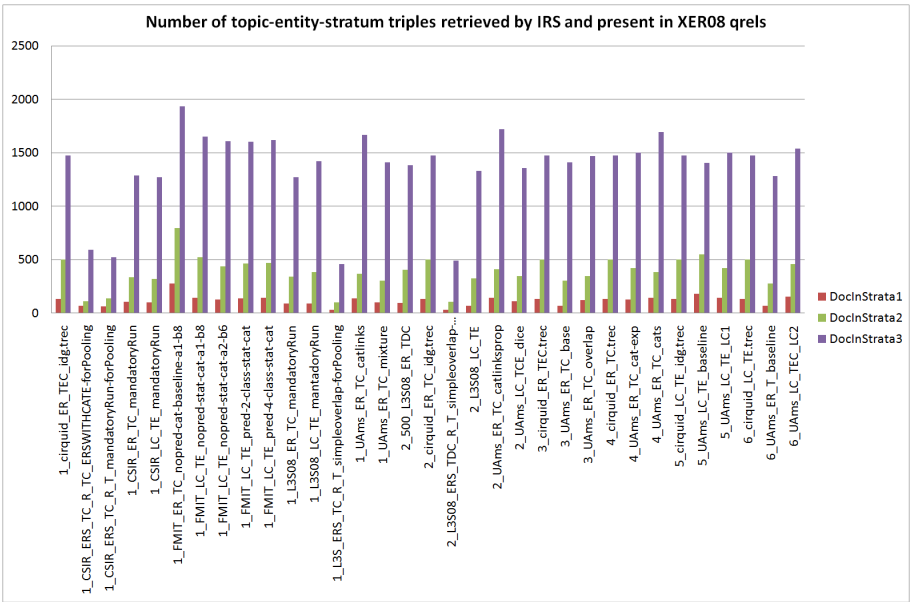


Figure 4.5 Number of retrieved entities which are present in each stratum of the pool for INEX-XER 2008 submitted runs.

Table 4.2 Evaluation results for LC runs at INEX-XER 2008.

Run	xinfAP
1_FMIT_LC_TE_nopred-stat-cat-a1-b8:	0.402
1_FMIT_LC_TE_pred-2-class-stat-cat:	0.382
1_FMIT_LC_TE_nopred-stat-cat-a2-b6:	0.363
1_FMIT_LC_TE_pred-4-class-stat-cat:	0.353
5_UAms_LC_TE_LC1:	0.325
6_UAms_LC_TEC_LC2:	0.323
1_CSIR_fixed:	0.322
2_UAms_LC_TCE_dice:	0.319
5_cirquid_LC_TE_idg.trec.fixed:	0.305
1_L3S08_LC_TE_mantadoryRun:	0.288
2_L3S08_LC_TE:	0.286
5_cirquid_LC_TE_idg.trec:	0.274
6_cirquid_LC_TE.trec.fixed:	0.272
1_CSIR_LC_TE_mandatoryRun:	0.257
6_cirquid_LC_TE.trec:	0.249
5_UAms_LC_TE_baseline:	0.133

relations, exploit category information) to produce a ranked list of Wikipedia articles that represents the retrieved entities. The best performing approach exploited a probabilistic framework ranking entities using similarity between probability distributions.

As we considered all the runs during the pooling phase and as some groups submitted more runs than others, we performed an analysis of possible bias in the pool. Figure 4.6 shows the *pool coverage* (i.e., the number of entities retrieved by the run which are present in the pool and, therefore, have been judged) as compared with the total number of retrieved entities. Figure 4.7 shows the *pool unique contribution* (the number of entities in the pool which were sampled only in this run) for each run submitted to INEX-XER 2009. The runs having worse coverage from the pool are also those that contribute most unique entities. This means that such runs are "different" from others in the sense that they retrieve different entities. Interestingly, runs from the AU-CEG group contributed a big number of unique relevant entities. We can see in Figure 4.8 that a relatively high proportion of retrieved entities belong to strata 1 and 2, which guarantees a fair evaluation. However, as some runs did not retrieve up to 8 results for a topic and as some systems did not run all the topics, not all runs have an equal number of entities covered in stratum 1 (which considers a complete sampling). For the Waterloo runs for example, only few entities have been sampled due to the low number of entities retrieved per topic (see also Figure 4.6).

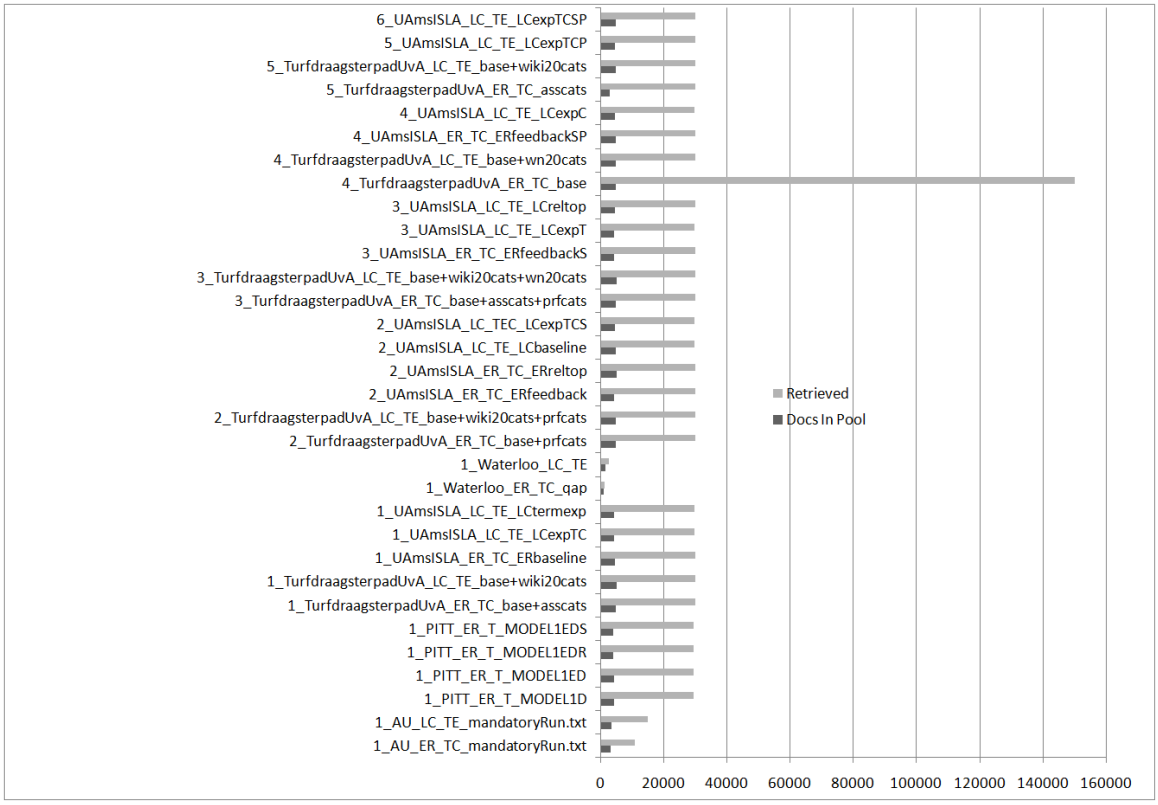


Figure 4.6 Pool coverage: number of entities retrieved by the runs and present in the pool.

Table 4.3 Evaluation results for ER runs at INEX-XER 2009.

Run	xinfAP
2_UAmsISLA_ER_TC_ERreltop:	0.517
4_UAmsISLA_ER_TC_ERfeedbackSP:	0.505
1_AU_ER_TC_mandatoryRun.txt:	0.270
3_UAmsISLA_ER_TC_ERfeedbackS:	0.209
2_UAmsISLA_ER_TC_ERfeedback:	0.209
1_TurfdraagsterpadUvA_ER_TC_base+asscats:	0.201
3_TurfdraagsterpadUvA_ER_TC_base+asscats+prfcats:	0.199
2_TurfdraagsterpadUvA_ER_TC_base+prfcats:	0.190
1_UAmsISLA_ER_TC_ERbaseline:	0.189
4_TurfdraagsterpadUvA_ER_TC_base:	0.171
1_PITT_ER_T_MODEL1EDS:	0.153
1_PITT_ER_T_MODEL1EDR:	0.146
1_PITT_ER_T_MODEL1ED:	0.130
1_PITT_ER_T_MODEL1D:	0.129
1_Waterloo_ER_TC_qap:	0.095
5_TurfdraagsterpadUvA_ER_TC_asscats:	0.082

4.3.5 Comparison of INEX-XER Collections

At INEX-XER 2009 we used a selected set of topics from previous years while using the newer and annotated Wikipedia collection. This allows us to perform some comparisons with previous collections.

Comparison on the number of relevant entities. Figure 4.9 shows the number of entities judged relevant for each topic at INEX-XER 2009 as well as in the previous years. While we would expect to find the same number of relevant entities while re-judging the same topic, we must take into account that the new Wikipedia is bigger and contains more up-to-date information. Thus, we expect the number of relevant entities to be greater or equal to that in the past year. This is not the case for 12 topics out of 60. The highest difference can be seen for topic 106 "Noble English person from the Hundred Years' War". By a manual inspection of the judgements on this topic, we observed a high disagreement between assessors in the two different campaigns.

Preliminary Comparison on Samples and Judgements. Based on the titles of the sampled pages we compared the pool and assessments against the previous years. As the 2007 and 2008 topics have been assessed on the old smaller corpus, we made a comparison between the two datasets based on the entity title performing a simple textual comparison.

Table 4.4 Evaluation results for LC runs at INEX-XER 2009.

Run	xinfAP
5_UAmsISLA_LC_TE_LCexpTCP:	0.520
3_UAmsISLA_LC_TE_LCreltop:	0.504
6_UAmsISLA_LC_TE_LCexpTCSP:	0.503
1_UAmsISLA_LC_TE_LCexpTC:	0.402
1_UAmsISLA_LC_TE_LCtermexp:	0.358
2_UAmsISLA_LC_TEC_LCexpTCS:	0.351
3_UAmsISLA_LC_TE_LCexpT:	0.320
1_AU_LC_TE_mandatoryRun.txt:	0.308
2_UAmsISLA_LC_TE_LCbaseline:	0.254
4_UAmsISLA_LC_TE_LCexpC:	0.205
4_TurfdraagsterpadUvA_LC_TE_base+wn20cats:	0.173
3_TurfdraagsterpadUvA_LC_TE_base+wiki20cats+wn20cats:	0.165
2_TurfdraagsterpadUvA_LC_TE_base+wiki20cats+prfcats:	0.160
5_TurfdraagsterpadUvA_LC_TE_base+wiki20cats:	0.157
1_TurfdraagsterpadUvA_LC_TE_base+wiki20cats:	0.156
1_Waterloo_LC_TE:	0.100

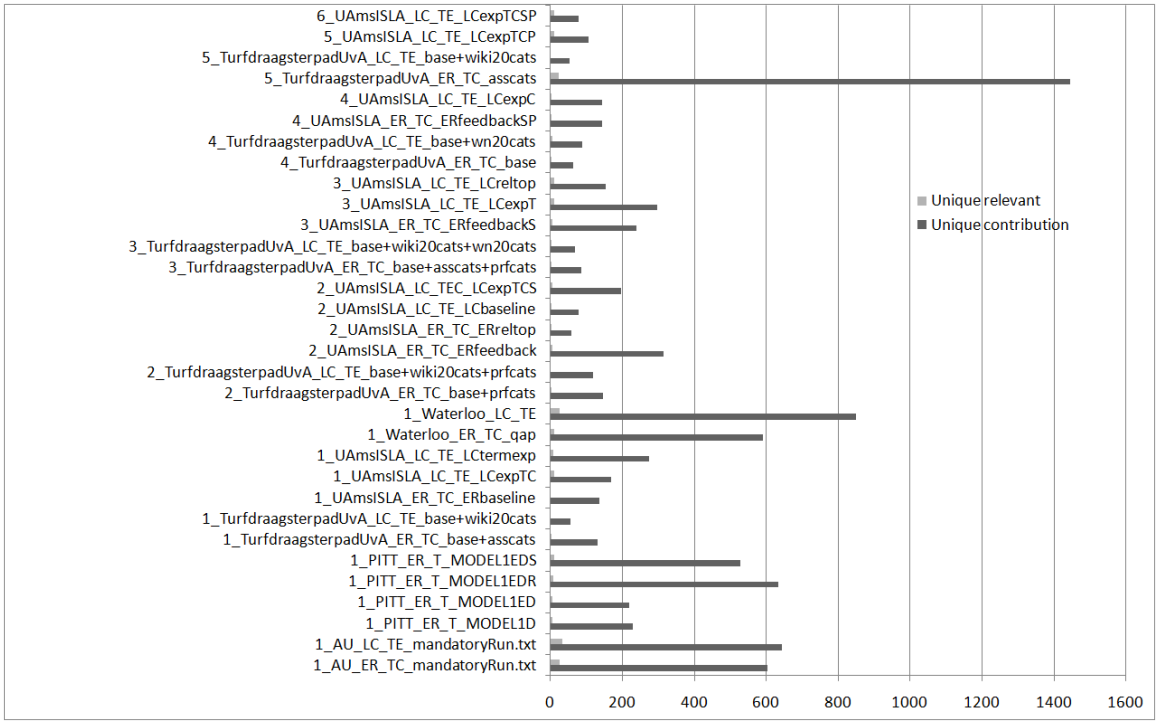


Figure 4.7 Pool Unique Contribution: number of (relevant) entities sampled only in this run.

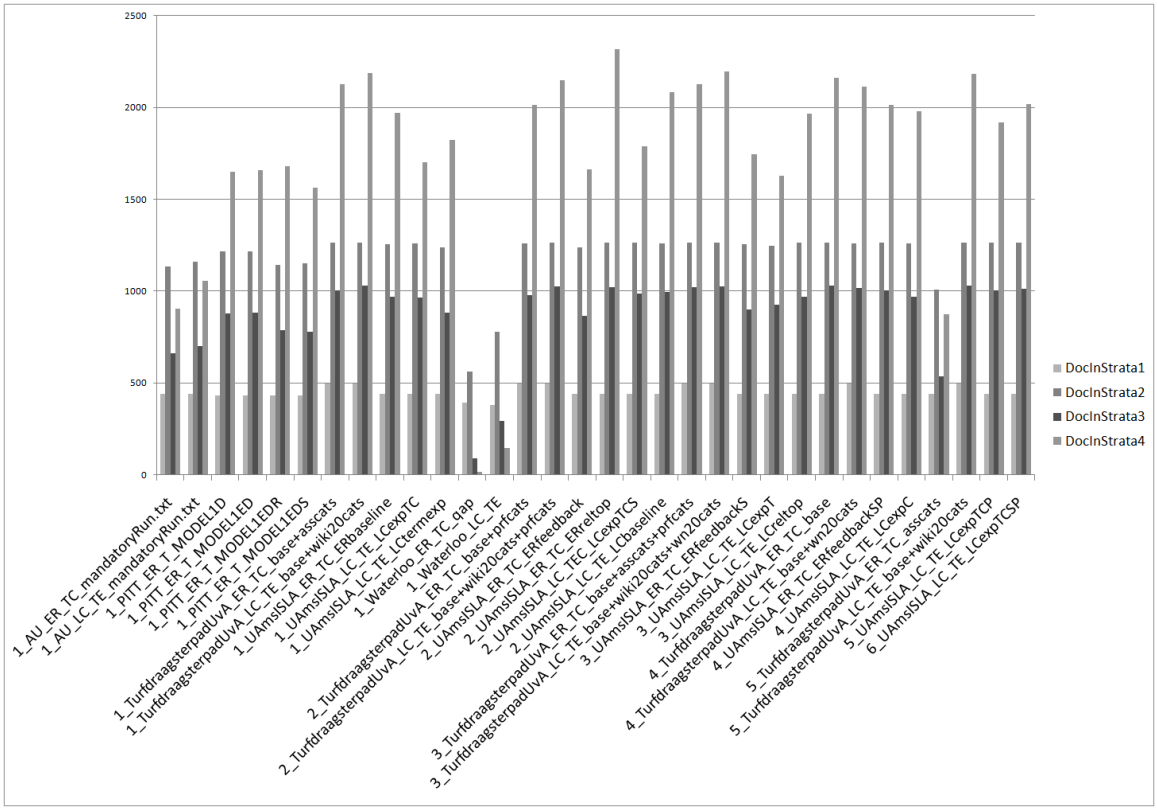


Figure 4.8 Per-stratum pool coverage: number of entities retrieved by runs in different strata and present in the pool.

Thus minor changes in the title of an entity in the two collections would lead to the entity not being identified as the same in the two collections. Table 4.5 shows the comparison results for the final set of 55 topics assessed in 2007/2008 and 2009. We show the following indicators in the table (average values per topic):

- S-co: the number of entities that have been sampled in both campaigns, based on entity title comparison;
- S-past: the total number of entities that have been sampled in the past campaign;
- S-2009: the total number of entities that have been sampled at INEX-XER 2009;
- R-past: the total number of relevant entities in the past campaign;
- R-2009: the total number of relevant entities at INEX-XER 2009;
- R-co: the number of entities assessed as relevant in both campaigns;
- I-co: the number of entities assessed as not-relevant in both campaigns;

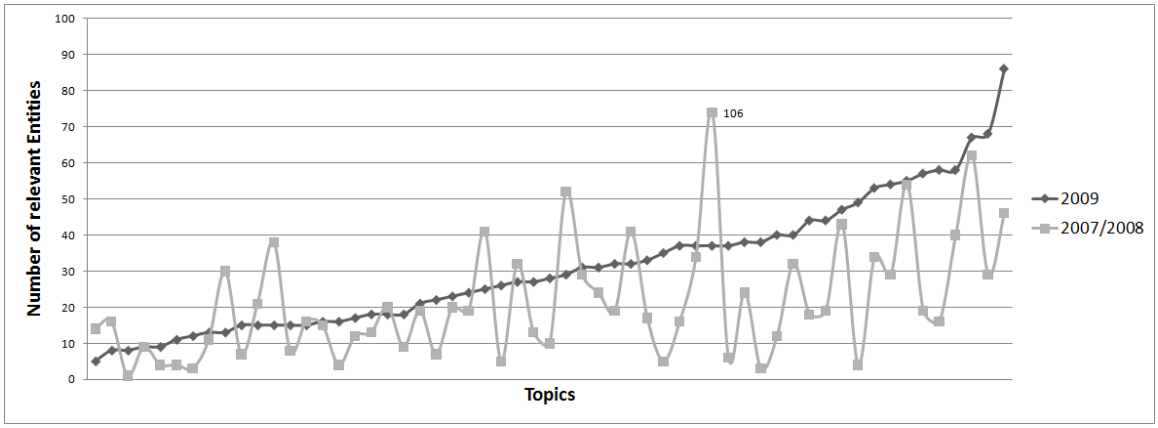


Figure 4.9 Number of relevant entities per topic compared to previous years.

- Ryear1-Iyear2: the number of entities assessed as relevant in year1 and as not-relevant in year2;
- UniRelYear: the number of entities that were both sampled and assessed as relevant only in the respective year.

Table 4.5 Comparison of samples and judgements between INEX-XER 2009 and previous years.

	Indicator	2007	2008
common	S-co	57.86	79.24
	S-past	490	400.03
	S-2009	295.27	314.55
	R-past	16.36	26.09
	R-2009	26.18	31.64
agreement	R-co	10	16.91
	I-co	41.86	54
disagreement	Rpast-I09	3.86	4.94
	R09-Ipast	2.55	3.24
	UniRelPast	2.5	4.24
	UniRel09	13.64	11.48

For the set of 22 topics assessed both in 2007 and 2009, from the entities sampled in both years (S-co), 17% were relevant in both years, and 72% were not relevant (the agreement between the assessments being of 89%). On the other hand, 6.7% entities were relevant in 2007 and assessed as not relevant in 2009, and 4.4% the other way around, thus amounting to a disagreement of 11%. Additionally, on average 2.5 entities relevant in 2007 have not been sampled in 2009 (UniRelPast), and 13.64 entities not sampled in 2007 have been sampled and are relevant in 2009 (UniRel09).

For the set of 33 topics assessed both in 2008 and 2009, from the entities sampled in both years, 21% were relevant in both years, and 68% were not relevant (the agreement between the assessments being of 89%). On the other hand, 6.2% entities were relevant in 2008 and assessed as not relevant in 2009, and 4.1% the other way around, thus amounting to a disagreement of 10%. Additionally, on average 4.2 entities relevant in 2008 have not been sampled in 2009 (UniRelPast), and 11 entities not sampled in 2008 have been sampled and are relevant in 2009 (UniRel09).

In conclusion, we observe that for both sets of topics, the agreement between assessments (R-co + I-co) is much larger than the disagreement (Rpast-I09 + R09-Ipast).

We have not explicitly forbidden the use of previous years assessments for training and tuning 2009 systems, and indeed, a group (i.e., ISLA) had used them. For this reason we analysed the effect of these runs on the pool. We removed from the pool the results uniquely contributed by these runs and analysed how this affects the number of common relevant results between two corresponding years. As it can be seen in Table 4.6 the main effect of the system using previous years assessments for training was in the sampling of less non-relevant entities. The sampling of relevant entities (R-co) was affected only a little.

Table 4.6 Analysis of usage of previous assessments for training and tuning in INEX-XER 2009.

2007-2009			
	S-co	R-co	I-Co
all runs	57.86	10	41.86
runs - ISLA	53.77	9.73	36.18
2008-2009			
	S-co	R-co	I-Co
all runs	79.24	16.91	54
runs - ISLA	73.79	16.64	44.45

4.3.6 Discussion

After the first edition of INEX-XER Track at INEX 2007 [164], the 2008 edition [60] created additional evaluation material for IR systems that retrieve entities instead of documents. INEX-XER 2008 created a set of 35 XER topics with relevance assessments for both the XER and LC tasks.

The INEX-XER 2009 track created additional evaluation material for IR systems that retrieve entities instead of documents. The new aspect in INEX-XER 2009 is the use of the new annotated Wikipedia collection, re-using topics developed for the two previous years of this track. The track created a set of 55 XER topics with relevance assessments for the XER task and 52 for the LC task.

The 2009 edition has been the last for INEX-XER which has built, over three years, a

set of 55 XER topics with relevance assessments for two different document collections. Moreover, we have observed, over three campaigns, an improvement in term of effectiveness and more advanced techniques being used by Entity Retrieval systems participating to the track.

4.4 A Formal Model for Entity Retrieval

The problem of ranking entities can be split in several steps. First, the user's information need has to be translated into a query which is sent to an ER system. Then, the query has to be interpreted and the *entity need* has to be extracted. The Search Engine has to understand what type of entity the user is searching for and what properties the retrieved entities should have. In the next step, relevant results are retrieved. The results have to be retrieved according to the entity description which can include different properties, e.g., the type. We propose in this section a model for the entire ER process to help with the understanding of the problem and of the general ER flow. This model can be instantiated in a number of different contexts such as, for example, Wikipedia.

4.4.1 Users' Information Need

We want to model a user searching for entities. We assume that a user has an information need, that is, she wants to find a list of entities that satisfy some properties. It is a user task to create, starting from the information need, a query, either using keywords or natural language questions, that can be processed by the system. The user query will describe the set of properties that an entity should satisfy for being relevant. For example, a query might indicate the type of entities to be retrieved (e.g., "cars") and distinctive features (e.g., "German", "hybrid"). A real world example is given by the Search Engine sindice.com where the user can issue queries like "Washington class:person" specifying the type of results she wants to get. A query q is defined, as a list of $(<attribute>, <value>)$ pairs. Thus, $q = \{(a_1, v_1) \ldots (a_n, v_n)\}$.

4.4.2 Entities

The central part of the model is the set of entities. An entity e^i is something that has separate and distinct existence and objective or conceptual reality[2]. An entity is represented by its unique identifier, and by a set of properties described as $(<attribute>, <value>)$ pairs (see Figure 4.10). The properties of an entity can include, for example, its name or its type. Moreover, it is important to notice that relations can be present between entities. It is possible to model these relations as other properties using $(<attribute>, <value>)$ pairs

[2]Clearly, it is not easy to define what an entity is and there are many studies in philosophy trying to define it. To keep the problem simple we consider retrievable entities all *objects/things* about which the Web contains information.

where the value would be the target entity of the relation. This representation of relations is consistent with previous work on entity relation search [173].

We can now define the entity description $d(e^i) = \{ID^i, P^i\}$ for the entity e^i as composed of an entity identifier $ID^i = id(e^i)$ and a set of properties $P^i = \{(a_1^i, v_1^i) \ldots (a_n^i, v_n^i)\}$ of the type $(<attribute>, <value>)$ pairs. For example, the soccer player "Alexandre Pato" could have as ID the automatically generated unique identifier $ap12dH5a$ and properties such as $(born\_in, 1989)$ or relations with other entities such as $(playing\_with, acm15hDJ)$ where $acm15hDJ$ is the ID of the soccer club "A.C. Milan".

4.4.3 Data Sources

In order to create the entity descriptions $d(e^i)$ (see Section 4.4.2) we need to extract data about entities from several sources. For example, for describing the company "Microsoft Corporation" we might want to harvest the Web in order to find all the facts and opinions about this entity. For this reason, we call *data sources* the provenance of the information we collect in an entity description. We define a data source s_j as any passage of a digital document. This can be an XML element, a paragraph of an e-mail, a blog post on the Web, etc. Each data source s_j can be about one or more entities. The aggregation of all the data sources about the same entity e^i (noted as $\bigcup s_j^i$) will create the properties part P^i of the entity description $d(e^i)$ as defined in Section 4.4.2. This would define inferring the description of an entity as: $\bigcup_j s_j^i \Longrightarrow P^i$. The relations between entities are also inferred from the data sources and are part of P^i.

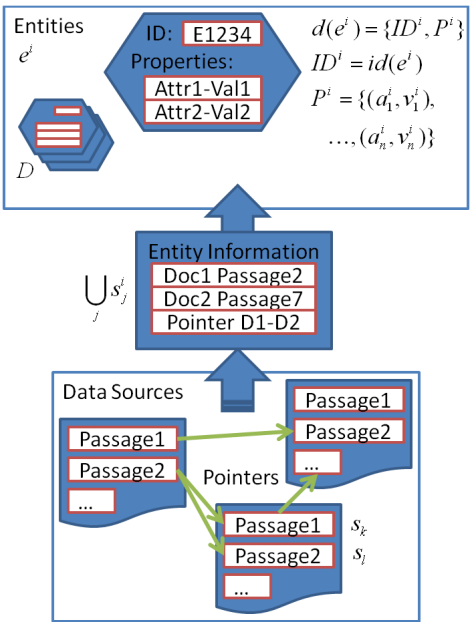


Figure 4.10 Entities and their extraction from different data sources.

4.4.4 Entity Retrieval System

At this point, a collection of entity descriptions $D = \{d(e^1) \dots d(e^n)\}$ and a user query q is available. An Entity Retrieval System (ERS) will now take as input these two elements and will return a ranked list of entities $E = \{e^i \dots e^j\}$ (Figure 4.11 shows a sketch of the flow inside the ERS). In order to do this, an ERS will hard-code a *scoring function* $\phi(q, d(e^i))$ that returns a score (i.e., a real number) for a given user query q and entity description $d(e^i)$. This score represents the confidence or probability of the entity e^i of being relevant to the query q. In this way the ERS will be able to rank the entire set of entities according to the confidence of being relevant to the user query. Of course, the scoring function can take into account several evidences of relevance such as the comparison between properties in q and properties in $d(e^i)$, the popularity value of the entities in the collection (e.g., PageRank), or give more importance to a particular property (e.g., the type of entities to be returned).

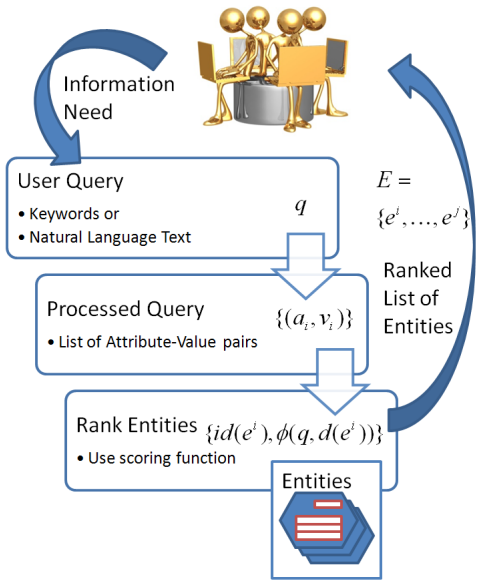


Figure 4.11 The Entity Retrieval flow.

4.4.5 Application Scenarios for ER

As an initialization step, it is necessary to assign a global identifier for each entity in the collection. Attempts to generate global unique identifiers are already underway, e.g., the OKKAM[3] European Integrated Project is dealing with ID generation on the Web. One simple application scenario would be ranking consumer products (i.e., entities) where a customer provides as query a list of constraints (e.g., brand, color, size, etc.). ER can be also performed on the Web, where the definition of an entity is not as trivial as in the enterprise example. The entity description will then contain attributes of the entities mentioned in

[3] http://fp7.okkam.org/

sentences of several Web pages referring to the entity. Relations between entities can then be constructed from links between Web pages as well as references between sentences or paragraphs.

Another application scenario which keeps the main information as in the Web application scenario but also adds some structure is the Wikipedia model for ER. In this case we consider in D any entity e^i that has its own page in Wikipedia. With this assumption we can easily see these pages as the entity description $d(e^i)$ and the set of the Wikipedia pages that describe an entity as the collection D. Of course, in Wikipedia there are pages which do not describe a particular entity as, for example, the "List of ..." pages. The challenge is to identify which are not entity pages and discard them from D. For each entity the $(<attribute>,<value>)$ pairs can be build, for example, out of the info-boxes of the Wikipedia pages which contain factual information about the described entity (for example, articles about people contain information about name, birth date, birth place, etc.). In the Wikipedia scenario the sources of information are the people and each s^i_j contributing to $d(e^i)$ can be reconstructed from the edit history of each page allowing also to associate trust values in order to weight more particular sources (see also [4] about such computation). For defining the *type* property in $d(e^i)$ the Wikipedia category information can be used. Relations between entities can be discovered analysing the Wikipedia internal links between pages. The query can be built by the user providing some keywords describing interesting properties plus the selection of a Wikipedia category in order to provide information about the type of entities which are requested. The ranking function $\phi(q,d(e^i))$ should use both information about the properties and the type in order to produce the best ranking.

The specific Wikipedia scenario is slightly different from the general Web scenario as Wikipedia is more clearly structured. It is easy to define an entity as having its own Wikipedia page (i.e., each Wiki page is about one entity) – in the general Web scenario we would have to segment Web pages to extract only sections related to the entity and discard other parts like advertisements or navigational headers. Moreover, it is also easy to extract the entity type from a Wikipedia page, as one of the entity attributes $d(e^i)$, by just considering the Wikipedia categories the page belongs to – the Web scenario would require a thorough NLP processing of the text in order to find phrases describing the entity (e.g., "Mexico *is a* country"). We also make use of the YAGO ontology (see Section 4.6.1) which is built from Wikipedia and WordNet. If the same system architecture were to be applied to the Web, a new ontology would have to be built in order to make the results comparable. YAGO is also being used in other scenarios than Wikipedia: Revyu.com [93] uses Yago class definition in order to assign types to the objects of reviews; in [138] the authors use links between DBpedia and YAGO for interlinking singers, songs, music events, etc. in music datasets. Finally, there is much more content to be found on the Web, while Wikipedia only focuses on some, more common, topics and entities (e.g., we can not find members of a particular organization only from Wikipedia). Nevertheless, Wikipedia is a very good starting point for the emerging task of Entity Retrieval, and we will focus on the Wikipedia scenario in the remainder of the chapter. Other algorithms might be developed

for ER on the Web still following the proposed model.

4.4.6 Comparison with the Vector Space based Model for ER

If we compare this model with that proposed in Section 3.3 we can see several differences. First, entities are represented in different ways. While the Vector Space-based model considers entities as vectors, that is, a linear combination of the space dimensions, the model proposed in this chapter defines entity profiles that consists of attribute-value pairs instead of simple a bag-of-words. In this way we can better model different types of entities with different properties while the first proposed model better fits the setting of expert finding where only one type of entity is considered and only one type of entity property is described (i.e., expertise). Additionally, as documents may refer to different entities, in this new model we consider document passages as sources for entity profiles instead of full documents as in the previous model.

4.5 A Baseline System for Finding Entities in Wikipedia

In this section we describe the INEX benchmark and the Wikipedia corpus we used for evaluating our ER algorithms and present a baseline system for Entity Retrieval in Wikipedia.

4.5.1 INEX and the Wikipedia Collection

As described in Section 4.3, in 2007, the evaluation initiative INEX[4] has started the XML Entity Ranking track (INEX-XER) to provide a forum where researchers may compare and evaluate techniques for systems that retrieve lists of entities [164]. In the INEX 2008 initiative a snapshot of the English Wikipedia [75] was used as evaluation corpus. The assumptions are that each entity described in Wikipedia is a possible candidate and the respective page represent the real entity to be retrieved. Additionally, the relevance is defined as binary.

Entity Retrieval can be characterized as 'typed search'. In the specific case of the INEX-XER track, categories assigned to Wikipedia articles are used to define the *entity type* of the results to be retrieved. Topics are composed of a set of keywords, the entity type(s), and, for the list completion task, a set of relevant entity examples. The two search tasks evaluated in the context of INEX-XER are entity ranking (XER) and entity list completion (LC). We will focus on entity ranking where the goal is to evaluate how well systems can rank entities in response to a query; the set of entities to be ranked is assumed to be loosely defined by generic categories, given in the query itself, or by some example entities respectively. In the entity list completion task, the categories from the queries are not used, but a set of

[4] http://www.inex.otago.ac.nz/

Table 4.7 INEX-XER 2008 Entity Ranking Topic example

Topic ID	#109
Title	National capitals situated on islands
Description	I want a list of national capitals that are situated on islands.
Narrative	Each answer should be the article about a nation-level capital whose geographic area consists of one or several islands.
Category	(10481) capitals
Examples	London, Tokyo, Jakarta

example entities provided for each topic. Thus from the example entities the system has to learn relevant information to describe the retrieved entities, such as category information and link information. For completeness, we will also present results on the LC task in order to show how our system can be adapted to the alternative LC setting.

The document collection used for evaluating our approaches is the Wikipedia XML Corpus based on an XML-ified version of the English Wikipedia in early 2006 [75]. The considered Wikipedia collection contains 659,338 Wikipedia articles. On average an article contains 161 XML nodes, where the average depth of a node in the XML tree of the document is 6.72. The original Wiki syntax has been converted into XML, using general tags of the layout structure (like *article, subsection, paragraph, title, list,* and *item*), typographical tags (like *bold, emphasized*), and frequently occurring link-tags. For details see Denoyer and Gallinari [75].

The official evaluation measure used at INEX-XER 2007 was Mean Average Precision (MAP) aiming at evaluating the overall entity ranking produced by the systems. In 2008, a new evaluation measure was introduced. INEX-XER 2008 used as official metrics xInfAP [169]. The metrics xInfAP is an estimation of Average Precision (AP) for the case where the judgement pool has been built with a stratified sampling approach. This means that the complete collection of documents is divided into disjoint contiguous subsets (strata) and then documents are randomly selected (sampling) from each stratum for relevance judgement. In this case it is possible to give more importance to documents retrieved higher by ER systems (e.g., by having a complete assessment of top 20 retrieved results) still going down into the list of retrieved entities (e.g., by having a partial assessment of results retrieved between rank 30-100). The metrics xInfAP is computed exploiting (similarly to infAP [168]) the estimation of Precision at each relevant documents in each stratum. In the following, we report values for xInfAP as it was used as official INEX-XER 2008 evaluation metrics and we use the INEX-XER 2008 testbed as it consists ER topics created for this purpose and assessed by participants. Even if it was shown to be redundant [165], we additionally report Precision at 10 retrieved entities (P@10) as it may be more intuitive for the reader to understand how well the system performs. The official set contains 35 ER designed topics. An example topic can be seen in Table 4.7.

Although this task seems easy given the Wikipedia corpus, we have to retrieve results

matching the sought type (i.e., the type resulting from the entire information need, not necessarily the assigned Wikipedia category). Relevant results for the example given in Table 4.7 ("National capitals situated on islands") would thus be: "London", "Tokyo", or "Jakarta", all of these being capitals situated on islands. Irrelevant results, although they still contain some information related to the Topic, would be the tourist attraction "London Eye" (which is situated in "London"), or even "List of capitals" (the page listing known capitals).

4.5.2 Processing Pipeline

The processing pipeline is presented in Figure 4.12. The first step is the creation of the inverted index from the XML Wikipedia document collection. We use standard retrieval techniques as initial step in order to focus on the ER-specific aspects of the approach that can improve the effectiveness of the system.

Starting from the raw structured XML documents, we created an inverted index using TFxIDF weighting scheme and cosine similarity as ranking function.[5] We indexed separately each article element (e.g., *title*, *text*, *textstem*, *categories*, *links*, etc.) so that we can perform the search on the different fields of a Wikipedia entity. The main advantage of such an approach is that we can search with different queries the content of the article and the category information. We can also exclude the title from the search as it usually does not contain discriminative information for ER queries. Important is also the anchor text of outgoing links in a page which usually describes related entities. For example, the Wikipedia page of Albert Einstein links to the page "Nobel Prize in Physics" using this same string as anchor text. By looking at this anchor text we can infer that the entity "Nobel Prize in Physics" is related to "Albert Einstein" which can help us in answering queries like, e.g., "Nobel prize winners".

After the creation of the index, the system can process the INEX-XER 2008 topics. Different approaches are adopted for building queries out of INEX Topics. Four modules are used interchangeably or complementary as main resources for our algorithms (see Figure 4.12):

1. *Wikipedia Taxonomy* is used for getting Wiki Category Links (see Section 4.6.2),
2. *Entity Linkage Information* is needed for exploring outgoing Links of Wikipedia entities (see Section 4.6.2),
3. the *NLP Processing* is used to find lexical compounds, synonyms and related words, named entities and filter information in the query (see Section 4.7), and
4. the *YAGO* ontology is used as underlying knowledge base (see Section 4.6.1).

[5] We used the Lucene tool with default configuration. http://lucene.apache.org

The INEX Topic is processed using these modules in order to create a disjunctive query starting from Title information, along with the specified Category from the Topic. For the XER task we only use the Title and Category information as the Description part of the topic contains complete sentences and it is realistic to assume that users would not post such long queries to a Search Engine. Moreover, the Narrative part of the topic is intended only as assessment guidelines. For the LC task we use the Title and the Example information from the topic. In this situation the desired category is learned from the given example entities. After the generation of the query, the index can be queried and a ranked list of retrieved entities is generated as output.

In all our experiments we use standard stemming and stopword removal techniques on the Wikipedia article content, while, when searching the category information, we do not apply a stemming algorithm in order to have a perfect match on the category information. Other combinations with regard to stopwords and stemming did not prove so effective. Moreover, we remove all results which have a title starting with "List of" as we do not expect such articles to be relevant entities to any ER query.

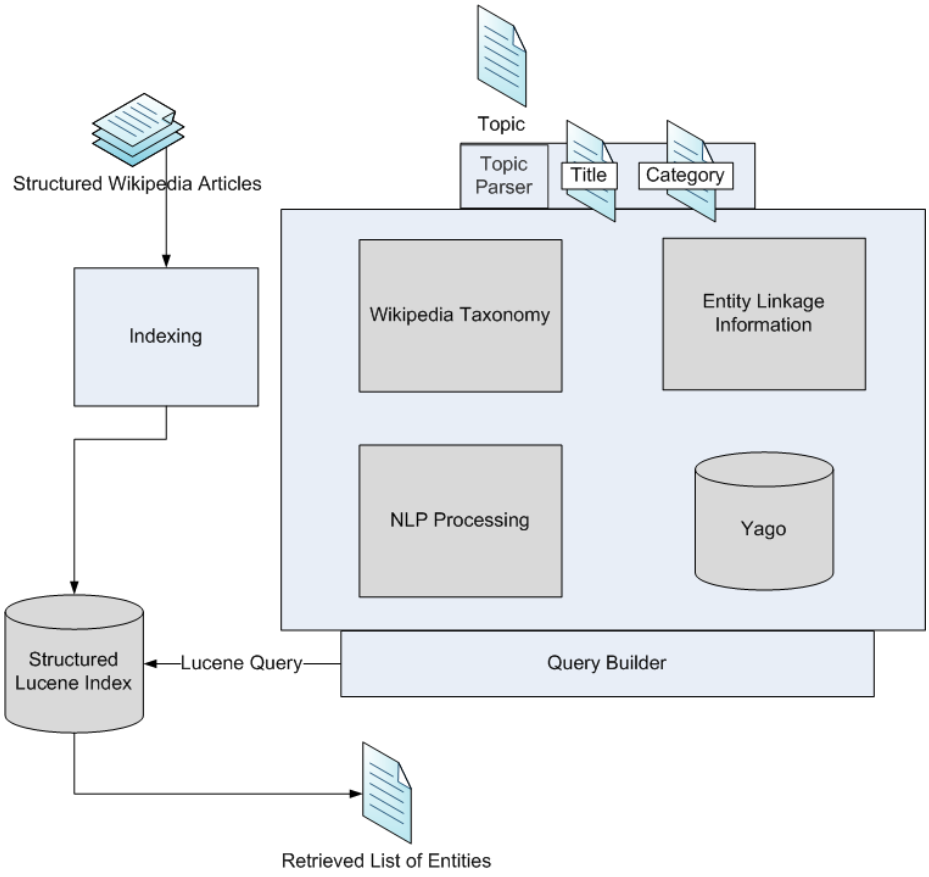


Figure 4.12 Processing Pipeline for the Entity Retrieval System.

Baseline Approach and Notations

We use the following notations for describing the algorithms presented throughout this article:

- $W^T = \{w_1^T, \ldots, w_n^T\}$ – the words in the given Topic Title;
- $W^C = \{w_1^C, \ldots, w_n^C\}$ – the words in the given Topic Category;
- $W^{LC} = \{w_1^{LC}, \ldots, w_n^{LC}\}$ – the words in the categories from the given Example Entities in the Topic.
- $W_{Adj}^T = \{w_{Adj_1}^T, \ldots, w_{Adj_n}^T\}$ – the adjectives in the Topic Title;
- $W_{Noun}^T = \{w_{Noun_1}^T, \ldots, w_{Noun_n}^T\}$ – the nouns in the Topic Title;

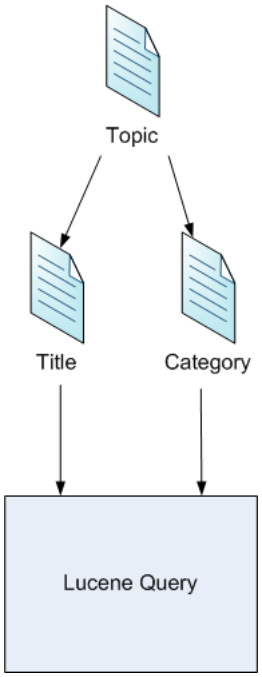


Figure 4.13 Query creation using only topic information.

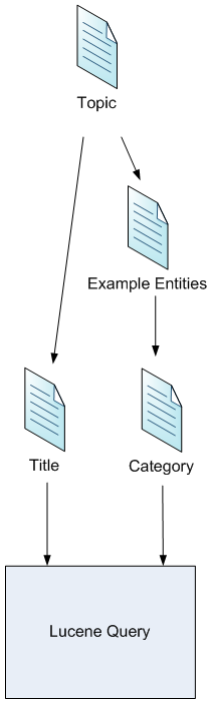


Figure 4.14 Processing Pipeline for the List Completion System.

As a baseline approach for constructing the query, we consider only the information given in the title part of the topic, as presented in Figure 4.13. For search we use the Vector Space Model and ranking is done using standard cosine similarity and TFxIDF weighting scheme[6]. We construct a disjunctive query containing both textual and contextual information (i.e., keywords and category information). For the textual part of the query we consider

[6] All search and ranking settings were left as default in Lucene.

the keywords from the title of the topic which we run against *TextStem* field (which contains the main textual area of a Wikipedia page, with stemmed terms) in the index. In the contextual part of the query we consider the category information from the topic which we run against the *Categories* field (containing the Wikipedia categories as listed on the Wikipedia pages).

The query part searched in the Wiki page text will thus contain following terms:

$$w_i \in W^T$$

For example, for the topic described in Table 4.7, the query resulting after stopword removal and stemming is the following:

text:(nation capit situat island)
category:(capitals)

In the case of the List Completion task we extract the categories from the given example entities, as presented in Figure 4.14. Each topic has between three to five example entities and we consider the categories that at least two example entities belong to, where applicable. For the topics where there are no categories with 2 common entities we move the threshold down. For example, for the topic in Table 4.7, the three example entities belong to the following categories:

- Categories with 2 common entities – capitals in asia, host cities of the summer olympic games, coastal cities;
- Categories with 1 entity – capitals in europe, cities in england, london, kanto region, tokyo, destroyed cities, harbours, visitor attraction in london cities in indonesia, provinces of indonesia, london squares, jakarta, london parks and commons.

As there are categories with two common example entities we use only those and the resulting LC query is the following:

text:(nation capit situat island)
category:(capitals in asia, host cities summer olympic games, coastal cities)

In the following sections we present two groups of approaches for improving effectiveness of ER in the Wikipedia context. In Section 4.6 we describe how ER effectiveness can be improved by extending the Wikipedia category information with external knowledge. In Section 4.7 we present approaches for refining the user query using IE and NLP techniques.

4.6 Structure Based Techniques for Entity Retrieval

One of the main issues in performing the ER task on Wikipedia is the incompleteness of the Wikipedia category structure. Relevant results may belong to different categories than the

ones provided by the user. In this section we first define algorithms aiming at improving the category information available by means of a highly accurate ontology build on top of WordNet and Wikipedia (Section 4.6.1). After this, we focus on how to better understand the user provided keyword query (Section 4.7) as a different approach for improving the effectiveness of the ER task [66].

4.6.1 Category Refinement by Means of a Highly Accurate Ontology

The lack and the imprecision of category information in Wikipedia can be attenuated by using Ontologies to identify relevant categories.

The YAGO Ontology YAGO[156][7] is a large and extensible ontology that builds on entities and relations from Wikipedia. Facts in YAGO have been automatically extracted from Wikipedia and unified with semantics from WordNet[8], achieving an accuracy of around 95%. All objects (e.g., cities, people, even URLs) are represented as entities in the YAGO model. The hierarchy of classes starts with the Wikipedia categories containing a page and relies on WordNet's well-defined taxonomy of homonyms to establish further *subClassOf* relations. We make use of these *subClassOf* relations in YAGO, which provide us with semantic concepts describing Wikipedia entities. For example, knowing from Wikipedia that *Married... with Children* is in the category *Sitcoms*, we reason using YAGO's WordNet knowledge that it is of the type *Situation Comedy*, same as *BBC Television Sitcoms*, *Latino Sitcoms*, *Sitcoms in Canada*, and 8 more.

We have implemented two approaches for Entity Retrieval in Wikipedia. Both approaches extend the traditional IR vector space model, enriching it with semantic information. Additionally to textual information from Wikipedia articles we also keep context information (i.e., category information) either extracted from Wikipedia or inferred using YAGO. The examples in the following sections are based on the topic described in Table 4.7.

Category Expansion While the category information which is present in the topic should contain most of or all the retrievable entities, this is for many topics not the case. Wikipedia is constructed manually by different contributors, so that the category assignments are not always consistent. Many categories are very similar and in some of these cases the difference is very subtle so that similar entities are sometimes placed in different categories by different contributors (e.g., hybrid powered automobiles are, inconsistently, either in the "hybrid vehicles" or the "hybrid cars" category and very seldom they are in both).

In the previous approach the category given in the topic was used to make the query retrieve entities from within that category. The method described here constructs an additional list of categories closely linked to the ones given in the topic description. This

[7] Available for download at http://www.mpi-inf.mpg.de/yago-naga/yago/downloads.html
[8] http://wordnet.princeton.edu/

Table 4.8 Category expansion for Topic #109: "National capitals situated on islands". Children are the result of filters on the subcategories, which may or may not remove terms from the subcategories.

Categories	capitals
Subcategories	capitals europe, capitals asia, capitals north america, capitals oceania, english county towns, capitals south america, historical capitals, capitals africa
Children	capitals europe, capitals asia, capitals north america, capitals oceania, english county towns, capitals south america, historical capitals, capitals africa
Siblings	cantonal capitals switzerland, capitals africa, capitals asia, capitals central america, capitals europe, capitals north america, capitals oceania, capitals south america, capitals caribbean, former us state capitals, etc.

extended list of categories is then used instead of the topic categories in query construction. The simplest starting point would be using merely Wikipedia subcategories looking at the Wikipedia categories hierarchy. Apart from this, we use three different types of category expansion, *Subcategories*, *Children* and *Siblings*.

Wikipedia itself has a hierarchical structure of categories. For each category we are presented with a list of *Subcategories*. This list of Subcategories is taken as-is and added to the query. For example, some of the subcategories for the "Actors" category are: "Animal actors", "Child actors", "Actors with dwarfism", "Fictional actors". More in detail, for this approach and the selected topic (see Table 4.7), the query would have additional *subcategories* as presented in Tables 4.8 and 4.9 added to the category search.

The *Children* list of categories is created by starting from the *Subcategories* list and filtering inappropriate ones out. It is more effective not to include all the Wikipedia subcategories in our *Children* list as some of them are not real subcategories, that is, they are not of the same type. As subcategories for a country, it is possible to have categories about presidents, movie stars, or other important persons for that country. This means that although we have as a starting category a country we end up having people as subcategories, which is not what we want in the entity retrieval context. The solution to this is selecting only those subcategories having the same class as the initial category. As described in Section 4.6.1, YAGO contains also class information about categories. We make use of this *subClassOf* information to identify suitable categories of the same type. Thus, a Wikipedia subcategory is included in the *Children* list only if the intersection between its ancestor classes and the ancestor classes in YAGO (excluding top classes like *entity*) of the initial category is not empty. The final list of *Children* will therefore contain only subcategories of the same type as the category given in the topic. Figure 4.15 presents an example of the *Children* list of the category "Sitcoms". For the selected topic (see Table 4.7), due to the fact that all the *Children* categories have the same type as the topic category, none of them

Table 4.9 Category expansion for Topic #124: "Novels that won the Booker Prize". Children are the result of filters on the subcategories, which may or may not remove terms from the subcategories.

Categories	novels
Subcategories	novels by author, **novel sequences**, novels by genre, **novelists**, novels by country, graphic novels, novels by year, novels based on computer and video games, modern library 100 best novels, **warcraft books**, first novels, light novels, autobiographical novels, r.o.d, sequel novels, forgotten realms novels
Children	novels by author, novels by genre, novels by country, graphic novels, novels by year, novels based on computer and video games, modern library 100 best novels, first novels, light novels, autobiographical novels, r.o.d, sequel novels, forgotten realms novels
Siblings	1902 novels, 1922 novels, 1947 novels, 1965 novels, 1975 novels, 1994 novels, agatha christie novels, alistair maclean novels, alternate history novels, american novels, anglo-welsh novels, anne mccaffrey novels, anthony trollope novels, arthur hailey novels, asian saga novels, australian novels, autobiographical novels, bret easton ellis novels, british novels, etc.

are filtered and the query looks the same as for the *Subcategories* approach (see Table 4.8). Table 4.9 presents an example where the subcategories in bold are filtered.

Using YAGO we can also retrieve categories of the same type as the starting category, not restricting just to the Wikipedia subcategories. We first determine the type of the starting category using the *subClassOf* relation in YAGO. Knowing this type we construct a list of all the categories of the same type and add them to the *Siblings* set. *Siblings* are, thus, all the categories of the exact same type as the initial category. Figure 4.16 shows how, starting from the category "Sitcoms", a list of *Siblings* is created.

Figure 4.17 depicts the inclusion of *Children* and *Siblings* in the query creation process. Constructing the query is done similarly to the naïve approach setting. The difference relies in the category matching part. In the naïve approach we had only the categories given within the topic while in this case we have the additional three lists of *Subcategories*, *Children* and *Siblings*. For the selected topic (see Table 4.7) the query would be extended with 23 *Sibling* categories, a part of which is shown in Table 4.8.

The resulting expanded list of categories is then matched against the *categories* field of the index. These extensions allows to find relevant entities with category information (e.g., "conifers" using the *Subcategory* or *Children* approach) different from the one which is present in the topic (e.g., "trees").

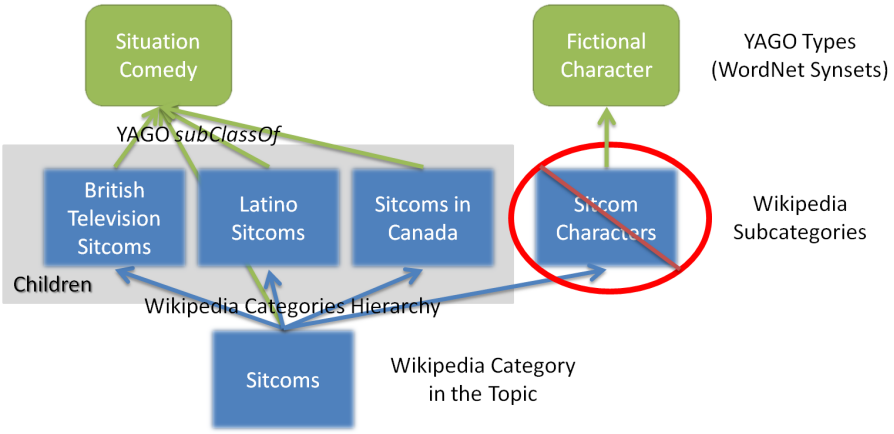


Figure 4.15 Example of *Children* identification starting from the "Sitcoms" category.

4.6.2 Using Wikipedia Links

Wikipedia, just like the Web, is highly interconnected. Search Engines make use of link information for traditional IR document ranking. Wikipedia pages, where each page represents an entity, has external links pointing to pages outside the Wikipedia corpus and internal links, which point to other Wikipedia entities. While external links are usually presented in a separate list at the end of the entity description, internal Wikipedia links appear inside the text. While indexing the entity pages, we have kept in the indexed text the names of the linked entities where they appear, and we have also indexed their titles in a separate field called *WikiLinks* to ensure that their importance is not lost in the entity text. In addition to the naïve approach, the contextual part of the query is searched also in the *WikiLinks* index field.

For example, for the query in Table 4.7 where *London, Tokyo* are relevant results, some of the entities that *London* links to are *Port, Capital City*, whereas *Tokyo* links to *Capital of Japan debate, Izu Islands, Ogasawara Islands* among others. There are many terms present in the list of linked entities, but, as the information in the linked entities field is more condensed than in the text field, linked entities can be a valuable source to improve the ranking of the search results.

4.6.3 Experimental Results

We performed evaluation experiments of our system using the 35 entity topics from the INEX 2008 Entity Ranking Track (see Section 4.5.1). We used the approaches presented in this section and combination of those, with the same notations as used previously, and some additional notations introduced here. Thus, a query is of the form:

$$q = \{(field_i, terms_j)\}$$

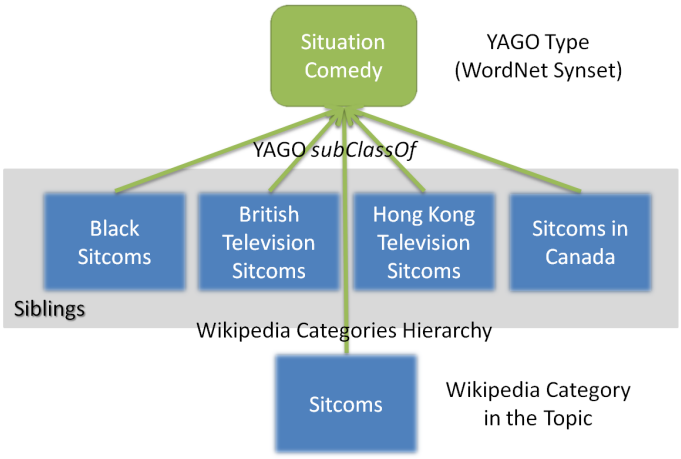


Figure 4.16 Example of *Siblings* identification starting from the "Sitcoms" category.

where $field_i$ is one of the fields in the Lucene index:

- *text* – the Wikipedia page text;
- *category* – Wiki categories of the pages;
- *outLinks* – outgoing links of the Wiki pages;

and $terms_j$ is a list of terms which should be searched in $field_i$:

- W^X – a list of words given in the Topic;
- $Sub(X)$ – extract the *subcategories* for the list of words X (e.g., $Sub(W^C)$);
- $Ch(X)$ – extract the *children* for X;
- $Sib(X)$ – extract the *siblings* for X;

We can combine terms from different approaches: e.g., $q = \{text, W^T \cup W^C\}, \{category, W^C\}$ would use the Category from the Topic and search this in the Wiki page text together with the Topic Title. Additionally the Topic Category is searched in the Wikipedia categories.

We evaluated our approaches against the *naïve approach* presented in Section 4.5.2 which has an xInfAP value of 0.2350 and a Precision for the first 10 results (P@10) of 0.306. Table 4.10 shows the first 10 results for Topic #109 ("National capitals situated on islands") using the *naïve approach*. We also show whether the result was assessed and if so whether it was considered relevant. As can be seen, not all relevant entities were assessed as relevant (e.g., capital of Solomon Islands is Honiara).

We performed the experiments on the evaluation dataset and we compared the algorithms which use category information and link information. The results (presented in

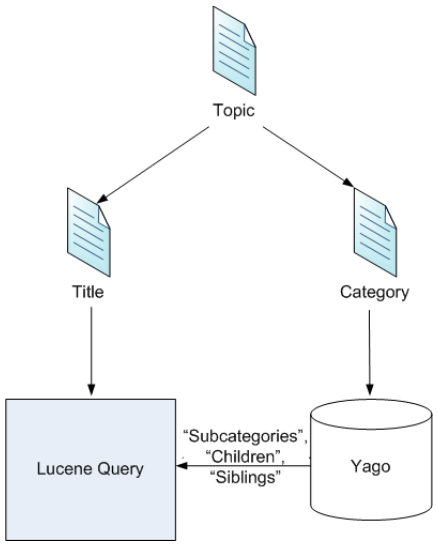


Figure 4.17 Query creation using category expansion techniques.

Table 4.11) show that using extra category information more than just the one present in the Topic does not improve the effectiveness, as the additional categories introduced contain too much noise on average. From the three category expansion techniques, the best performing approach, in terms of xInfAP, is obtained using the Subcategories. What we observed is that the use of Siblings gave even worse performance as the Siblings categories were greater in number than the Subcategories. We also included an approach where no category information is searched. Although overall results are worse than the baseline, results improved for topics where the categories are assigned inconsistently in Wikipedia (see Section 4.8 for an analysis).

Another observation we made is that the YAGO ontology is up-to-date and does not match all of the categories present in the XML Wikipedia dataset, which is a crawl of 2005. Thus the evaluation assessments might not consider relevant information which is present today in YAGO.

We also used the internal link structure within Wikipedia in order to improve the results of the search for entities. From the results presented in Table 4.11 we can see that the simple title search in the outgoing links of a page improves the effectiveness over the baseline by 9% in terms of xInfAP.

For evaluating the impact on the users, we can look at the value of the $P@10$ which gives an intuition on how many relevant entities the system retrieves at the top (i.e., the part of the results that the user would care most about) while xInfAP evaluates the overall ranking generated by the system.

For completeness, we performed the experiments for the LC task, where the starting categories are extracted from the topic example entities. The results, presented in Table 4.12, are consistent with those of the XER task.

Table 4.10 Top 10 Results using the naïve approach for Topic #109 ("National capitals situated on islands") together with the containing Wikipedia category of each result. Relevance is 0=assessed not relevant; 1=assessed relevant; na=not assessed.

Rank	Entity	Most relevant Category	Relevance
1	County town	capitals	0
2	Kinston, Norfolk Island	capitals in oceania	0
3	Palikir	capitals in oceania	0
4	Washington Capitals	washington capitals	0
5	Port Vila	capitals in oceania	1
6	Belmopan	capitals in north america, cities in belize	na
7	Victoria, Seychelles	capitals in africa	1
8	Honiara	capitals in oceania	0
9	Capital	capitals	0
10	Avarua	capitals in oceania	1

Table 4.11 Average Precision and Precision over the first 10 results for *Categories Based Search* in the XER task. The results marked with * are statistically significant (two-tailed t-Test, $p < 0.05$) as compared to the baseline.

Nr	Method	Query; $q = \ldots$	xInfAP	P@10
1	Baseline	$\{text, W^T\}, \{category, W^C\}$	0.2350	0.3057
2	No Category	$\{text, W^T\}$	0.1125*	0.1429*
3	Title as Category	$\{text, W^T\}, \{category, W^C \cup W^T\}$	**0.2641***	0.3286
4	Category as Title	$\{text, W^T \cup W^C\}, \{category, W^C\}$	0.2190*	0.2571
5	Subcategories	$\{text, W^T\}, \{category, W^C \cup Sub(W^C)\}$	0.1618*	0.2085*
6	Children	$\{text, W^T\}, \{category, W^C \cup Ch(W^C)\}$	0.1616*	0.2057*
7	Siblings	$\{text, W^T\}, \{category, W^C \cup Sib(W^C)\}$	0.1111*	0.1228*
8	Wiki Links	$\{text, W^T\}, \{category, W^C\}, \{outLinks, W^T\}$	0.2561*	**0.3399***

Table 4.12 Average Precision and Precision over the first 10 results for *Categories Based Search* in the LC task. The results marked with * are statistically significant (two-tailed t-Test, $p < 0.05$) as compared to the baseline.

Nr	Method	Query; $q = \ldots$	xInfAP	P@10
1	Baseline	$\{text, W^T\}, \{category, W^{LC}\}$	0.2885	0.3399
2	No Category	$\{text, W^T\}$	0.0998*	0.1200*
3	Title as Category	$\{text, W^T\}, \{category, W^{LC} \cup W^T\}$	0.2836	0.3342
4	Category as Title	$\{text, W^T \cup W^{LC}\}, \{category, W^C\}$	0.2841	0.3428
5	Subcategories	$\{text, W^T\}, \{category, W^{LC} \cup Sub(W^{LC})\}$	0.2466	0.2857
6	Children	$\{text, W^T\}, \{category, W^{LC} \cup Ch(W^{LC})\}$	0.2509	0.2885
7	Siblings	$\{text, W^T\}, \{category, W^{LC} \cup Sib(W^{LC})\}$	0.1108*	0.1171*
8	Wiki Links	$\{text, W^T\}, \{category, W^{LC}\}, \{outLinks, W^T\}$	**0.3006**	**0.3685***

4.7 NLP based Techniques for Entity Retrieval

The approaches proposed so far in this chapter focused on the the category and link structure of Wikipedia while they did not exploit the high-quality language used in the articles. We expect that the use of techniques such as Named Entity Recognition, query expansion using synonyms and other related words may improve the overall effectiveness in the ER tasks. In this section we present our algorithms for improving ER in Wikipedia using Information Extraction (IE) and Natural Language Processing (NLP) techniques [64].

For comparison reasons, we also search the textual part of the query in the *outLinks* index field as presented in Section 4.6. This approach can easily be combined with others to improve performance (e.g., searching the Topic Title in the *text* field AND in the *outLinks* field).

4.7.1 Using Lexical Compounds

Anick and Tipirneni [9] defined the *lexical dispersion hypothesis*, according to which an expression's lexical dispersion (i.e., the number of different compounds it appears in within a document or group of documents) can be used to automatically identify key concepts in the input document set. Although several possible compound expressions are available, it has been shown that simple approaches based on noun analysis are almost as good as highly complex part-of-speech pattern identification algorithms [6]. Verbs, for example, are not very helpful since they are typically too general and used in a variety of different contexts. Lexical Compounds have been already used in different settings for refining Web search queries [46]. We thus extract from simple text all the Lexical Compounds of the following form: $\{\ adjective?\, noun+\ \}$. All such compounds could be easily generated for Title in Topics using WordNet. Moreover, once identified, they can be further sorted depending

on their dispersion within each Topic. We then use Lexical Compounds as search terms in the query, as they present the essential information in a more concise manner. We consider two approaches to using Lexical Compounds in constructing the query. The first uses only Lexical Compounds for constructing the textual part of the query, to search over all the text index field. In the second approach we use the text from the Topic title along with the extracted Lexical Compounds to search over the *textstem* field. For example, for the Title of the Topic in Table 4.7, *National capitals situated on islands*, our algorithm extracted three Lexical Compounds: *national capitals*, *islands* and *national*.

4.7.2 Synonyms and Related Words

Wikipedia, just as the general Web, presents its information in natural language. There is no formal representation and only limited structured information. After describing how to use the structured information, like category information or link structures, we examine different approaches exploiting *natural language properties*.

The first approach accommodates the fact that there are various ways of conveying the same meaning within natural language sentences or even words. This observation lead us to the conclusion that only using the present keywords in the Title, Description, or Category fields is not enough. Therefore, starting from previous research on query expansion through WordNet synonyms ([163], [162], [94], [28], [94], etc.) we extended the query using *related words* and *synonyms* of the extracted keywords.

To add the correct synonyms and related words to the query we need to identify the nouns of a query. For this we use part-of-speech tagging from LingPipe [5] – a suite of Java libraries for NLP. The part-of-speech tagger was trained on the manually labelled Brown corpus, a collection of various types of text documents, to obtain statistical models to perform part-of-speech tagging.

The synonyms and related words were automatically generated using the WordNet semantic lexicon [84]. WordNet can be seen as a dictionary that groups English words into sets of synonyms and stores the various semantic relations between these synonym sets (*synsets*). As there are several synsets available for each term in WordNet, we first perform Word Sense Disambiguation, as done in [146], to choose the correct meaning for the nouns in the query. Then we extend the query with additional information about each noun: (1) add all synonyms from the previously identified synset; (2) add all words that have a relationship (except for antonyms) to the identified synset. The additional words are then used to enrich the query to improve the recall of our system:

$$w_i \in W^T \cup Synonyms(W^T) \text{ or } w_i \in W^T \cup RelatedWords(W^T)$$

4.7.3 Core Characteristics

To make the query more precise, we examined the results for removing parts of the query. On the one hand we *removed duplicate information* in the title by finding synonym nouns

occurring in the category field. This was achieved using WordNet as described in 4.7.2. Since we try to find entities and not categories, the idea is to remove category keywords from the query. Making use of synonym information makes this approach more robust and helps to extract core characteristics from the user query. On the other hand we used LingPipe's part-of-speech Tagger to identify verbs, nouns, adjectives, etc. and removed all except *nouns* and *adjectives*. Observations showed that nouns and adjectives are especially helpful to describe entities, whereas verbs mostly introduce noise to the results due to their generality. The formal notation for this approach is:

$$w_i \in W^T_{Adj} \cup (W^T_{Nouns} \setminus (W^C \cup Synonyms(W^C))$$

4.7.4 Named Entity Recognition

Another well known concept in IE is *Named Entity Recognition*. The knowledge about named entities in the query can be a valuable hint to identify what kind of entity is expected in the answer. We use Named Entity (NE) Recognition provided by LingPipe. Finding named entities can be done using dictionary matching, regular expressions, or statistical approaches. We used a machine learning approach with a model gained from supervised training on a large news article corpus. We identified different named entities like *organizations*, *locations*, and *persons*. The found named entities were then used to perform a keyword search using the following terms:

$$w_i \in W^T \cap \{NamedEntities\}$$

Table 4.13 shows an example of the different Approaches for topic #109 *National capitals situated on islands*.

4.7.5 Experimental Results

Similarly to the previous evaluation methodology, presented in Section 4.6.3, we used the Wikipedia collection provided by INEX. We used the approaches presented in this section and combination of those, with the same notations as used previously, and some additional notations introduced here. Thus, a query is of the form:

$$q = \{(field_i, terms_j)\}$$

where $field_i$ is one of the fields in the Lucene index:

- *text* – the Wikipedia page text;
- *title* – the Wikipedia page title;
- *category* – Wiki categories of the pages;

Table 4.13 Topic #109 after applying the different strategies.

Title	National capitals situated on islands
Category	capitals
Synonyms	capitals islands on National ”working capital” situated
Related Words	**Synonyms** plus additional concepts related mainly to capitals capitals Bahrein ”Galveston Island” ”Hawaii Island” ”Molokai Island” Kriti ”Faroe Islands” Zanzibar Haiti Anglesey ”Vancouver Island” Nihau Corse Ceylon Kahoolawe Moluccas ”South Island” Papua Hibernia Hispaniola ”seed money” Saba ”Aegadean Islands” ”St. Kitts” ”Saint Lucia” ”Visayan Islands” ”Puerto Rico” Sulawesi Iceland ”New Zealand” Curacao Guadeloupe Barbados ”Spice Islands” ”St. Martin” ”Netherlands Antilles” Sicilia ”British Isles” Azores ”Aran Islands” Tobago ”quick assets” Montserrat Formosa Hondo ”Falkland Islands” ”Martha’s Vineyard” Maui situated GU isle Crete Bisayas ”risk capital” Honshu ”Republic of China” Anglesea ”Wake Island” Taiwan ”Kodiak Island” Mindoro Maldives ”Viti Levu” ”Canary Islands” Fijis Krakatao ”St. Eustatius” ”solid ground” Cyprus ”Maui Island” Krakatau Vieques Principe Hokkaido Bali Bougainville ”Baffin Island” Borneo Bonaire ”Oahu Island” Staffa ”Isle of Man” Kodiak Kalimantan assets ”Catalina Island” ”Kahoolawe Island” Corsica Okinawa Saipan Ithaki
Core Characteristics	national
Named Entities	national islands

- *outLinks* – outgoing links of the Wiki pages;

and $terms_j$ is a list of terms which should be searched in $field_i$:

- W^X – a list of words given in the Topic;
- $LexComp(X)$ – extract the *Lexical Compounds* from X;
- $SY(X)$ – apply the *synonyms* approach on the list of words X (e.g., $SY(W^T)$);
- $RW(X)$ – apply the *related words* approach on X;
- $NE(X)$ – extract only the *named entities* from X;
- $CC(X)$ – apply the *core characteristics* approach on X;
- $X \cup Y$ – union of all terms in X and Y;
- $+X$ – all terms in X have to be present in the searched field (conjunction);
- $-X$ – all terms in X must *not* be present in the searched field (negation);

Table 4.14 presents the Average Precision (xInfAP) and Precision for the first ten retrieved results (P@10) of our approaches. Additional to the query presented for each approach, the Category given with the Topic was also searched in the *category* field of the index. The baseline used is approach #1 with a Average Precision and P@10 values of 0.2350 and 0.3057.

Table 4.14 Average Precision and Precision for the first 10 results for NLP based techniques for the XER task. The results marked with * are statistically significant (two-tailed t-Test, $p < 0.05$) as compared to the baseline (#1).

Nr	Query; $q = \{category, W^C\} \cup \ldots$	xInfAP	P@10
1	$\{text, W^T\}$	0.2350	0.3057
9	$\{text, W^T\}, \{outLinks, W^T\}$	0.2556*	0.3371*
10	$\{text, W^T\}, \{outLinks, CC(W^T)\}$	0.2511	0.3114
11	$\{text, W^T\}, \{outLinks, NE(W^T)\}$	0.2504*	0.3171
12	$\{LexComp(W^T)\}$	0.2284	0.2971
13	$\{text, W^T \cup LexComp(W^T)\}$	0.2506	0.3257
14	$\{text, W^T \cup LexComp(W^T)\}, \{outLinks, W^T \cup LexComp(W^T)\}$	0.2616	0.3457
15	$\{text, W^T \cup SY(W^T)\}$	0.2439*	0.3257
16	$\{text, W^T \cup RW(W^T)\}$	0.2398	0.3199
17	$\{text, W^T \cup CC(W^T)\}$	0.2509*	0.3257
18	$\{text, W^T \cup NE(W^T)\}$	0.2530*	0.3257
19	$\{text, W^T \cup SY(W^T) \cup RW(W^T) \cup CC(W^T) \cup NE(W^T)\}$	0.2705*	0.3571*
20	$\{text, W^T \cup SY(W^T) \cup RW(W^T) \cup CC(W^T) \cup NE(W^T)\}$, $\{outLinks, CC(W^T)\}$	0.2682*	0.3599*
21	$\{text, W^T \cup SY(W^T) \cup RW(W^T) \cup CC(W^T) \cup NE(W^T)\}, \{category, W^T\}$	**0.2909***	**0.3971***
22	$\{text, +W^T \cup SY(W^T) \cup RW(W^T) \cup CC(W^T) \cup NE(W^T)\}$	0.0813*	0.1124*
23	$\{text, W^T \cup SY(W^T) \cup RW(W^T) \cup +CC(W^T) \cup NE(W^T)\}$	0.2627	0.3857
24	$\{text, W^T \cup SY(W^T) \cup RW(W^T) \cup CC(W^T) \cup NE(W^T)\}$, $\{outLinks, CC(W^T)\}, \{title, -W^T\}$	0.2748*	0.3657*
25	$\{text, W^T \cup SY(W^T) \cup RW(W^T) \cup CC(W^T) \cup NE(W^T)\}$, $\{outLinks, CC(W^T)\}, \{title, -W^C\}$	0.2534	0.3314

We evaluated our algorithms both independently and as combinations of several approaches. All our approaches improved in terms of both Average Precision and P@10 over the baseline, with the combination of all approaches showing the highest improvement. When compared to the official submissions at INEX-XER 2008[9] our best run with a xInfAP score of 0.29 would place us third participating group as the second best score used also example entities and, therefore, can not be compared. The best performing system at INEX-XER 2008 [159] obtained a score of 0.341 by learning system parameters based on the topic difficulty which is an orthogonal approach to the ones presented in this chapter.

For completeness, in Table 4.15 we present the results for the LC task. The results are again consistent with the XER task. As the LC task was not the focus of our research, we tried only a simple approach for learning the categories and our best result ranks in the middle of the LC runs submitted at INEX-XER 2008. In the following we discuss how different approaches designed for the XER task performed.

Outgoing Links. Approaches #9, #10, #11 from Table 4.14 show the results for searching with the terms from the Topic Title, with the Core Characteristics, and with the Named Entities approaches in the outgoing links text of Wikipedia pages, respectively. The simple (#9) approach shows 10% improvement in P@10 over the baseline. This proves extracting concept names (done as outgoing links in Wikipedia) from entity descriptions to be a

[9] http://www.l3s.de/~demartini/XER08/

Table 4.15 Average Precision and Precision for the first 10 results for NLP based techniques for the LC task. The results marked with * are statistically significant (two-tailed t-Test, $p < 0.05$) as compared to the baseline (#1).

Nr	Query; $q = \{category, W^{LC}\} \cup \ldots$	xInfAP	P@10
1	$\{text, W^T\}$	0.2885	0.3399
9	$\{text, W^T\}, \{outLinks, W^T\}$	0.3006	0.3685*
10	$\{text, W^T\}, \{outLinks, CC(W^T)\}$	0.2995	0.3657*
11	$\{text, W^T\}, \{outLinks, NE(W^T)\}$	0.2919*	0.3571*
12	$\{LexComp(W^T)\}$	0.3011	0.3628*
13	$\{text, W^T \cup LexComp(W^T)\}$	0.3054	0.3571
14	$\{text, W^T \cup LexComp(W^T)\}, \{outLinks, W^T \cup LexComp(W^T)\}$	0.2872	0.3914*
15	$\{text, W^T \cup SY(W^T)\}$	0.3020*	0.3486*
16	$\{text, W^T \cup RW(W^T)\}$	0.2969	0.3342
17	$\{text, W^T \cup CC(W^T)\}$	0.3012*	0.3599*
18	$\{text, W^T \cup NE(W^T)\}$	0.2979	0.3543
19	$\{text, W^T \cup SY(W^T) \cup RW(W^T) \cup CC(W^T) \cup NE(W^T)\}$	0.3187*	0.3771*
20	$\{text, W^T \cup SY(W^T) \cup RW(W^T) \cup CC(W^T) \cup NE(W^T)\}$, $\{outLinks, CC(W^T)\}$	0.3116	0.3743*
21	$\{text, W^T \cup SY(W^T) \cup RW(W^T) \cup CC(W^T) \cup NE(W^T)\}, \{category, W^T\}$	**0.3237**	0.3828
22	$\{text, +W^T \cup SY(W^T) \cup RW(W^T) \cup CC(W^T) \cup NE(W^T)\}$	0.0914*	0.1286*
23	$\{text, W^T \cup SY(W^T) \cup RW(W^T) \cup +CC(W^T) \cup NE(W^T)\}$	0.3221	**0.3914**
24	$\{text, W^T \cup SY(W^T) \cup RW(W^T) \cup CC(W^T) \cup NE(W^T)\}$, $\{outLinks, CC(W^T)\}, \{title, -W^T\}$	0.3093	0.3686*
25	$\{text, W^T \cup SY(W^T) \cup RW(W^T) \cup CC(W^T) \cup NE(W^T)\}$, $\{outLinks, CC(W^T)\}, \{title, -W^{LC}\}$	0.2885	0.3486

valuable additional information for raising early precision values.

Lexical Compounds. In order to evaluate the approaches based on syntactic information we extracted the Lexical Compounds from the Topic Title and we performed several comparisons. The results are presented as #12, #13, #14 in Table 4.14. The simple extraction of Lexical Compounds from the Topic does not show improvements but it is possible to see that the combined usage of Lexical Compounds and the Topic Title performs better than the baseline. Combining the most promising approaches (i.e., searching in the outgoing links and using Lexical Compounds together with Topic terms) improves the results even more.

Synonyms and Related Words. Adding only synonyms of nouns (#15) results in better performance than adding all related words of the nouns (#16). This is due to the vast amount of noise added by RW. Also SY adds some noise as although Word Sense Disambiguation was performed prior to adding the synonyms, still some synonyms are misleading and might need a further filtering step.

Core Characteristics. Approach #17, when used for searching in the whole page text shows the same level of improvement as RW. But when combining it with other methods it improves results significantly. The average numbers for this case are misleading since,

e.g., approach #23 accounts for a couple of top results on a per topic evaluation. This shows extracting the key concepts from both the page text and the query text as being useful for improving early precision.

Named Entity Recognition. Similar to CC, NE (#18) shows statistically significant improvements of 8% for Average Precision. We see that searching with more weight (i.e., duplicating NE words, as they already appear in the Topic Title) the named entities helps improving the ranking.

Combining the approaches. All approaches improve but each ranks entities differently. This leaves room for improvement by combining the single approaches. We performed several combinations and present only the best performing ones. When searching in the page text, we found that including all methods in the query (#19) improves Average Precision by 15% and P@10 by 17%. Adding category (#21) improves even more and reaches 0.291 Average Precision and 0.397 P@10; both with a statistically significant difference with the baseline. This is an improvement of 24% and 30%, respectively.

Efficiency considerations. We have implemented all the presented approaches in the Java programming language. Our test system is an Intel(R) Core(TM)2 CPU (2GHz) and 2 Gbyte RAM, running with *Ubuntu Karmic OS (Kernel 2.6.31-16-generic)*. As Java compiler we use OpenJDK Runtime Environment (IcedTea6 1.6.1). The Lucene index with all the Wikipedia records has a size of 6.0 GByte. Each run (for 35 topics) took on average 0.987 minutes (59.271 seconds), when the system ran single threaded using only one CPU and 1Gbyte RAM.

4.8 Why Are Some Queries More Difficult?

The size of the INEX-XER 2008 corpus is 35 topics. This is not enough to cover all facets of Entity Retrieval or give a complete overview of all different types of entity related queries. It is, however, possible to identify certain patterns which influence the performance of different algorithms on the used test collection and identify different types of queries.

4.8.1 Wikipedia Categories vs. Topic Categories

Category information can be very useful to identify the information needed by the user. Unfortunately, the given category from the user and the existing categories in Wikipedia do not always match as expected. Approaches to solve this problem have been proposed in the first two editions of INEX-XER. For example, in [158] the authors propose to walk the category hierarchy graph up to 3 levels starting from the given topic category in order to find other possible Wikipedia categories that best match the information need. In the following

we analyse INEX-XER 2008 topics with the goal of understanding which topic categories can be used directly and which need to be automatically refined before the retrieval step.

Correctly Assigned Categories The analysis showed that for different types of queries particular approaches perform well while others perform worse. In general we identified on the one hand a set of queries which yielded good results for most of the systems participating at INEX. These "easy" queries have the property that at least one of their assigned topic categories are rather specific categories in Wikipedia as opposed to general categories.

By comparing the scores of the text only approach and the baseline approach (text and category search) we discovered that there are 21 topics for which the given categories help when searching. For eight of these topics, the improvement when using the category information is over 40% xInfAP. When analysing the categories of these topics, we noticed that the given categories are topic specific and also a high ratio of the pages assigned to them are relevant for the query. By specific categories we mean categories that have few pages assigned. For example, for the topic #140, the category *airports in germany* has only 30 pages assigned to it, out of which 28 have been assessed as relevant. Also, for topic #114, category *formula one grands prix* has 27 pages assigned with 15 being relevant.

On the other hand, when the topic categories were too general with respect to the query (i.e., with many pages assigned to them and few of these pages being relevant to the topic), we observed that searching with the text only performed better than using also the category information. For example, for topic #104, the category *harry potter characters* (with 10 relevant out 114 assigned pages), proved to be too general when actually searching for characters that were on the Gryffindor Quidditch team.

Misleading and Inconsistent Category Assignments In the topic set we identified six topics that have low performance (xInfAP smaller than 25%) for all the systems participating at INEX. From these "difficult" topics we noticed that three had categories with no or few pages assigned to them. For example, for topics #106, #125 and #133, the categories *peerage of england* and *country* are empty and whereas *countries* has only two pages assigned to it. For two of these "difficult" topics the categories have been assigned wrong at topic creation time. Thus, for topic #147, the category *eponyms* denotes people, when the topic is about chemical elements. Also, for topic #118, about car models, the specified categories are about car manufacturers.

Another observation is that the topic categories are usually either denoting the type of the desired entities or some property of the relevant entities, or a combination of the two. What happens usually is that the property type categories are too general for the topics, thus hindering the ER performance, as they lead to the retrieval of many different types of entities. For example, category *harry potter* when searching for characters in topic #104 contains no relevant pages in our test corpus. Also, categories such as *hundred years war* when looking for people in topic #106, and *geography of switzerland* for cantons in topic #119, are not particularly helpful on their own.

Interestingly, for the topic #127 ("german female politicians"), where both categories (*politicians*, *women*) represent indeed the types of the desired entities, all the systems had poor performance. This happened because both of the categories had no relevant entities in the Wikipedia corpus used in the experiments. They are all similar to the relevant Wikipedia category, e.g., "bond girls" for Topic #128. Also two third of the easy queries is about persons. On the other hand we have queries which none of the systems could answer satisfactorily. These queries share a broad, respectively general category information. Best performing group on difficult topics is cirquid, see [141]: expanding category information walking 3 steps in the category hierarchical graph, using thus also the categories that are ancestors to the topic categories. This improves then results for topics which had initial categories unrepresented in the INEX Wikipedia corpus (i.e., the categories were empty or had few entities).

4.8.2 Query Terms

Several queries contain certain terms which can change the query meaning almost completely, but are not perceived as such by a classical IR system. For example, by requesting "*living nordic* classical composers" instead of just "classical composers" puts a very high restriction on the query but the means by which an IR system can identify important terms (e.g., Inverse Document Frequency – IDF) will not rate much higher documents containing "nordic composers" instead of "classical composers". The same applies also for "fifa world cup national team *winners* since 1974" where "winners", from the point of view of the system, is just one term out of eight which are similarly important.

4.8.3 Time Span Queries

Certain queries restrict the result set based on time spans, e.g., "French car models in the 1960's". These restrictions require special attention on dates, in query analysis as well as in analysis of potential result pages. To improve precision, systems need to be more time sensitive. The results show that queries with time constraints are difficult for all systems.

4.8.4 Winning Approaches and Topic Types

We have grouped the topics into four sets based on the query methods that had the maximum performance, see Table 4.16 for an overview of the best approaches per topic.

Method 2 – Ignoring Category Information When not using the topic categories and searching only with the topic titles we had maximum performance for six of the 35 topics. These topics were had categories that were either too general (topics #104, #112) or wrongly assigned (topics #118,#147). For example, category *guitarist* has 501 entities assigned to it and out of this only 22 were assessed as relevant to the topic #112.

Methods 3 and 21 – Using the Title as Category For the methods where we additionally searched with the topic titles in the *category* field, we had improvement on 8 topics. This usually happened when the topic title had additional content words that have been used as in category names in the Wikipedia corpus. For example, for the topic #136 with category *baseball players*, the additional words in the title would be *Major League Baseball*. There 10 more categories related to this in the corpus, three of these are related to players. Also, for some of the topics, the title can contain synonyms to words in the category, e.g., for topic #141, we have *Catalunya* in the title and *catalan* in the category.

Methods 22 and 23 – Requiring all Terms to Be Matched When using the NLP techniques that we have mentioned in the previous section we introduce also noise in the queries. Thus, when we make restrictions such as "the result must contain all words from the topic title" or "the result must contain all words from the topic title core characteristics", we give high importance to keywords that might otherwise get lost in the NLP query. This approaches work mostly on short titles, or on topics where the core characteristics are meaningful.

Methods 24 and 25 – Negating Terms in Wiki Titles One of the main issues that we noticed in our experimental results was that we retrieved many non-entity pages. We tried to filter these out by restricting results as to not to contain keywords from the title or category in the result name. This improved the performance on four topics, by filtering out additional related but yet not relevant pages. For example, for topic #130 with the title "star trek captains" and having as categories three of the Star Trek movie titles, there are at least 159 pages that where excluded. All this excluded entities contained the keywords "star trek" in their title as means of disambiguation, that they appeared in the "Star Trek" environment. Movie characters (e.g., *Jean-Luc Picard*) do not contain the movie title in their names.

4.8.5 Concluding Remarks

In this section we have presented an analysis of system performance on the different topics available in the test collection. We have shown how it is important for systems to find good categories also by walking the Wikipedia category graph.

We have seen how easy-to-answer topics have specific categories with few pages assigned to them and that categories contain the desired type of entity (e.g., persons). Difficult topics have categories which are empty or too general, that is, they contain different types of entities in it. It is also important for systems to identify key terms in the query (e.g., "living", "winners") and to process queries containing time spans (e.g., "since 1974").

We have also seen how for some topics it is necessary to ignore the category information provided and for others to use the query for searching the *category* field as the query contains information about the desired entity type. For short queries we have seen that it helps performing phrase queries.

While we only analysed different types of topics used in Entity Retrieval, creating an adaptive system depending on user input is the logical next step to pursue. Similar to [159], the system should employ different algorithms and ranking criteria according to the types of topics identified previously.

Table 4.16 Effectiveness values (xInfAP) on each INEX-XER 2008 topic. We report our best results along with the best performing approach for each topic. See Tables 4.11 and 4.14 for the methods' descriptions.

ID	Title [Categories]	xInfAP	Method
104	Harry Potter Quidditch Gryffindor character [harry potter, harry potter characters]	0.5397	2
106	Noble english person from the Hundred Years' War [peerage of england, hundred years war]	0.1952	20
108	State capitals of the United States of America [u.s. state capitals, capitals, capital cities]	0.6789	23
109	National capitals situated on islands [capitals]	0.2567	23
110	Nobel Prize in Literature winners who were also poets [nobel prize in literature winners]	0.5936	21
112	Guitarists with mass-produced signature guitar models [guitarists]	0.1241	2
113	Formula 1 drivers that won the Monaco Grand Prix [racecar drivers, formula one drivers]	0.1879	23
114	Formula one races in Europe [formula one grands prix]	0.5058	16
115	Formula One World Constructors' Champions [formula one constructors]	0.4014	23
116	Italian nobel prize winners [nobel laureates]	0.4663	23
117	Musicians who appeared in the Blues Brothers movies [musicians]	0.0838	23
118	French car models in 1960's [automobile manufacturers, french automobile manufacturers]	0.0341	2
119	Swiss cantons where they speak German [geography of switzerland, cantons of switzerland]	0.8853	24
121	US presidents since 1960	0.3224	22

Continued on next page . . .

Table 4.16 (continued)

ID	Title [Categories]	xInfAP	Method
	[presidents of the united states, u.s. democratic party presidential nominees, u.s. republican party presidential nominees]		
122	Movies with eight or more Academy Awards [best picture oscar, british films, american films]	0.1585	2
123	FIFA world cup national team winners since 1974 [football in brazil, european national football teams, football in argentina]	0.094	2
124	Novels that won the Booker Prize [novels]	0.4646	23
125	countries which have won the FIFA world cup [countries]	0.1111	23
126	toy train manufacturers that are still in business [toy train manufacturers]	0.2573	21
127	german female politicians [politicians, women]	0.1088	3
128	Bond girls [film actors, bond girls]	0.7294	22
129	Science fiction book written in the 1980 [science fiction novels, science fiction books]	0.3969	3
130	Star Trek Captains [star trek: the next generation characters, star trek: voyager characters, star trek: deep space nine characters]	0.1705	25
132	living nordic classical composers [21st century classical composers, finnish composers, living classical composers]	0.0523	4
133	EU countries [country]	0.0663	23
134	record-breaking sprinters in male 100-meter sprints [sprinters]	0.1339	19
135	professional baseball team in Japan [japanese baseball teams]	0.7453	18
136	Japanese players in Major League Baseball [baseball players]	0.3107	3

Continued on next page . . .

Table 4.16 (continued)

ID	Title [Categories]	xInfAP	Method
138	National Parks East Coast Canada US [national parks, national parks of the united states, national parks of canada]	0.2286	3
139	Films directed by Akira Kurosawa [japanese films]	0.9198	24
140	Airports in Germany [airports in germany]	0.9912	10
141	Universities in Catalunya [catalan universities]	0.7058	21
143	Hanseatic league in Germany in the Netherlands Circle [cities, cities in germany]	0.215	9
144	chess world champions [chess grandmasters, world chess champions]	0.7167	3
147	Chemical elements that are named after people [eponyms]	0.0469	2

4.9 Discussion

In this chapter we first described our effort on creating standard and reusable test collection for ER over two different Wikipedia corpora. Then, we presented a general model for ranking entities and we showed how the model can be applied to different real world scenarios. We described in detail a possible instantiation of the model and a set of algorithms designed for the Wikipedia dataset. We make use of the Wikipedia structure – page links and categories – and employ an accurate ontology to remove possible noise in Wikipedia category assignments. The results show that, in the used test collection, category assignments can be both very helpful for retrieval as well as misleading depending on the query syntax. We also employ several NLP techniques to transform the query and to fill the gaps between the query and the Wikipedia language models. We extract essential information (lexical expressions, key concepts, named entities) from the query, as well as expand the terms (by means of synonyms or related words) to find entities by specific spelling variants of their attributes. By combining several techniques we can achieve a relatively high effectiveness of the ER system; still, further improvement is possible by selectively applying the methods for different queries. The experimental evaluation of the ER algorithms has shown that by combining our approaches we achieve an average improvement of 24% in terms of xInfAP and of 30% in terms of P@10 on the XER task of the INEX-XER 2008 test collection. While the proposed techniques were designed for the XER task, experimental results for the list completion task are consistent. While more experimentation is needed to

conclude that the proposed techniques perform well in general, we have shown how they improve effectiveness on the used test collection.

We also saw that it might be possible to apply and/or combine different approaches depending on the query in order to maximize effectiveness – for example, by using our methods we achieve an xInfAP value of over 0.7 for 20% of the queries of the used test collection and the mean xInfAP can be further boosted by 27% only by selecting the appropriate approach for each given topic. We leave as future work the research question of automatically selecting appropriate approaches for each query (e.g., by estimating the expected number of relevant results). We also point out that initial steps toward this goal have been done in [159] by applying machine learning techniques to predict query difficulty.

In this chapter and in related work (see Section 4.2) it is possible to notice that precision values are low overall. This indicates that the Entity Retrieval research field is only at its beginning and needs more work focusing on high precision algorithms in order to provide the users with a satisfying search experience. It is worthwhile investigating how to automatically determine (e.g., by statistics about the number of pages in the sought category or by frequency of categories for pages) when the category information should be used as-is and when this should be further processed or even ignored. Also, more focused research on NLP based techniques should be performed to broaden or narrow the query specificity depending on prediction of effectiveness by means of query analysis and classification. Finally, search effectiveness of the XER task can be further improved by using available collections (e.g., Wikipedia) annotated with state of the art NLP tools or enriched with semantic information (see, e.g., [101, 145]). A current limitation of this work is that the described algorithms are designed for the Wikipedia setting and can not be directly applied to the Web at large. It will be focus of our future work to extend the proposed methods for the Web of Entities. In the next chapter we focus on additional dimensions of the ER problem, that is, entity relevance evolution over time and public opinion about politicians.

5 Additional Dimensions of ER: Time and Opinion

5.1 Introduction

In the previous chapter we have proposed approaches for finding entities in Wikipedia which is seen as a static document collection. We now focus on different dimensions of ER: time and opinions. In this chapter, we will study the ER task when performed on a time-stamped document collection as well as the problem of estimating public opinion over time about specific entities in the political domain.

5.1.1 Motivation

As we have seen in the previous chapters, ER can allow users to find more than just Web pages: also people, books, movies, cars, etc. It helps people find directly the information they are after and to reformulate the query precisely in a natural way. For such reasons, ER is becoming a major area of interest in IR research and is quickly being adopted in commercial applications. One of the promising areas of application of ER models in the commercial world is in *news search*[1].

A possible application consists in enriching the user interface by placing retrieved entities next to the news article the user is currently looking at. For example, we can imagine a user looking for the event of Charles Schultz death. A classic Search Engine will present the user with a relevant news article about the event. In this case, by exploiting ER techniques, we can enrich the visualization with relevant entities such as locations, prizes, characters, etc. One example mock-up interface is shown in Figure 5.1 where, additionally to the news article matching the user query, important entities are shown. The study of which methods are suitable for identifying such entities is the focus of our work. News Retrieval has also been the focus of much attention in the IR research community (e.g., [76, 88]), but to our knowledge there have been no ER tasks defined for news.

[1] http://news.bbc.co.uk, http://news.google.com, http://news.yahoo.com

Figure 5.1 Example of user interface for Entity Retrieval in news articles.

Dealing with ER in news is particularly interesting as news articles are often focused on entities such as people, companies, countries, etc. It is also a challenging task as, differently for standard ER tasks, there is the time dimension involved. Given a news topic, the decision about which entities should be retrieved or not changes with time. Not all frequently appearing entities should be considered relevant to the topic (e.g., news agencies) and new important entities may appear only later in the story (e.g., witness of a murder).

We propose a system which takes into account both information from the current news article as well as from the past relevant articles in order to detect the most important entities in the current news.

Another interesting dimension that evolves over time is public opinion about specific entities. The blogosphere has attracted an active Web community in the recent years and has become a popular forum for sharing experiences, opinions, and thoughts on a variety of issues. Topics discussed range from rather casual themes such as sports, concerts, or celebrities to more complex and polarizing political ones such as abortion, elections, or immigration. Opinions on products are usually present as well, examples being books or movie reviews, and experiences made with laptops or digital cameras.

Blog data constitutes a powerful source for mining information about opinions and trends. Software providing blog writing functionalities has become easy to use enabling not only young people and technical experts to publish on the Web but allowing any kind of Web user to generate content. Blog entries are connected to metadata providing additional information useful for data mining such as timestamps, and, quite often, information about user profiles including attributes such as gender, age, or location. The possibility for users to quickly publish new content at any point in time (e.g., immediately after a particular event) allows for up-to-date views on a large spectrum of topics and opinions.

Public opinion on different topics is usually estimated by professional services conducting surveys on a sample of the population. For instance, opinions about political elections are estimated by interviewing electors on the phone. This is clearly an expensive activity for both the company carrying out the interviews as well as for the sampled electors who have to spend their time answering questions. Such high costs also limit the applicability of this approach as it can not be carried out on a large scale both in terms of regarded opinion

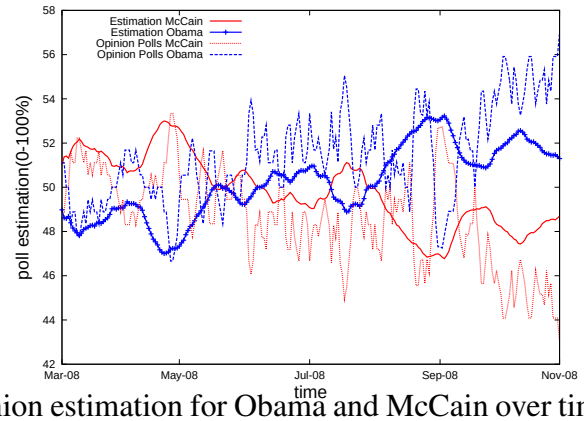


Figure 5.2 Opinion estimation for Obama and McCain over time using smoothing by moving average on 15 intervals.

holders (usually samples of at most 1000 persons are considered) as well as in terms of covered topics.

In this chapter, we propose an approach towards overcoming the drawbacks of opinion polls, and present models for automatically estimating public opinions from the blogosphere by mining and aggregating information extracted from blogs over time.

We focus on the scenario of the US 2008 election campaign for our evaluation as it constitutes a highly polarizing topic for which a large amount of blog entries is available. Furthermore, there exists sufficient ground truth in form of large, professional opinion polls which we can use as test data for our approach. Figure 5.2 depicts an example comparing our automatically computed estimate (exclusively based on blog data) to traditional opinion polls about the main rival candidates Obama and McCain over time, illustrating how blogs are reflected in the real world.

5.1.2 Contributions

In this chapter we first address the problem of ranking entities in news applications. We introduce the original task of Time-Aware Entity Retrieval (TAER) which takes into account the evolution of entity relevance over time in a news topic thread. To evaluate the effectiveness of systems performing such task we develop an extension of the TREC 2004 Novelty corpus [152], annotating relevance at the level of entities. We then develop features and ranking models for the original TAER task.

Specifically, we design both *local* and *history* features exploiting appearance of entities in the text. Our main findings presented are: sentence novelty is worse than pure sentence relevance as an indicator of entity relevance; entities that become relevant have a high probability of remaining relevant the next article and the entire news thread; the relevant history of an article (e.g., the previous relevant articles) can be exploited as a source of information for TAER.

For the opinion mining task, we define a pipeline of several IR and NLP components

to determine the public opinion towards the two US 2008 presidential candidates, Obama and McCain. To this end, we first retrieve blog postings relevant to a topic and analyse opinions expressed in these using both thesauri with sentiment annotations and sentiment classification techniques. Then, we aggregate sentiments for blog postings of given time intervals in order to determine the evolution of public opinion over time. Moreover, we show how model parameters can be automatically tuned in cases where a small history of (traditional) poll results is available. To this end, we adjust the bias of the blogosphere, that is, the difference of opinions introduced by using as sample the set of bloggers instead of a carefully selected population sample. Another parameter that we account for automatically is the time lag: people can express their opinions in blogs at any point in time while phone interviews are carried out, their outcome is processed, and, finally, results are published, resulting possibly in delays compared to the blogosphere. Finally, we combine our approach with techniques from time series analysis and smoothing methods to reduce noise.

This chapter is organized as follows. Section 5.2 presents related work on entity search, novel content retrieval, time-based IR, blog analysis, prediction markets and time series analysis. In Section 5.3 we present our work on Time-Aware Entity Retrieval. In detail, Section 5.3.1 defines the task we address comparing it to standard ER and introduces the dataset we created for evaluating time-aware entity search. Section 5.3.3 introduces and motivates several features extracted from documents and entity history in order to rank entities. Section 5.3.4 presents an experimental evaluation of the aforementioned features, and 5.3.5 describes an additional task, i.e., entity profiling, with some preliminary results. In Section 5.4 we present the work on estimating public opinion about entities in the political domain. In detail, we describe several techniques for extracting opinions from blogs in Section 5.4.1, both in a supervised and unsupervised context. In Section 5.4.2 we show the results of the evaluation of our automatic estimation methods for the US 2008 election campaign providing both qualitative and quantitative results. We conclude and show directions of our future work in Section 5.5.

5.2 Related Work

To our knowledge, ER has not been studied in the context of news search or time-stamped collections. However the different components needed for this task (sentence retrieval in news, Entity Retrieval, etc.) are active areas of research in the IR community.

Entity Retrieval Searching for entities is a common user activity on the Web as revealed by query log analysis on commercial Search Engines [106]. There is an increasing effort in the research community in developing entity search techniques and in building evaluation benchmarks. One example is the expert finding task evaluated in the context of the TREC Enterprise Track [16], where the goal is to find entities (people) that have relevant expertise about a topic of interest. Language models-based approaches [17, 18] and voting models [119] are among the most promising techniques for ranking experts. The INEX Entity

Ranking Track is another evaluation initiative where the task is to return a list of relevant Wikipedia entities for a given topic using an XML collection [60]. Vercoustre et al. [161], use Wikipedia categories and link structure together with entity examples to significantly improve ER effectiveness. Rode et al. [142] demonstrate the advantage of combining the Wikipedia article's score with the propagated scores from text segments of different sizes. A different approach is taken in [170] where the ER task is carried out by annotating entity occurrences in the text and, thus, without the assumption that an entity is represented by a Wikipedia page. In the TREC 2009 Entity Track [15] the task of finding related entities given one entity as query (e.g., "Airlines that currently use Boeing 747 planes") was investigated. Compared to previous work on ER we analyse the usefulness of the time dimension for this task.

Novel Content Retrieval With respect to the news domain, the TREC Novelty Track defined a task that takes into account the chronological order of documents. How to identify sentences that carry novel information with respect to previous retrieved content has been evaluated in [91, 152, 153]. The best performing approach (in terms of F-Measure) at the TREC Novelty Track 2004 used a variation of TF-IDF in order to detect new sentences [88]. It has been shown that exploiting the presence of entities can improve the effectiveness of novel sentence retrieval. Li and Croft presented an approach based on the presence of named entities [112, 113, 114]. Zhang and Tsai [171] employed named entity recognition and part-of-speech tagging to propose a mixed method for novel sentence retrieval. Novelty retrieval studied the retrieval of novel information in documents. We selected the TREC Novelty collection as it allows us to exploit information about novel content and the time dimension of the document collection in order to consider novel entities which may be of interest for TAER.

Time-based Information Retrieval Time-based Information Retrieval is an active related research area. The time dimension can be exploited in several search tasks [8]. Some authors [10, 111] have proposed an adaptation of language models to incorporate temporal expression in order to enable text search based on time. Diaz [76] proposed models for classifying whether Web search queries are news-worthy or not over time. He used information from previous clicks predicting the usefulness of displaying news in the result page at a given point as topics and interests develop over time. Alonso et al. [7] studied how the time dimension can enhance the presentation of query results. Berberich et al. [27] proposed an extension of the inverted index for temporal search (i.e., text search over temporally versioned document collections). Compared to previous work on Time-based Information Retrieval we focus on retrieving entities instead of documents.

Blog Mining Blog mining has primarily been studied in the context of the Text REtrieval Conference[2] (TREC) Blog Track since 2006. The main tasks are about retrieving opinion-

[2] http://trec.nist.gov/

ated postings and their polarity, and retrieving topical items both at posting and feed level. In 2009 the track introduced a new dataset of blogs from January 2008 to February 2009 which is a good collection for the case study of the US presidential election held in November 2008 that we conduct in this thesis. In order to retrieve opinions most approaches adopt a two-stage process. First, documents are ranked according to topical relevance, using mostly off-the-shelf retrieval systems and weighting models (e.g., language models adapted from expert finding [22]). Results are then re-ranked or filtered by applying one or more heuristics for detecting opinions. In order to detect opinionated items machine learning techniques (e.g., Supporting Vector Machines [98]) or lexicon-based approaches (based, for instance, on SentiWordNet and the Amazon review data [108]) are applied. In contrast to TREC Blog tasks and previous work we are using the described techniques rather as preprocessing steps towards providing an aggregated and time-aware view on opinions.

In the NIST Text Analysis Conference[3] (TAC) the tasks are rather about creating summaries of blogs, mainly by retrieving appropriate sentences, and to recognize textual entailment[4], that is, given two text fragments to decide whether the corresponding statements concur, contradict each other, or are not related. Our approach does not consider such techniques, and uses simpler and more efficient algorithms enabling large scale analysis and aggregation. Other work on blog analysis aims at extracting knowledge from blogs of individual authors. In [126], for instance, the authors identify books the blogger would buy in the near future. In [23] the evolution of blogger moods in order to identify events is studied. Blogs are classified into categories "personal" vs. "non-personal" in [78]. However these articles do not compute any aggregated estimates of temporal developments. Related to the scenario of US 2008 elections is the work of Leskovec *et al.* [109] who, however, focus on the mutual influence of news messages and blogs. More related to our work is that by Liu *et al.* [115] in the context of sales prediction where the authors predict movie incomes by mining blogs, and combine latent analysis techniques for determining sentiment with techniques from time series analysis. In contrast to our work, the approach always requires the availability of training samples, and predictions are made just for a very short time period (typically around 10 days). In their recent work [128], O'Conner *et al.* study the aggregation of opinions on Twitter data, using more simplistic methods for aggregating sentiment over a sample of tweets. However, specifically for the challenging US election scenario, they report just a very low correlation between their estimates on Twitter and traditional polls, and they are not able to capture trends.

Prediction markets and Forecasting The idea of prediction markets is to create a speculative market similar to a stock market targeted at a particular event (e.g., "Who will become the next US president?"). People can buy and sell "shares" of a possible result receiving at the end a reward in case the event really happens. Additionally, participants can buy and sell before the event happens given that the price changes over time. Thus, market prices can be viewed as predictions of the probability of an expected result. This is

[3] http://www.nist.gov/tac/
[4] http://www.nist.gov/tac/2009/RTE/index.html

realistic as people invest real money and we can assume they use all available knowledge to predict the final result. Evaluations [85, 167] show that the analysis of prediction markets returns a good estimate of the final result also providing a temporal evolution of the probabilities (of people voting for a candidate as if the election would be today). Compared to this approach, we aim at mining already available content on the Web in order to estimate the evolution of opinions over time without requiring additional effort from users.

In the area of forecasting, information about historical and present data is extrapolated in order to predict future developments. Common features in the context of elections are the popularity of the current president, economic growth, inflation rate, and "time-to-change", i.e., how many consecutive terms in the White House a certain party had [2, 151]. These approaches require lots of manual effort and domain knowledge, and cannot be easily generalized. Time Series Analysis (TSA) techniques [166] typically learn regression models to predict future values from series of data observed in the past; however, they do not make use of additional information sources such as the blogosphere and, thus, do not capture external influences.

5.3 Time-Aware Entity Retrieval

In this section we study the problem of Entity Retrieval for news applications, and in particular the importance of the news trail history (i.e., past related articles) in determining the relevant entities in current articles. This is an important problem in applications that display retrieved entities to the user, together with the news article. For example, a user looking to a news article about the soccer World Cup 2010 Final may be presented, next to the article text, with a list of important entities such as, for example, Andrs Iniesta, Spain, Johannesburg, South Africa, etc.

We begin with the TREC 2004 Novelty collection and develop a new labelled corpus at the entity level for this task. We analyse and discuss some statistics about entities in news trails, unveiling some unknown findings such as the persistence of relevance over time. We focus on the task of query dependent Entity Retrieval in news over time. For this task we develop and evaluate several features, and show that an article's history (previous related articles) can be effectively exploited to improve effectiveness. Finally, we show that combinations of the proposed features significantly improves performance.

5.3.1 Task Definition

Standard Entity Retrieval is defined over a set of documents D and a query q as follows:

- Entity Retrieval (ER): Given a query and a document collection, retrieve a set of entities appearing in the collection which are relevant to the query.

For example, the ER task was performed in [170] using Wikipedia as a document collection.

Consider the following user scenario: a user types a query (or topic) into a News Search Engine and obtains a list of relevant results, ordered by time. Furthermore, the user subscribes to this query so in the future she will continue to receive the latest news on this query (or topic). We are interested in ER tasks related to this user scenario. Standard ER could be used to show to the user the most interesting entities *for the query*. The temporal dimension is not needed here.

However, if the user is observing a current document, we may want to show the most relevant entities of the document for her query (or topic). This prompts a first task definition:

- Time-Aware Entity Retrieval (TAER): Given a query and a document relevant to it, and possibly a set of previous related documents (the *history* of the document), retrieve a set of entities that best summarize the document.

This is a newly defined task that can be useful, for example, in news verticals for presenting the user more than just a ranked list of documents. In the news context we define the task for most considered entity types: persons, locations, organizations, and products. More formally, we define a "news thread" relevant to a query as the list of relevant documents $D = [d_1 \dots d_n]$. Then, given a document d_i we define its history as the list of relevant documents $H = [d_1 \dots d_{i-1}]$ chronologically ordered pre-dating the document d_i. Given an entity e, we note as $d_{e,1}$ the first document in which the entity occurred in the news thread. Note that such a document is not necessarily the first document in D as entities may appear only in subsequent documents. Additionally, we will note as $d_{e,-1}$ as the last document in H which contains e.

5.3.2 Evaluation Testbed for Time-Aware Entity Retrieval

The TREC Novelty Track in 2004 consisted on a collection of news articles and a set of topics for evaluating retrieval of novel information over ranked lists of documents for each topic. The systems had to retrieve information (i.e., sentences in this case) relevant to the topic and not yet present in the retrieved results [152]. That is, a novel sentence 1) is relevant to the topic and 2) contains new information compared to the sentences retrieved before it. A time-stamped list of documents is provided for every topic reflecting the temporal flow of the story the topic is about.

Dataset Description

We selected the 25 'event' topics from the latest TREC Novelty collection (2004). We annotated the documents associated with those topics using state of the art NLP tools [11, 170] in order to extract entities of type person, location, organization, and product based on WSJ annotations. The annotation system detected 7481 entity occurrences in the collection: 26% persons, 10% locations, 57% organizations, and 7% products.

Table 5.1 Example entities and their judgements in the first two articles for topic N79. Some entities keep their relevance status and others change it. Entities could appear only in some articles. Some annotations may not represent entities.

Topic N79: Charles Schulz Dies		
	APW19991001.0198	APW19991027.0332
Schulz	Relevant	Relevant
Peanuts	Relevant	-
Charlie_Brown	Related	Related
Snoopy	Related	NotRelevant
Santa_Rosa	Related	Relevant
center	-	NotAnEntity

Six human judges assessed the relevance of the entities in each document with respect to the topic grading each entity on the 3-points scale: Relevant, Related, Not Relevant. An additional category, i.e., 'Not an entity', was used to mark entities which had been wrongly annotated by the NLP tool. A total of 21213 entity-document-topic judgements were obtained in the collection[5]. Examples of judged entities over two documents for a specific topic are shown in Table 5.1. The topic is about the event of the Peanuts' author death. Over the entire news thread some entities are relevant all the time (e.g., Schulz), some appears only after some time (e.g., his wife), and others are always present but with different relevance status (e.g., the city of Santa Rosa sometimes commemorates him and is therefore relevant, while other times it is just the place where the news has been written and is therefore not relevant). We can see entities of different types (e.g., persons, cities) and that some are highly relevant and stay relevant over different documents (e.g., Schulz). Other entities may change relevance status from related to relevant (e.g., Santa_Rosa) as they play a critical role in the current article, or to not relevant (e.g., Snoopy) as they are just mentioned in the current article. Moreover, some annotations do not represent named entities and are judged accordingly (e.g., center).

We performed double assessments on six topics in order to check the assessors' agreement obtaining an average Cohen's Kappa of 0.5232. Looking at agreement rates in other relevance judgement settings (0.49 on 40 topics at TREC 2006 Legal Track [24], 0.34 on 100 documents for opinion detection [129], 0.55 on sentence relevance at TREC 2004 Novelty Track [154]) we can see how entity relevance in a news corpus is less subjective than in other settings such as, for example, opinion detection.

[5]The evaluation collection we have created is available for download at: http://www.l3s.de/ demartini/deert/

Analysis of the Dataset

The TREC 2004 Novelty collection consists of an average of 31.2 news articles per topic distributed over time. After the annotation process, each document in the collection contains on average 26.5 annotated entities among which 7.6 were judged relevant. On average each topic contains 63.4 entities which have been marked relevant at least once over the topic timeline.

We now investigate the relation between entities, sentence and relevance. Let n_s, r_s indicate that a sentence s is novel or relevant respectively. Let t_e indicate the type of entity e, and let us denote by r_e the fact that e is relevant, and $\bar{r}_e$ otherwise.

On average, a sentence contains 1.46 entities, a relevant sentence contains 1.88 entities, and a novel sentence contains 1.92 entities which indicates the presence of more information. The unconditional probability of a relevant entity in a sentence $P(r_e)$ is 0.411 (we first sample a sentence and then an entity in that sentence). The probability of finding a relevant entity in a relevant sentence $P(r_e|r_s)$ is 0.547 with a 95% bootstrap confidence interval of $[0.534-0.559]$, well above $P(r_e)$. The probability of a relevant entity in a novel sentence $P(r_e|n_s)$ is 0.510 $[0.491-0.531]$ which is below the probability in a relevant sentence.

This gives the following high level picture. Relevant sentences contain slightly more entities than non-relevant ones. Novel sentences contain slightly more entities than relevant (but not-novel) sentences; however, entities in novel sentences are more likely to be irrelevant than in not-novel sentences.

In Table 5.2 we look at relevance probabilities per entity type (e.g., the probability of person entity being relevant would be noted $P(r_e|t_e = person)$). We show again that sentence novelty is less important than sentence relevance *regardless of the entity type*. Organization entities are more likely in a relevant sentences than the rest (63% of those appearing in relevant sentences have been marked relevant).

As compared to a classic document collection, in a news corpus the time dimension is an additional available feature. How useful is the information from past news articles? The probability of an entity being relevant in a document given that it was relevant the first time it appeared ($d_{e,1}$) is 0.893 $[0.881-0.905]$ which shows how in most cases an entity which is relevant at the beginning of its appearance stays relevant for the rest of the news thread. It is also important to observe just the previous document where the entity appeared. The probability of an entity being relevant in a document given that it was relevant the previous time it appeared is 0.701 $[0.677-0.726]$. Conversely, the probability of a relevant entity changing relevance status form one story to the next is 0.3. Another characterization of this is the probability of an entity being relevant in a document given that it was relevant in the i-th document of its history. This is shown in Figure 5.3 for relevant, related and not-relevant entities. We can see that relevant entities are the most stable over time while related entities tend to change relevance status over time (either to relevant or to not-relevant).

Table 5.2 Probabilities of relevance for different entity types with 95% confidence intervals.

$P(r_e\|t_e = person)$	0.406 [0.391-0.421]
$P(r_e\|t_e = person, r_s)$	**0.560** [0.533-0.588]
$P(r_e\|t_e = person, n_s)$	0.496 [0.451-0.541]
$P(r_e\|t_e = organization)$	0.479 [0.471-0.487]
$P(r_e\|t_e = organization, r_s)$	**0.631** [0.616-0.646]
$P(r_e\|t_e = organization, n_s)$	0.587 [0.564-0.612]
$P(r_e\|t_e = product)$	0.179 [0.164-0.194]
$P(r_e\|t_e = product, r_s)$	**0.237** [0.210-0.265]
$P(r_e\|t_e = product, n_s)$	0.189 [0.151-0.228]
$P(r_e\|t_e = location)$	0.284 [0.271-0.297]
$P(r_e\|t_e = location, r_s)$	**0.403** [0.379-0.427]
$P(r_e\|t_e = location, n_s)$	0.397 [0.363-0.432]

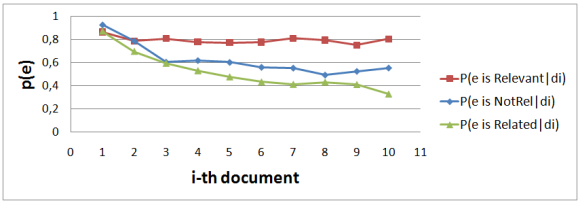


Figure 5.3 Probabilities of entity relevance given its relevance in the i-th document of its history (i.e., past related articles).

5.3.3 Features for Time-Aware Entity Retrieval

In the TAER task we are given a query q and we want a ranking function that sorts the set of entities e_i occurring in document d according to their relevance. As we have seen in Section 5.3.2, relevant entities often appear in relevant sentences. More interestingly, the past articles seem to be a good evidence for entity relevance. For such reasons in the following we present a set of features that can help ranking entities in news articles both considering attributes from the current article as well as from the news published in the past on the same topic.

Local Features

We aim at retrieving entities described in a document d, thus, the first thing to do is to exploit entity occurrences in d. As most important entities should appear more often, the first feature we consider is the frequency of an entity e in a document d, noted $F(e,d)$. In the following we will use this feature as our baseline.

As we have seen in Section 5.3.2, relevant sentences contain more relevant entities.

Therefore, a natural extension of the baseline is obtained considering the relevance score of the sentences where e appears with respect to q. We computed the BM25 scores [139] of sentences with respect to a disjunctive query consisting of all the terms in the topic title[6]. We can therefore rank entities according to the average or the sum of BM25 score of the sentences where e appears in d (noted $AvgBM25s(e,d)$ and $SumBM25s(e,d)$ respectively).

Key entities are often those performing certain actions in a news story. After running a dependency parsing over the sentence collection, it is possible to consider if an entity appears as a subject of a sentence as this is generally the person or thing carrying out an action. Hence, we define the $F_{subj}(e,d)$ as the number of times an entity e appears as subject of a sentence in the document d.

In the writing style adopted for news it is a common practice to summarize the story at the beginning of the article and provide more details in the following. Thus, we expect to find key entities toward the beginning and less important entities afterwards. We additionally propose two position-based features that take into account where in document d an entity e appears. Let $FirstSenLen(e,d)$ be the length of the first sentence where e appears in document d and $FirstSenPos(e,d)$ be the position of the first sentence where e appears in d (e.g, the fourth sentence in the document).

History Features

As the dataset analysis shown that past related articles may contain important information about entity relevance (see Section 5.3.2), we now introduce a number of features that take into consideration the document history H. As defined for the current document, we can obtain a simple feature just by counting the occurrences of an entity in the past. Let $F(e,H)$ be the frequency (i.e., the number of times it appears) of the entity e in the history H.

As documents may have different length and, thus, contain more or less entities, it is possible to refine the previous feature taking this fact into account. Instead of counting each entity occurrence a simpler variation considers the number of documents in which the entity e has appeared so far. We thus define $DF(e,H)$ as the document frequency of e in H.

More than just looking at the entire set of past documents we can also consider specific documents. When a news story begin the key entities are already present. We thus define $F(e,d_{e,1})$ as the frequency of entity e in the first document where the entity appeared. As we have seen in Section 5.3.2, the previous document is also an important evidence of entity relevance. We define $F(e,d_{e,-1})$ as the frequency of entity e in the previous document where the entity appeared.

In news stories important entities interact with many other ones. We can compute $CoOcc(e,H)$, the number of other entities with which the entity co-occurred in a sentence in the set of past documents H. Finally, we leave the study of the influence of recency in the effectiveness of these features (i.e., do more recent documents provide better evidence as compared to older ones?) for future work.

[6] We computed BM25 with $b = 0.75$, $k_1 = 1.2$.

We can have an initial analysis of such features by checking how entity relevance probability changes with the features value. Figure 5.4 shows the probability of an entity being relevant given different values of the features described above. We see that all are correlated with relevance over their entire domain.

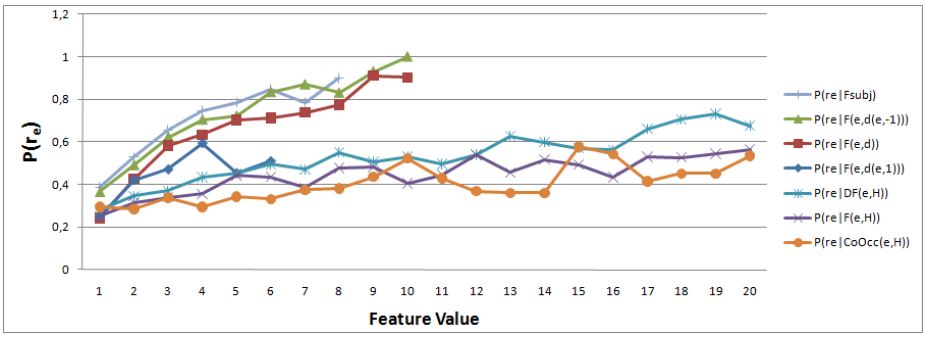


Figure 5.4 Probability of an entity being relevant given different feature values for several features.

5.3.4 Experimental Results

In this section we present the experimental evaluation of the features proposed for the TAER task.

We compare the effectiveness of different features and some feature combinations using several performance metrics. In order to evaluate the complete entity ranking produced by the proposed features, we compute Mean Average Precision (MAP). For completeness, as we aim at showing the user few entities, we check for early precision as well. We report values for Precision@3 (P@3), Precision@5 (P@5), and we test for statistical significance using the t-test. Because there were defined three levels of relevance when evaluating the test collection, we need to fix a threshold for binarising the relevance. In the following we consider related entities as non-relevant. As future work we will study effectiveness of our approach on Related entities. Many of the features we use are based on entity frequency, hence entity scores in the ranking will have many ties. For this reason, the evaluation measures we have computed are aware of ties, that is, they consider the average value of the measure for all possible combinations of tied scores [123].

Evaluation of single features

Local Features Table 5.3 shows effectiveness values obtained when ranking entities in a document according to local features, where no single feature performs better than the simple frequency of entities in the document. For comparison, a feature that assigns the same score to each entity would obtain a MAP value of 0.42 with a ties-aware measure. The feature $F(e,d)$ obtains the best MAP value (0.60). The second best local features is

Table 5.3 Effectiveness of local features for TAER.

Local Features	P@3	P@5	MAP
All Ties	0.34	0.34	0.42
F(e,d)	**0.65**	**0.56**	**0.60**
FirstSenLen	0.37	0.36	0.45
FirstSenPos	0.31	0.31	0.43
F_{subj}	0.49	0.44	0.50
AvgBM25s	0.27	0.30	0.41
SumBM25s	0.50	0.44	0.52

*SumBM*25*s* (0.52 MAP) which takes into consideration relevance of sentences where the entity appears. On the other hand, the features looking at the first sentence where the entity appears in the news article (FirstSenLen, FirstSenPos) do not perform well (0.45 and 0.43 MAP respectively). In order to exploit the position of the first sentence where an entity appears we need to deal with the problem of headers in news articles (e.g., news agency codes): as articles have different header lengths, it is not easy to detect the beginning of the article body. Additionally, three different news agencies contributed articles to the collection each of them having different formatting standards. For example, the agency NYT can have articles where the title and body do not start before the tenth sentence while for others (e.g., XIE) the interesting content can start already at the third sentence. The transformations of FirstSenPos that we explored did not improve performances of this feature.

History Features Table 5.4 presents the performance of TAER using history features. In general, history features perform better than local features and the highest performance is obtained by ranking entities according to its frequency in the past documents ($F(e,H)$). All history features but $F(e,d_{e,1})$ significantly improved over the baseline in terms of MAP. In terms of early precision (P@5) only $F(e,H)$ and the similar feature $DF(e,H)$ improve over the baseline. Moreover, features using the entire history H are performing better than features looking at single documents in the past.

It is also interesting to note that, when identifying relevant entities for a document, the frequency of the entity in the previous document in the story $F(e,d_{e,-1})$ is a better evidence than the frequency in the current document. This may be an indication of how people read news: some entities become relevant to readers after repeated occurrences. If an entity appears in the current and previous documents it is more likely to be relevant.

We additionally weighted the scores obtained from different documents in H with both the document length and BM25 score of the document with respect to the query. This approach did not improve the effectiveness of the original features without per-document weighting.

Given these results we conclude that the evidence from the past is very important for ranking entities appearing in a document. Thus, we expect effectiveness of methods that

Table 5.4 Effectiveness of history features for TAER and improvement over $F(e,d)$. In brackets the % improvement over $F(e,d)$. *(**) indicates statistical significance w.r.t. $F(e,d)$ with p¡0.05(0.01).

History	P@3	P@5	MAP
$F(e,d)$	.65	.56	.60
$F(e,d_{e,1})$	.58(−11%)	.53 (−6%)	.56 (−7%)
$F(e,d_{e,-1})$	.64 (−2%)	.56 (±0%)	.62* (+3%)
$DF(e,H)$	.63 (−3%)	.57* (+1%)	.65** (+8%)
$F(e,H)$	**.66** (+1%)	**.59**** (+5%)	**.66**(+10%)
$CoOcc(e,H)$	.62 (−5%)	.57 (+1%)	.65** (+8%)

exploit the past to improve as the size of H grows. That is, the more history is available the better we can rank entities for the current news.

The y-axis of Figure 5.5 plots the average MAP for all the documents with history size $|H|$ using the feature $F(e,H)$. For $|H| < 20$ the effectiveness of $F(e,H)$ increases together

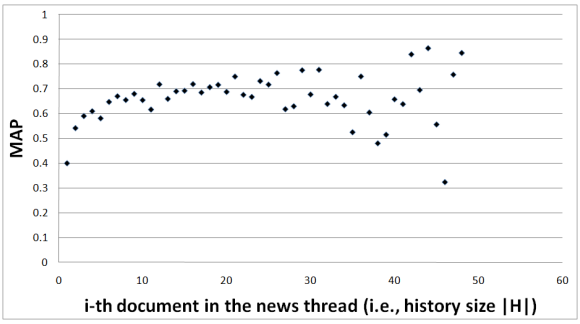


Figure 5.5 Mean Average Precision values for documents having a certain history size.

with $|H|$ up to values of 0.7. Results for higher values of $|H|$ show no clear trend due to the fact that there are just a few datapoints.

Feature combination

So far we have presented different features for ranking entities that appear in a document. Combining them in an appropriate manner yields a better ranking of entities; however, because the distribution of relevance probability is different among features, we need a way for combining them. The following experiments rank entities in a document according to a score obtained after combining several features together. We consider linear combination of features (transformed with a function as explained in [51]). Finally, we will consider a combination of all the features using machine learning techniques.

Linear Combination of Features Let the score for an entity e and a vector $\vec{f}$ of n features be

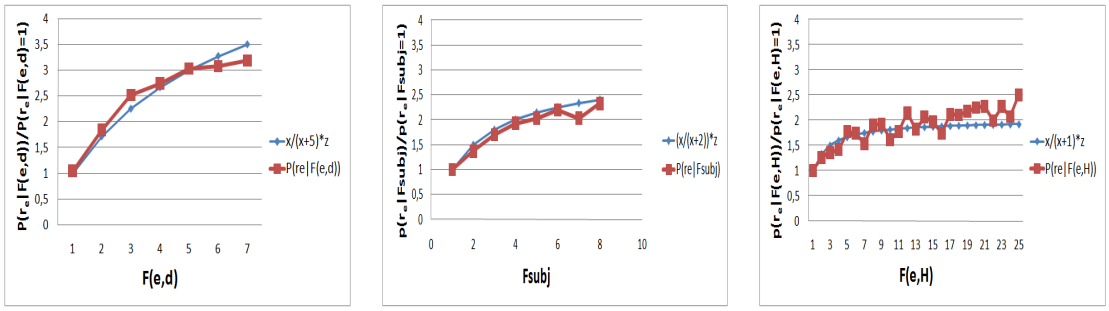
$$score(e,\vec{f}) = \sum_{i=0}^{n-1} w_i g(f_i,\theta_i)\ , \tag{5.1}$$

where w_i is the weight of each feature and g is a transformation function for the feature f_i using a given parameter θ_i. Since we are only interested in the ranking we can eliminate one weight parameter by fixing $w_0 = 1$ [51]. In this thesis we employ a transformation function of the form:

$$g(x,\theta) = \frac{x}{x+\theta} \tag{5.2}$$

as suggested in [51], where x is the feature to transform and θ is a parameter. We also tried a linear transformation but it did not perform as well. More complex non-linear transformations could also be explored.

In order to combine features we then need to find a parameter θ_i for the function g and a weight w_i for each feature f_i. In Figure 5.6 we show how some of the functions we employed fit the distribution of probability for different features. The probability values are normalized in a way that the plot starts from the point $(x = 1, y = 1)$. The same is done for the g function using a multiplicative constant $z = (1+k)$.



(a) (b) (c)

Figure 5.6 Normalized probabilities of an entity being relevant for a given feature value and the selected g function normalized with a constant z.

We tested two and three features combinations, where the variables θ_i, and the combination weights w_i have been tuned with 2-fold cross validation of 25 topics training to optimize MAP. In order to find the best values we used an optimization algorithm that performs a greedy search over the parameter space [140]. Figure 5.7 presents MAP obtained for different values of the combination weight w_1 when different features are combined with $F(e,d)$. In some cases the combination performs better than the original baseline with the best performing features being robust to all values of w. Features from the local document such as F_{subj}, $FirstSenLen$, and $FirstSenPos$ show performance improvements only for small combination weights whilst features from the history have a higher robustness to high values of w. The two features that looks at individual documents in the history

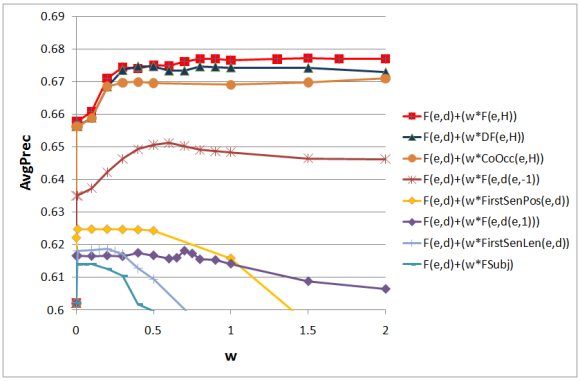


Figure 5.7 Mean Average Precision values for different values of w when combining features with $F(e,d)$.

Table 5.5 Effectiveness of two features combined with $F(e,d)$.* (**) indicates statistical significance w.r.t. $F(e,d)$ with p¡0.05(0.01). †(††) indicates statistical significance w.r.t. $F(e,H)$ with p¡0.05(0.01).

f_1, f_2	P@3	P@5	MAP
$F(e,d_{e,-1})$ $F(e,H)$	**.70****††(+8%)	**.62****††(+11%)	**.69****††(+15%)
$CoOcc(e,H)$ $F(e,H)$	.69**††(+6%)	.62**††(+11%)	.68**††(+13%)

($F(e,d_{e,-1})$ and $F(e,d_{e,1})$) decrease their performance as w increases. On the other hand, features looking at the entire set of past documents H are most robust.

Table 5.7 summarizes the results for all the features using 2-fold cross validation. Combining $F(e,d)$ with another feature is able to outperform the baseline for some range of the weight w that can be learned on a training set. For some features (*AvgBM25s*, *SumBM25s*) the original baseline score is not improved by the combination. The best effectiveness is obtained when combining $F(e,d)$ and $F(e,H)$ obtaining an improvement of 13% in terms of Mean Average Precision. Other features, when combined with the baseline, also obtain high improvements performing as good as the combination with $F(e,H)$ ($CoOcc(e,H)$ having 12% and $DF(e,H)$ having 13% improvement in terms of MAP). The feature $F(e,d_{e,1})$, which is poorly performing as individual feature (see Table 5.4), obtains a limited improvement of 2% in terms of MAP. These results also hold for early precision measures.

In order to combine three features we need to find suitable values for two different weights w_1 and w_2 (we tune parameters and report the performance using 2-fold cross validation). The results for two different combinations of features with $F(e,d)$ are presented in Table 5.5. Results show that combining the baseline with two features from the history we can reach an improvement of 15% in terms of MAP over the baseline.

Table 5.6 Effectiveness of two features combined with $F(e,d)$ using logistic regression. The list of features in presented in Tables 5.3 and 5.4. In brackets the % improvement over $F(e,d)$. * (**) indicates statistical significance w.r.t. $F(e,d)$ with p¡0.05(0.01). †(††) indicates statistical significance w.r.t. $F(e,H)$ with p¡0.05(0.01).

Features	P@3	P@5	MAP
Local	.65 (±0%)	$.58^{*}$ (+4%)	$.63^{**}$ (+5%)
History	.65 (±0%)	$.60^{**}$ (+7%)	$.66^{**}$ (+10%)
All	$\mathbf{.70}^{**\dagger\dagger}$ (+8%)	$\mathbf{.63}^{**\dagger\dagger}$(+12%)	$\mathbf{.69}^{**\dagger\dagger}$(+15%)

Using Machine Learning for combining features In order to combine two or more features together we used machine learning techniques. We performed 2-fold cross validation training a multinomial logistic regression model with a ridge estimator [107] with default parameters for ranking entities in each document. Results show that when combining any feature with $F(e,d)$ using logistic regression the results are comparable to those obtained with manual tuning (see Table 5.7).

Table 5.6 presents a combination of every local and history feature. The combination

Table 5.7 Effectiveness of features when combined with $F(e,d)$. Bold values indicate the best performing runs. In brackets the % improvement over $F(e,d)$.* (**) indicates statistical significance w.r.t. $F(e,d)$ with p¡0.05(0.01). †(††) indicates statistical significance w.r.t. $F(e,H)$ with p¡0.05(0.01).

Feature	P@3	P@5	MAP
FirstSenLen	.65 (±0%)	$.57^{*}$ (+2%)	$.62^{**}$ (+3%)
FirstSenPos	$\mathbf{.67}^{**}$ (+3%)	$.58^{*}$ (+4%)	$.62^{**}$ (+3%)
FirstSenPosTrans	$.67^{**}$ (+3%)	$\mathbf{.58}^{**}$ (+4%)	$\mathbf{.64}^{**}$ (+7%)
F_{subj}	.65 (±0%)	.56 (±0%)	.61 (+2%)
AvgBM25s	.65 (±0%)	.56 (±0%)	.60 (±0%)
SumBM25s	.65 (±0%)	.56 (±0%)	.60 (±0%)
$F(e,d_{e,1})$	.65 (±0%)	$.57^{**}$ (+2%)	$.61^{**}$ (+2%)
$F(e,d_{e,-1})$	$.68^{**\dagger}$ (+5%)	$.60^{**}$ (+7%)	$.65^{**}$ (+8%)
$F(e,H)$	$\mathbf{.70}^{**\dagger\dagger}$ (+8%)	$\mathbf{.62}^{**\dagger\dagger}$ (+11%)	$\mathbf{.68}^{**\dagger\dagger}$ (+13%)
$CoOcc(e,H)$	$.68^{**\dagger\dagger}$ (+5%)	$.61^{**\dagger\dagger}$ (+9%)	$.67^{**\dagger\dagger}$ (+12%)
$DF(e,H)$	$.69^{**\dagger\dagger}$ (+6%)	$.61^{**\dagger\dagger}$ (+9%)	$.68^{**\dagger\dagger}$ (+13%)

of local features performs better then the baseline and then most of the single local features (see Table 5.3). Finally, when all the features are combined (local+history) we obtain the best effectiveness which is anyway not better than the combination of the three best features, i.e., $F(e,d)$, $F(e,d-1)$, and $F(e,H)$ (see Table 5.5). Such improvements are anyway negligible if compared with the best 2 features combination, that is, $F(e,d)$ and

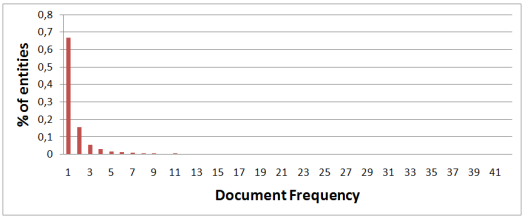


Figure 5.8 Entities with given document frequency in the topic.

$F(e,H)$ obtaining a MAP of 0.68. Therefore, we can see how these two simple features perform very well and that it is difficult to improve over such approach.

5.3.5 Building Entity Profiles

In this section we present an initial discussion about an additional task (i.e., the Entity Profiling task) providing some statistics on the test collection we have built.

In a search interface, we may wish to show to the user relevant entities in the entire document history and not just entities from the current document. This prompts a second task definition:

- Entity Profiling (EP): Given a query and the set of related documents over time, create for each entity a plot showing the temporal development of entity relevance (i.e., which entities become relevant and which become not relevant over time). This is related to new user interfaces being proposed in commercial systems[7] and can help the user understanding which are the key entities in the story even if they do not appear in the news article she is reading.

Given that for a single event (a topic in the TREC collection) there are many entities (31 documents per topic and 27 entities per document) appearing, an important question is: for which entities should we build and show a profile? Figure 5.8 presents the distribution of document frequencies for entities, where 67% of entities appear only in one document. For such entities it does not make sense to build a time-based profile as there is no evolution of their relevance.

As we have already stated, relevant entities tend to stay relevant. It is therefore also not interesting for the user to see entity profiles which are flat, that is that do not change over the story line. In Figure 5.9 we can see that half of the entities which are relevant at least once are relevant any time they appear. We would therefore build entity profiles on entities which are relevant at least once and which do not have always the same judgement. There are 708 entities like this over the 25 topics in the collection.

[7] http://correlator.sandbox.yahoo.net/, http://entitycube.research.microsoft.com, http://newstimeline.googlelabs.com/

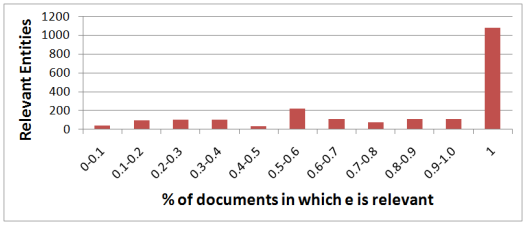


Figure 5.9 Duration of relevance for entities relevant at least once.

A system has then to decide for each entity and each day when it appears in news whether to trigger ("ON") the entity or not ("OFF"). In order to evaluate such system we can define a true positive as the situation where the entity is relevant and the system returns "ON", true negative when the entity is non-relevant and the system returns "OFF", false positive when the entity is non-relevant and the system returns "ON", and false negative when the entity is relevant and the system returns "OFF". A system answering always ON will then get Precision of 0.56 and Recall of 1. A system exploiting the history and answering ON for entities appearing more than once in the current document or having $F(e,H) > (|H|/t)$ would get Precision of 0.60 and Recall of 0.54 for $t = 5$. Such result shows that a simple baseline performs well.

Focus of our future work will also be an alternative to the Entity Profiling task. We imagine queries of the type: *'Will entity e be relevant in the future?'*. The task can be defined as predicting appearance (and relevance) of an entity e in future documents given that 1) e has appeared in the past (as relevant) and 2) e does not appear today. Analysing relevance assessments we can see that 7% of entities appear at least twice as relevant with a gap (i.e., they do not appear in a particular day) in between. Being able to predict entity relevance would enable retrieval systems to extend the produced TAER resultset including entities which are not present in the current news article which are anyway important for the overall story.

5.4 Mining Opinion about Entities on the Web

In the last years, the blogosphere has become a vital part of the Web, covering a variety of different points of view and opinions on political and event-related topics such as immigration, election campaigns, or economic developments. Tracking the public opinion is usually done by conducting surveys resulting in significant costs both for interviewers and persons consulted. In this section, we propose a method for extracting political trends in the blogosphere automatically.

One possible scenario is the USA 2008 presidential election where both the Democratic and the Republican party proposed one candidate for the White House. In that case, Barack Obama and John McCain were facing each other during several month in 2008. Until November both of them were appearing in news, television shows and in public conventions. Their actions and talks triggered a lot of opinions in the general audience and,

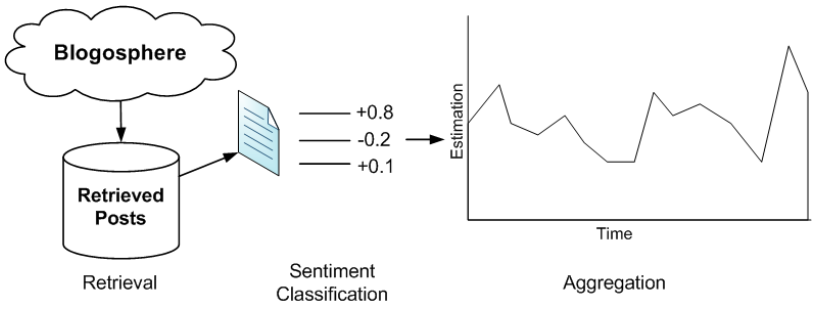


Figure 5.10 System Overview

consequently, many Web users were expressing their opinions and thoughts in blogs.

This example shows that evidence of public opinion may be available in the blogosphere motivating our attempt to mine Web content in order to estimate the public opinion about political elections. To this end, we apply sentiment and time series analysis techniques in combination with aggregation methods on blog data to estimate the temporal development of opinions on politicians. We consider both purely data-driven, unsupervised settings, as well supervised scenarios where a limited amount of traditional poll results is used to learn parameters for better matching blogosphere and "real world". Our experiments focus on election scenarios using professional opinion polls as ground truth, and show promising results towards estimating trends for political opinions from the blogosphere.

5.4.1 Extracting opinions from blogs

In this section we present our methods for extracting bloggers' opinions on two target entities (O and M) and aggregating them over time. Identifiers O and M are motivated by our example of the US 2008 election campaign and the two main rival candidates Obama and McCain; however, our described methods are not specific for that scenario, and can be easily generalized to situations with two or more concurrent entities.

The approach we propose for estimating the public opinion about politicians consists of three main steps: 1) retrieval, 2) sentiment classification, and 3) aggregation (see Figure 5.10). In the first step we use simple IR techniques for retrieving blog postings to gather data about the candidates of interest. Then, in order to extract the sentiment expressed about the candidates in such postings we exploit a lexical sentiment resource and text classifiers. Finally, we aggregate the computed scores over time for estimating the temporal development of the public opinion.

We first describe opinion analysis in an unsupervised setting where the only information available comes from the blogs. For that setting, we elaborate on different techniques for extracting sentiment from blogs and for aggregating the resulting values. Then, we show how parameters of the aggregation and prediction models can be learned in a supervised setting where an additional small history of training data in form of poll results is available.

Unsupervised Techniques

Retrieving Relevant Postings From the blogosphere we first retrieve a set $P = \{p_1, \ldots p_n\}$ of blog postings relevant to entities O and M. In the Obama vs. McCain example we were conducting this based on simple keyword search, assuming that all postings containing at least one of the candidates names were potentially useful for our further analysis. For more complex and ambiguous topics more enhanced techniques based, for instance, on Latent Semantic Analysis or NLP can be necessary in this preprocessing step. For example, components for extracting [80, 81] and globally identifying entities [33] could be plugged into our system in case that entity names are highly ambiguous, or the entities frequently occur under different names. The retrieval step is not the main focus of our thesis, though.

Assigning Sentiment Scores Simple counting of postings about the entities of interest does not take opinions into account, and is, thus, insufficient for highly polarizing topics such as election campaigns. Instead, we aim to assign sentiment values $s(p)$ to blog postings p. For our scenario with two concurrent entities O and M, we assume that $s(p)$ lies in the interval $[-1,+1]$ with $s(p) = 1$ referring to a maximum positive opinion expressed about O, and, conversely $s(p) = -1$ corresponding to a totally positive opinion on the competing entity M. How can we obtain these sentiment values for blog postings?

Sentiment Thesaurus-based Approach The first option is to exploit a lexical resource for opinion mining such as SentiWordNet [79]. SentiWordNet is built on top of WordNet [84], a thesaurus containing textual descriptions of terms and relationships between them. In SentiWordNet a triple of three *senti values* (pos, neg, obj) (corresponding to positive, negative, or rather neutral sentiment flavour of a word respectively) are assigned to each set of synonymous words w in WordNet. Sentivalues in SentiWordNet that were partly assigned by human assessors and partly automatically, are in the range of $[0,1]$, and sum up to 1 for each triple. For instance $(pos, neg, obj) = (0.875, 0.0, 0.125)$ for the term "good" or $(0.25, 0.375, 0.375)$ for the term "ill". We define $Sent_O(p)$ as the set of sentences in a posting p which contains entity O but not M and we analogously define $Sent_M(p)$. Let $Lex(st)$ be the set of words w from sentence st that can be assigned a sentiment value based on the available lexicon. For a sentence st we aggregate positive and negative sentivalues for opinionated words as follows:

$$senti_e(st) = \frac{\sum_{w \in Lex(st)} pos(w) - neg(w)}{|Lex(st)|}, e \in \{O, M\} \tag{5.3}$$

where, $pos(w)$ and $neg(w)$ are the positive / negative values for the word w in the lexical resource, and $st \in Sent_e(p)$.

We can now compute the sentiment of the posting p about each of the two considered entities independently. Thus, we define:

$$s_e(p) = \frac{\sum_{st \in Sent_e(p)} senti_e(st)}{|Sent_e(p)|}, e \in \{O, M\} \tag{5.4}$$

Such defined estimators allow us to follow opinion trends about each candidate individually. The values of these estimators lie in the interval $[-1,1]$ where -1 indicates a strong negative opinion and $+1$ a strong positive opinion about the candidate e.

Then we compute the overall sentiment score for posting p as

$$s(p) = \frac{s_O(p) - s_M(p)}{2} \tag{5.5}$$

Note that sentiment values for O and M push the overall score into opposing directions accounting for the competitive relationship between these entities.

The generalization to the scenario with m competing entities $E = \{e_1, \ldots, e_m\}$ would require $s(p)$ to be a point in a $(m-1)$-dimensional space. For example, in the case of 3 competing entities, $s(p)$ can be represented as a point inside a triangle having as vertexes the entities. In our setting, the estimator $s(p)$ assumes values in $[-1,1]$ where -1 indicates a preference towards M while $+1$ indicates a preference for O.

Sentiment Classification Approach As second option for computing the sentiment of bloggers with respect to given entities we describe the application of machine learning, more specifically, text classification on a sentence level.

Linear support vector machines (SVMs) construct a hyperplane $\vec{w} \cdot \vec{x} + b = 0$ that separates a set of positive training (corresponding to positive opinions in our case) examples from a set of negative examples (negative opinions) with maximum margin. For a new previously unseen sentence $\vec{st}$, the SVM merely needs to test whether it lies on the "positive" side or the "negative" side of the separating hyperplane. The decision simply requires computing a scalar product of the vectors $\vec{w}$ and $\vec{st}$. In this thesis, we use a standard Bag-of-Words representation of sentences with TF weighting and an SVM classifier trained on both a sentence polarity dataset consisting of 10,662 movie reviews [133] as well as on manual judgements from the polarity task at the TREC 2008 Blog Track [130] consisting of 18,142 opinionated blog postings. As shown in [74], training on a combination of documents from different domains can achieve good results in the context sentiment classification.

For SVMs a natural confidence measure is the distance of a test sentence vector from the separating hyperplane (with positive or negative sign depending on its position relative to the hyperplane). After linearly normalizing classification scores for a sentence st to be in the range $[-1,1]$, we compute the sentiment values for posting p with respect to entities O and M as described in equation 5.4. These values are then combined to a sentiment value $s(p)$ as described in equation 5.5.

Sentiment Aggregation How can we compute an aggregated temporal development of opinions in the blogosphere? For a time interval $t = [t_1, t_2]$ we consider all postings $P_t = \{p \in P : t_1 \leq TS(p) \leq t_2\}$ relevant to entities O and M published in that interval, where $TS(p)$ is the timestamp of posting p. We then use data extracted from this set to estimate an opinion value $poll(t) \in [-1,+1]$ for time interval t. Values $poll(t) = 1$ and -1

would correspond to a situation where the whole population would be pro entity O or M respectively.

Given the sentiment scores $s(p)$ for postings obtained as described in the previous section we want to estimate $poll(t)$ through aggregation. Let Sel be a function for selecting the subset $Sel(P_t) = P'_t \subseteq P_t$ we want to consider for aggregation, and $f : [-1,+1] \to [-1,+1]$ be a monotonically increasing function with $f(-1) = -1$ and $f(1) = 1$. Then, we can compute an aggregated value as follows:

$$poll(t) = \frac{1}{|Sel(P_t)|} \sum_{p \in Sel(P_t)} f(s(p)) \tag{5.6}$$

describing, in general form, an increase of the estimate $poll(t)$ with an increasing number of high sentiment values $s(p)$ (and, analogously, a decrease in case of many negative sentiments).

How to choose appropriate instances of f and Sel?
The simplest instantiation of such an aggregation is an *Averaging model* with all postings selected and no transformation of scores (i.e., f being the identical function):

$$poll(t) = \frac{1}{|P_t|} \sum_{p \in P_t} s(p) \tag{5.7}$$

Alternatively, we can apply a *Counting model* using thresholds on the sentiment scores:

$$Sel(P_t) = \{p \in P_t | s(p) < thres_1 \vee s(p) > thres_2\} \tag{5.8}$$

where $thres_1$ and $thres_2$ are thresholds used for discarding objective postings for which there is no clear sentiment assigned. Scores can then be transformed into explicit binary "votes" to be counted:

$$f(x) = \begin{cases} 1 & \text{if } x > thres_1 \\ -1 & \text{if } x < thres_2 \\ 0 & \text{otherwise} \end{cases} \tag{5.9}$$

These votes are then averaged over as described in equation 5.6.

Adjusting Aggregated Opinion Estimates

Although blogs provide a rich source of information, the assumption that blog articles fully reflect the opinion of the overall population is rather unrealistic. In standard election polls a major effort is spent on constructing a sample that properly represents the entire population. In our setting we can obtain a much bigger sample which may be biased in some ways, though. We want to address several of the resulting issues, and introduce model parameters to adjust the poll estimation function $poll(t)$ described above:

- The first issue to account for are publishing delays: People can express their opinions in blogs at any point in time while phone interviews are carried out, their outcome is processed, and, finally, results are published, resulting possibly in delays compared to the blogosphere. In order to address the delay issue, we introduce a *lag* factor that shifts $poll(t)$ earlier or later in the timeline.

- Although blog writing technology has become more and more accessible to a broad range of people, users writing blogs and choosing to publish their opinion are not necessarily representative for the whole population. To account for this difference we introduce an additive *bias* constant to our model.

- Finally, sentiment values computed as described in the previous section might require some re-scaling in order to reflect the actual "strength" of the opinions. We therefore introduce a constant multiplicative factor (*scale*) for the produced estimate.

Considering all these factors we transform poll estimation for a time interval t in the following way:

$$poll(t, lag, bias, scale) = (poll(t + lag) + bias) \cdot scale \tag{5.10}$$

To account for general noise in the data we introduce a *smoothing* function over the estimates. In this thesis we apply a simple moving average over past estimates

$$poll(t, k) = \frac{\sum_{j=0}^{k-1} poll(t - j)}{k} \tag{5.11}$$

where k is the number of past time intervals considered for smoothing. Other possible smoothing techniques include alternative moving average functions or curve fitting techniques such as spline interpolation.

How to obtain the mentioned parameters? Our experiments in Section 5.4.2 show already surprisingly good results for quite simple and obvious default configurations. Otherwise, some domain knowledge might help to manually adjust some of the parameters. In case additional information on polls is given we can learn parameters as described in the below section.

Supervised Techniques

In addition to the blog information a history of traditional opinion poll values might be available, resulting in a supervised scenario in which parameters can be learned, and temporal developments can be interpolated.

We *optimize parameter values* introduced in the previous section by minimizing the average root mean squared error of $poll(t, lag, bias, scale)$ compared to the values obtained from poll results of the given "training" history. In detail, we learn the best parameter values using the Nelder and Mead simplex approach [127], an algorithm for minimizing an

objective function (i.e., the error) in multi-dimensional spaces. Specifically, the objective function is the following:

$$\underset{lag,bias,scale,k}{\operatorname{argmin}} \sqrt{\frac{1}{n}\sum_{t=1}^{n}(p(t,lag,bias,scale,k)-gt(t))^2} \tag{5.12}$$

The method builds a n-dimensional triangle (i.e., a simplex) and compares function values at the n vertexes of the simplex. The worst is rejected and replaced with a new vertex that creates a new simplex. Iteratively, function values become smaller until a minimum is found. Of course, there is no guarantee of finding the global minimum as the algorithm may converge to a local one. Our experiments show that applying this learning step can lead up to a 39% improvement compared to the unsupervised approach.

Furthermore, we apply *Time Series Analysis (TSA)* techniques in order to predict the continuation of a data series of opinion polls. In this thesis, we use linear forecasting [166] and polynomial regression [47] for predicting values $pTSA(t)$ of the poll at time t which can, for instance, be linearly combined with the blog data-driven prediction $poll(t)$:

$$poll_{TSA+blog}(t) = \lambda \cdot poll(t) + (1-\lambda) \cdot pTSA(t) \tag{5.13}$$

where λ is a tuning parameter for controlling the influence of one or the other type of prediction, and which might be learned, e.g., on a held-out subset of the training history.

5.4.2 Experimental Results

In this section we first describe and analyse the dataset we use for evaluating the proposed models in the context of the US 2008 election campaign scenario. We then describe our experimental setup, do a preliminary evaluation of individual subcomponents (retrieval and sentiment classification), and finally show qualitative and quantitative results illustrating the potential of our opinion aggregation approach.

Scenario and Data

In this thesis we focus on the scenario of the political campaign for the US 2008 presidential election held on November, 4th 2008. We analyse its two main rival candidates Obama and McCain. For computing opinion estimates we applied our methods to the TREC 2009 Blog Track dataset[8] which is composed of 1,303,520 feeds for a total of 28,488,766 permalink documents (i.e., blog postings) posted from the 14th January 2008 to the 10th February 2009. Using a language classifier [37] we estimated that about 65% of the postings are in English. Building an inverted index on the English postings, we retrieved 670,855 relevant to the disjunctive query "obama OR mccain" which formed our working set. Figure 5.11 shows the amount of retrieved postings over time. In general, there is more content available about Obama rather than about McCain, and most of the postings about McCain also

[8] http://ir.dcs.gla.ac.uk/test_collections/blogs08info.html

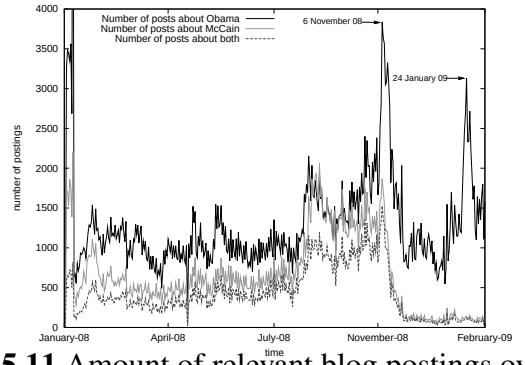


Figure 5.11 Amount of relevant blog postings over time.

talk about Obama. There is a growing amount of data as the election date comes closer; in particular, we can observe large peaks close to the inauguration date (January, 20th) and the election date (November, 4th).

As *ground truth* for opinion polls we used the data provided by Gallup[9], a professional service, tracking public opinion by means of telephone polls. This data set provides results of interviews with approximately 1500 US national adults conducted over the time period from March until November 2008 for a total of 230 polls. The value for each time interval represents the percentage of registered voters who would support each candidate if the presidential election was held on the day of the interview. Polls are based on a five-day moving average until June 8th and a three-day average starting at June 9th. We additionally normalized the ground truth so that for each time interval the values for the two candidates sum up to 100%. This simplification allows us to disregard the difficult problem of the undecided voters. Undecided voters are difficult to identify by mining the blogosphere as both non-topical (e.g., about sport) and non-opinionated postings (i.e., neither positive nor negative) are not indicators of indecision as they just do not contain enough information.

Preprocessing and Setup

After obtaining the set of relevant postings we processed them using state-of-the-art techniques for Web page template removal [104] in order to restrict our analysis to the actual content of the postings and to avoid non-informational surroundings, e.g., advertisements or navigational links. After this initial preprocessing step, we used NLP tools[10] to split postings into a set of sentences and to POS-tag them. On the resulting output we did lookups in SentiWordnet [79] to obtain positivity and negativity scores for each adjective in a sentence. For the machine learning-based sentiment score assignment, preprocessed sentences were categorized using the SVMlight [100] implementation with default paramaterization to obtain positive or negative scores (see Section 5.4.1).

The models proposed in Section 5.4.1 return, for a given time interval, estimate scores

[9] http://www.gallup.com/
[10] http://gate.ac.uk/

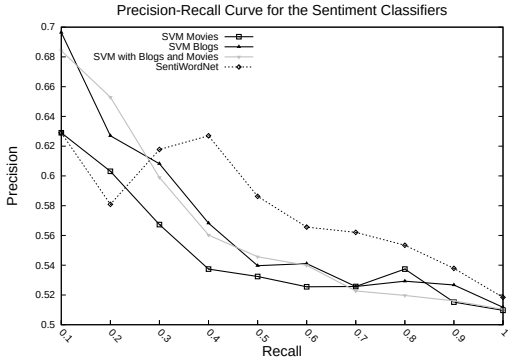


Figure 5.12 Precision-Recall curves for sentiment classification methods.

between -1, indicating a completely pro-McCain estimate, and 1, indicating a completely pro-Obama estimate. Values close to 0 indicate an almost equal estimation for the two target entities. We therefore normalized our ground truth accordingly in the interval [-1,1]. In order to evaluate the effectiveness of the proposed models, we computed the Root Mean Squared Error (RMSE) between the estimation and the true poll value:

$$RMSE(p,gt) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(p_i - gt_i)^2} \tag{5.14}$$

where p_i is the estimate and gt_i is the poll value for the i-th time interval.

In the *unsupervised* setting (see Section 5.4.1), the task was to estimate the temporal development of opinions given only the blog data while for the *supervised* setting (see Section 5.4.1) we assumed that a history of past polls was given. Experiments presented in this thesis use the first 50% of the ground truth for learning model parameters and the remaining 50% for evaluating the models.

Evaluation of Individual Components

As different components are involved in the overall estimation process (see Section 5.4.1), we performed a user evaluation of the individual modules.

In order to evaluate the quality of the posting selection step we manually judged a random sample of 200 retrieved blog postings for relevance to the topic of the considered election campaign. In this way, we observed a precision value of 0.79 which, due to the large amount of data matching our queries (see Figure 5.11), is sufficient for obtaining a reasonable set of topic-related postings.

We evaluated the sentiment estimations computed by SentiWordNet as well as by three different SVM classifiers trained using respectively the movie review dataset [133], the TREC 2008 Blog Track [130] judgements, and both together. In order to evaluate the quality of the different sentiment classification techniques we constructed a sample consisting of 500 sentences randomly sampled from the top 10% sentences ordered by the classifiers

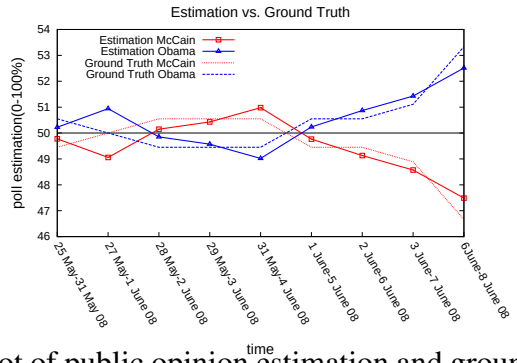


Figure 5.13 Snapshot of public opinion estimation and ground truth opinion polls for an interval around the event "Obama becomes presidential nominee" on June 3rd, 2008.

score and of 500 sentences from the bottom 10%. On such sample we computed Precision scores at different Recall levels obtaining the curves shown in Figure 5.12. From the results we can see that the SVM classifier performs better when using training data from opinions expressed in the blogosphere. Moreover, we see that the approach based on SentiWordNet, even if worse than others for low recall levels, generally performs best.

Qualitative Example

In this section, we provide an analysis of a key event in the campaign and its effect on the public opinion, and show opinion estimates near the time of the event. As many different events cause changes in public opinion, it can be insightful to study specific peaks (i.e., high values of public opinion for one or the other entity) and opinion switches (i.e., changes between a positive/negative poll value). One example of such an event is the internal election of the Democratic party representative featuring as candidates Hillary Clinton and Barack Obama. On June 3rd, 2008, Obama was elected presidential nominee for the Democratic party. Figure 5.13 shows a snapshot of the estimation provided by the unsupervised counting model using a lexical approach for obtaining sentiment scores (see Section 5.4.1). We can see that after the event Obama attains a clear advantage over McCain which is reflected both by the polls and our estimates. Table 5.8 shows example passages extracted from postings published close to the event. We observe that the Obama / Clinton discussion is ending and that the focus shifts towards the Obama / McCain rivalry. An example on the entire timeline of the estimation performed by an unsupervised model is shown in Figure 5.14.

Quantitative Experiments

In this section, we quantitatively measure and compare effectiveness of different methods for estimating public opinion about political elections. We want to examine whether our

Table 5.8 Example passages connected to the event 'Obama becomes presidential nominee'.

Date	Text
June, 5th	`Democratic women are not going to vote for McCain and and throw Roe on the funeral pire of Hillary campaign.`
June, 5th	`Everyone in the media says Hillary will endorse Obama this week, perhaps even with an appearance together.`
June, 6th	`Obama is a master at rousing his audience's emotions.`
June, 6th	`Picture the contrast between them: Obama, young and vibrant, exhorting us to answer the better angels in our nature; McCain, old and pinched, promising a Hundred Years of War and, by the way, the jobs ain't coming back.`

sentiment aggregation methods are useful for predicting public opinion trends and whether supervised techniques as well as TSA methods can help at this task.

We first measured the quality of the purely data-driven unsupervised models and aggregation techniques suggested in Section 5.4.1:

- **LexAvg**: lexicon-based model using SentiWordNet to compute sentiment of postings and the Averaging aggregation model
- **ClassifyAvg**: classification-based model using an SVM classifier trained on movie reviews and blog postings and the Averaging aggregation model
- **LexCount**: lexicon-based model using SentiWordNet and the Counting aggregation model
- **ClassifyCount**: classification-based model using an SVM classifier trained on movie reviews and blog postings and the Counting aggregation model

For these techniques, we also test the performance of their supervised counterparts (see Section 5.4.1) where parameters are learned individually on the initial 50% of the data. Moreover, we compare against Time Series Analysis methods that only exploit past available ground-truth to estimate future trend of opinions (see Section 5.4.1):

- **LinFor**: a linear model based on forecasting techniques (we use the Hunter's simple exponential smoothing forecast model [166])

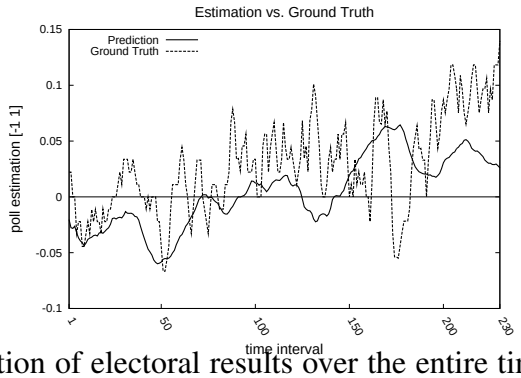


Figure 5.14 Estimation of electoral results over the entire timeline using the unsupervised Counting model, lexicon based sentiment estimation (LexCount), and smoothing by moving average over 15 intervals.

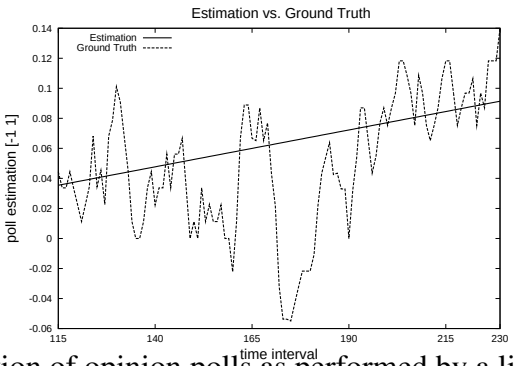


Figure 5.15 Estimation of opinion polls as performed by a linear forecaster (LinFor).

- **LinReg**: a linear regression model to fit a deterministic trend to available training data [47]
- **QuadReg**: a polynomial regression model of degree 2 to fit a deterministic trend to available training data [47]

Finally, we compute the linear combination of our best supervised approach, **LexCount**, with best the TSA model, **LinFor**, and we note it **LexCountLinFor**.

In addition, we compare the proposed techniques with a very simple baseline just taking into account the plain number of retrieved postings for each candidate in a time interval. More specifically, we compute $poll(t) = \frac{o+om}{o+m+om} - \frac{m+om}{o+m+om}$ where o indicates the number of postings in t about Obama, m about McCcain, and om about both (**Count1**) and another variant where the postings containing both keywords are not taken into account (**Count2**).

Table 5.9 shows the RMSE values for all of the compared approaches[11]. Figures 5.14, 5.15 and 5.16 show the detailed temporal developments for the best unsupervised, time

[11] Experiments show that the best thresholds for the counting model described in Section 5.4.1 are (0,0), that is, taking the sentiment of all postings into consideration.

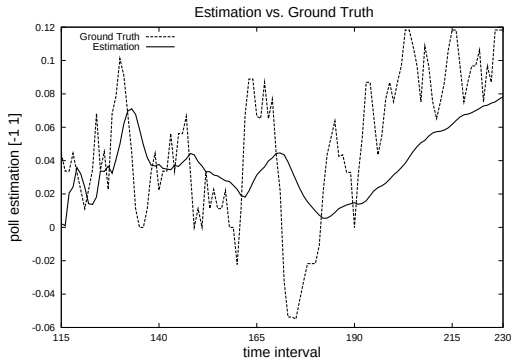


Figure 5.16 Estimation preformed by the combination of a linear TSA model with our supervised Counting Model (LexCount).

series-based, and best overall approach (supervised data-driven learning linearly combined with time series analysis) respectively; the ground truth using traditional polls from Gallup is shown as dotted line. Finally, the detailed estimations for all of the remaining methods are shown in Figure 5.17 in the appendix. The main observations from our experiments are:

- All methods using sentiment-based blog analysis approximately capture trends, peaks, and switches in public opinions. Even purely data-driven unsupervised methods using aggregated sentiments extracted from the blogosphere can already estimate public opinion about electoral candidates quite well (RMSE = 0.0572 for **LexCount**). Figure 5.14 further illustrates the ability of unsupervised **LexCount** to match opinion changes.

- The supervised counterparts of sentiment-based methods that exploit the history of past opinion polls improve the quality of the up to 39% (RMSE = 0.0499 for **ClassifyCount**) compared to the unsupervised approach by tuning model parameters that take into account the intrinsic sample bias (see Section 5.4.1). Learned parameters are lag, taking into account time delays in estimations, smoothing to reduce noise into the underlying data, bias to tackle the problem of a non-representative sample, and scale to either amplify or reduce the estimation to better fit the ground truth.

- The combination of Time Series Analysis techniques with supervised models learned on the available data (**LexCountLinFor**) shows the best performance of all approaches resulting in overall improvement of 39% compared to the best unsupervised approach (**LexCount**) and of 69% compared to the best approach based on posting volume (**Count1**) (see also Figure 5.16 for the temporal development).

Simply counting postings published about a candidate over time results in a comparatively large estimation error (with RMSE = 0.1272 and 0.3422 for **Count1** and **Count2** respectively), and, thus, is not suited for the opinion estimation problem (see Table 5.9).

Table 5.9 RMSE values for the estimation models on the last 50% of the timeline. The initial 50% are used for training the model parameters of the supervised models. Learned parameter values for (lag, smoothing, bias, scale) are shown in brackets.

Method	RMSE	
Count1	0.1272	
Count2	0.3422	
	unsupervised	supervised
LexAvg	0.0628	0.0556 (1, 1, -0.0125, 1.0281)
ClassifyAvg	0.0616	0.0548 (0, 6, -0.0125, 1.0281)
LexCount	**0.0572**	**0.0483** (0, 15, -0.0314, 0.8159)
ClassifyCount	0.0813	0.0499 (6, 10, -0.034, 0.6332)
LinFor	0.0397	
LinReg	0.0405	
QuadReg	0.0999	
LexCountLinFor	**0.0394** (lambda=0.2)	

For both supervised and unsupervised settings, **LexCount** outperforms approaches based on sentiment classification using machine learning. The negative bias values shown in Table 5.9 indicate that, compared to the average US electorate, there is a higher preference for Obama in the blog dataset. Forecasting methods, while respecting the general trend of the public opinion, do not provide an indication of peaks and switches (see Figure 5.15), in contrast to our data-driven models. On the other hand, the combination of TSA with our models results in a low RMSE and better captures changes of the public opinion.

5.5 Discussion

In this chapter we have first addressed the problem of entity search and ranking in news streams. For this purpose, we defined an original entity search task and further created a time-stamped test collection for evaluating it.

We have produced an analysis of entity relevance in news topics. One of the conclusions is that determining the relevance of a sentence is very important to determine the relevance of an entity; more so than determining sentence novelty. In fact novel sentences introduce more entities than non-novel sentences, but many of these are not relevant. This is a counter intuitive finding, which challenges our view of novel sentences as introducing relevant entities. Our interpretation is that when an entity is first introduced (in a relevant and novel sentence), the reader cannot yet decide if the entity is truly relevant or not; only after repeated occurrences does the entity become relevant to the reader.

We have proposed features both from the current document and from previous ones in the document's history in order to find relevant entities in a given document. We have

experimentally shown that past frequency of entities is the most important of the features explored so far, more important than entity frequency in the current document another important feature. Position of the entity in the document (e.g., its first occurrence) is a weak indicator of its relevance, and it is specially difficult to use due to the different headers and introduction sentences present in different sources. We have tested several combinations of proposed features obtaining an overall statistically significant improvement of 15% in terms of Mean Average Precision over the baseline that considers the frequency of entities in the document.

Additionally, we have provided some preliminary observation on the Entity Profiling task concluding that an important challenge is the selection criteria of entities for which to build such profiles. As future work, besides testing our features on different time-aware document collections, we aim to develop and evaluate techniques for the Entity Profiling task.

In the second part of this chapter we proposed techniques for extracting public opinion about specific entities from the blogosphere. Today's blogosphere is widely used to express thoughts and opinions about political topics. In this chapter we presented an approach for estimating the development of public opinions over time by extracting and aggregating sentiments about politicians from the blogosphere. To this end, we combined techniques for sentiment analysis with different aggregation, parameter estimation, and forecasting techniques. Our experimental study in the context of the US 2008 election campaign showed that purely data-driven, unsupervised approaches can already capture trends, peaks, and switches for public opinions in the political domain. We achieved further improvements by utilizing, in addition to blog data, a history of past opinion poll results. In this scenario, learning parameters that capture intrinsic differences between blogosphere and the "real world" and combination of this approach with traditional Time Series Analysis techniques lead to more accurate estimates.

In our future work, we aim to enhance our approach towards more fine grained estimates taking into account various user characteristics such as gender, age, or location. Another interesting aspect is the influence of different electoral topics on opinions. Analysing evolution of topics over time using latent topic analysis methods, might allow for automatically inferring how newsworthy topics are correlated with political preferences. Similarly, we want to study events as a trigger of the public opinion (for instance, after the public reaction of a candidate, opinion about him may be subject to change). Moreover, we observed that error measures like RMSE fail in capturing changes and intermediate trends in temporal developments. We therefore want to explore alternative, more fine-grained metrics, that emphasize key time intervals for evaluating the effectiveness of opinion analysis approaches. Blogosphere characteristics can be subject to substantial variations due to technological developments or changing user demographics. Therefore, for the supervised scenario, it might be more appropriate to consider parameters such as lag and variance as dynamic variables. Furthermore, we aim at obtaining better estimations by integrating additional Natural Language Processing techniques for improving sentiment analysis and by performing more advanced associations between entities and sentiments. Finally, the

link structure of the blogosphere can be used for weighting opinions according to authority scores, or for adjusting opinion estimates by exploiting identified linked communities.

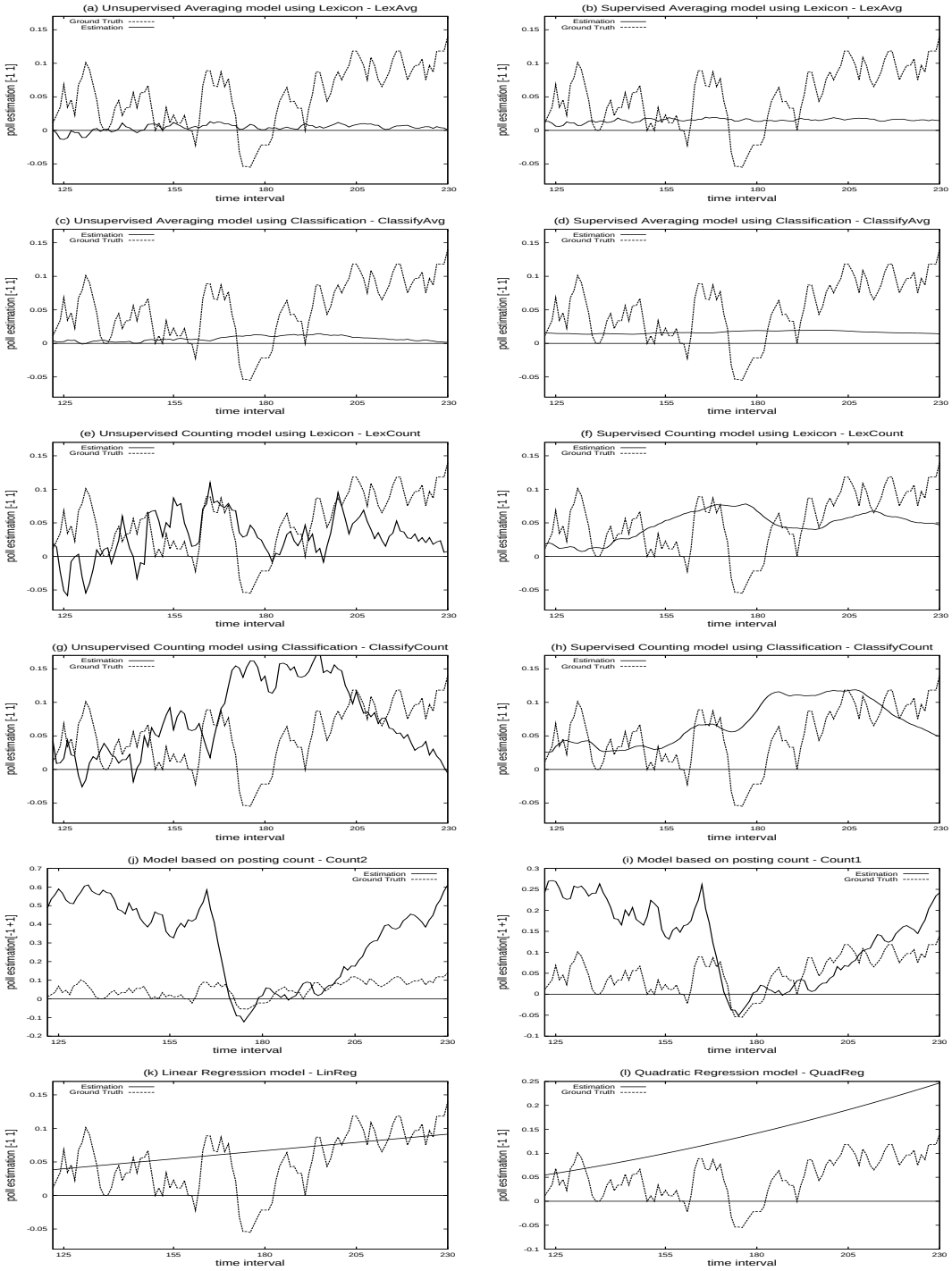


Figure 5.17 Estimations from methods described in Section 5.4.2.

6 Conclusions and Future Work

At the current status of the Web, the main entry points for people to the vast amount of content are Search Engines. Given a keyword query, current Search Engines instantly provide as result a list of links to Web pages. While this is a very good functionality for many types of queries, more complex tasks can not be easily solved by such presentation of the results.

Next generation retrieval systems will allow the user to perform complex tasks by better answering complex queries. In agreement with such vision, we deal with the task of Entity Retrieval which can be seen as a new functionality of Search Engines which, more than retrieving documents, will be able to retrieve entities.

In this thesis we contribute to the field of Information Retrieval by proposing methods for finding entities in document collections which are relevant to a user query. Such techniques could be applied, for example, for enriching current Search Engine results by showing the users more than just links to Web pages.

This final section summarizes our main contributions to the field and describe open problems to be taken into account in future investigations on the topic.

6.1 Summary of Contributions

In Chapter 3 we addressed the Entity Retrieval problem in the Enterprise setting. In detail, we first presented a model for ranking entities showing how it can be applied to different scenarios. We described a possible instantiation of the model and a set of techniques for the Expert Finding task. Experiments on standard test collections has shown that by combining the proposed approaches we achieve an improvement of 35% in terms of Mean Average Precision and of 53% in terms of Precision@10 over our baseline. The prototype application we developed shows the flexibility of the proposed Entity Retrieval model and how it is possible to apply it to any kind of scenarios where documents about entities are available.

Chapter 4 addressed the problem of Entity Retrieval in the Wikipedia setting. We first presented our efforts in creating standard and reusable test collections for evaluating effectiveness of Entity Retrieval systems working on top of Wikipedia. We then presented original methods based on Natural Language Processing, Information Extraction, and Link Analysis using a large ontology to find relevant entities in Wikipedia. We made use of the Wikipedia structure – page links and categories – and employed an accurate ontology to remove possible noise in Wikipedia category assignments. The experimental results shown that category assignments can be both helpful for retrieval as well as misleading depending on the query. Additionally, we employed several NLP techniques to transform the query and to fill the gaps between the query and the Wikipedia language models. We extracted essential information (lexical expressions, key concepts, named entities) from the query, as well as expanded the terms (by means of synonyms or related words) to find entities. The experimental evaluation of the Entity Retrieval algorithms has shown that by combining our approaches we achieve an average improvement of 24% in terms of xInfAP and of 30% in terms of Precision@10 on the XER task of the INEX-XER 2008 test collection. We also studied combinations of approaches depending on the query in order to maximize effectiveness. For example, by using our methods we achieved an xInfAP value of over 0.7 for 20% of the queries of the used test collection and the mean xInfAP can be further boosted by 27% only by selecting the appropriate approach for each given topic.

In Chapter 5 we looked at the Entity Retrieval problem from different perspectives taking into consideration the time dimension. We dealt with the problem of retrieving entities over a time-stamped document collection as well as with the estimation of public opinion about a given entity from the blogosphere.

First, we analysed entity relevance in news. One of the conclusions is that determining the relevance of a sentence is very important to determine the relevance of an entity; more so than determining sentence novelty. Then, we proposed features both from the current news article and from previous ones in the article's history in order to find relevant entities. Experiments shown that past frequency of entities is the most important of the features explored so far, more important than entity frequency in the current document. Position of the entity in the document (e.g., its first occurrence) is a weak indicator of its relevance. We tested several combinations of features obtaining an overall statistically significant improvement of 15% in terms of Mean Average Precision over the baseline that considers the frequency of entities in the document.

In the second part of Chapter 5 we described methods for extracting public opinion about entities from the blogosphere. In detail, we presented an approach for estimating the development of public opinions over time by mining and aggregating sentiments about politicians. We combined techniques for sentiment analysis with different aggregation, parameter estimation, and forecasting techniques. Experimental results obtained in the context of the US 2008 election campaign showed that purely data-driven, unsupervised approaches can already capture trends, peaks, and switches for public opinions in the political domain. We achieved further improvements by utilizing, in addition to blog data, a history of past opinion poll results. In this scenario, learning parameters that capture intrin-

sic differences between blogosphere and the "real world" and combination of this approach with traditional Time Series Analysis techniques lead to more accurate estimates.

6.2 Open Directions and Future Work

In this thesis we presented different models and methods for retrieving entities given a user query. As we have already discussed specific future work in each of the technical chapters, in the following we present some more general directions that could be taken in the future.

Some of the interesting open research questions deal with how to aggregate information about a single entity for providing better profiles to be searched. A structured representation of entities can be created and user keyword queries can be run on top. This can be done exploiting techniques for answering unstructured queries over structured datasets. Another possible application of the proposed techniques can be an ER system on top of desktop collections. For example, an expert finding algorithm may exploit the user desktop content for finding relevant people names. More challenging, but another possible direction, is the extension the proposed methods for the entire Web of Entities where Semantic Web techniques and data can play a critical role.

All techniques proposed in this thesis (i.e., expert finding, ER over static collections, public opinion estimation, TAER) can be seen as vertical and specialized Search Engines that can be used to answer a given user query. A possible application of such techniques would then be the creation of a novel Search Engine result page that, rather than showing a standard list of ten blue links, can present a richer result set aggregating results from several verticals. Then, new interesting research questions can be addressed as, for example, how to evaluate such novel ways of user interaction with search result pages. At this point, more than result relevance, the overall search task success should be taken into consideration analysing the overall interaction between the user and the system.

Moreover, we can see the results shown by the Search Engine as a very limited sample of the entire set of relevant results available on the Web. Therefore, another interesting questions is how to check whether there is a bias in the result set selected by the Search Engine (e.g., for a controversial query, it only contains results with negative opinions) and how to measure the quality of the sample as compared to the entire result set. One possible approach to create such result set is by means of search result diversification techniques where the aim is to cover as many user intents a query may have as possible.

ACKNOWLEDGMENTS

There are many persons I want to thank for supporting me over the last 5 years in the rocky path of getting a Ph.D. First, I would like to thank prof. Wolfgang Nejdl for giving me the great opportunity to work in the splendid environment of the L3S Research Center. This allowed me to learn not only how to do research but also to see how a big research group can be effectively managed and how to excel in the task of fund raising.

I would also like to thank prof. Arjen P. de Vries for his help, advices, and for asking me to be part of INEX. I thank prof. Heribert Vollmer for being part of my Ph.D. committee.

I thank Dr. Hugo Zaragoza for the great opportunity of spending 3 months at Yahoo! Research in Barcelona where I met a fantastic group of hard-working people who really enjoy what they do.

I thank Claudia, Paul, and Stefan for mentoring and teaching me so many different things from purely research-oriented aspects up to how to effectively manage projects and project proposals.

I thank all the colleagues at L3S that worked with me for their invaluable help in the research work, for all the different and important skills I could learn from them, and, most important, for their friendship.

I am grateful to prof. Stefano Mizzaro for being the first showing me the fun of doing research and to Rossana Dell'Andrea, my computer science teacher in 1997-2000, for making me love this topic.

I thank my parents for being proud and happy for me even if they have not completely realized what my job is about. Last, and most important, I want to thank Monica for countless reasons among which her presence, support, and, overall, for her love.

Gianluca Demartini
Hannover, April 2011

A

Curriculum Vitae

Gianluca Demartini, born on December 6^{th} 1981, in Gorizia, Italy.

Mar. 2006 -	Junior researcher and Ph.D student at Forschungszentrum L3S, Universität Hannover
Sep. 2009 - Nov. 2009	Intern at Yahoo! Research Barcelona, Spain
Nov. 2005 - Feb. 2006	Software Engineer at INSIEL S.p.A., http://www.insiel.it
2003-2005	Master Studies in Computer Science, University of Udine, Italy
2000-2003	Bachelor Studies in Computer Science, University of Udine, Italy

Bibliography

[1] Steven Abney, Michael Collins, and Amit Singhal. Answer extraction. In *Proceedings of the sixth conference on Applied natural language processing*, pages 296–301, Morristown, NJ, USA, 2000. Association for Computational Linguistics.

[2] A.I. Abramowitz. Forecasting the 2008 Presidential Election with the Time-for-Change Model. *PS: Political Science and Politics*, 41(04):691–695, 2008.

[3] S.F. Adafre, M. de Rijke, and E.T.K. Sang. Entity retrieval. *Proceedings of RANLP, Bulgaria*, September 2007.

[4] B. Thomas Adler and Luca de Alfaro. A content-driven reputation system for the wikipedia. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 261–270, New York, NY, USA, 2007. ACM.

[5] Alias-i. LingPipe Named Entity Tagger, 2008. Available at: http://www.alias-i.com/lingpipe/.

[6] James Allan and Hema Raghavan. Using part-of-speech patterns to reduce query ambiguity. In *SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 307–314, New York, NY, USA, 2002. ACM.

[7] Omar Alonso, Michael Gertz, and Ricardo Baeza-Yates. Clustering and exploring search results using timeline constructions. In *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management*, pages 97–106, New York, NY, USA, 2009. ACM.

[8] Omar Alonso, Michael Gertz, and Ricardo A. Baeza-Yates. On the value of temporal information in information retrieval. *SIGIR Forum*, 41(2):35–41, 2007.

[9] Peter G. Anick and Suresh Tipirneni. The paraphrase search assistant: terminological feedback for iterative information seeking. In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 153–159, New York, NY, USA, 1999. ACM.

[10] Irem Arikan, Srikanta J. Bedathur, and Klaus Berberich. Time Will Tell: Leveraging Temporal Expressions in IR. In Ricardo A. Baeza-Yates, Paolo Boldi, Berthier A. Ribeiro-Neto, and Berkant Barla Cambazoglu, editors, *WSDM (Late Breaking-Results)*. ACM, 2009.

[11] J. Atserias, H. Zaragoza, M. Ciaramita, and G. Attardi. Semantically annotated snapshot of the English Wikipedia. *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, 2008.

[12] Ricardo A. Baeza-Yates and Prabhakar Raghavan. Next generation web search. In Stefano Ceri and Marco Brambilla, editors, *SeCO Workshop*, volume 5950 of *Lecture Notes in Computer Science*, pages 11–23. Springer, 2009.

[13] Peter Bailey, Nick Craswell, Ian Soboroff, and Arjen P. de Vries. The CSIRO enterprise search test collection. *SIGIR Forum*, 41(2):42–45, 2007.

[14] Peter Bailey, Arjen P. de Vries, Nick Craswell, and Ian Soboroff. Overview of the TREC 2007 Enterprise Track. In Ellen M. Voorhees and Lori P. Buckland, editors, *Proceedings of The Sixteenth Text REtrieval Conference, TREC 2007, Gaithersburg, Maryland, USA, November 5-9, 2007*, volume Special Publication 500-274. National Institute of Standards and Technology (NIST), 2007.

[15] K. Balog, A. de Vries, P. Serdyukov, P. Thomas, and T. Westerveld. Overview of the TREC 2009 Entity Track. In *The Eighteen Text Retrieval Conference Proceedings (TREC 2009)*. NIST, 2009. Special Publication.

[16] K. Balog, I. Soboroff, P. Thomas, N. Craswell, A. P. de Vries, and P. Bailey. Overview of the TREC 2008 Enterprise Track. In *The Seventeenth Text Retrieval Conference Proceedings (TREC 2008)*. NIST, 2009. Special Publication.

[17] Krisztian Balog, Leif Azzopardi, and Maarten de Rijke. 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, New York, NY, USA, 2006. ACM.

[18] Krisztian Balog, Leif Azzopardi, and Maarten de Rijke. A language modeling framework for expert finding. *Inf. Process. Manage.*, 45(1):1–19, 2009.

[19] Krisztian Balog, Marc Bron, and Maarten de Rijke. Category-based query modeling for entity search. In Cathal Gurrin, Yulan He, Gabriella Kazai, Udo Kruschwitz, Suzanne Little, Thomas Roelleke, Stefan Rger, and Keith van Rijsbergen, editors,

Advances in Information Retrieval, volume 5993 of *Lecture Notes in Computer Science*, pages 319–331. Springer Berlin / Heidelberg, 2010.

[20] Krisztian Balog and Maarten de Rijke. Determining expert profiles (with an application to expert finding). In Manuela M. Veloso, editor, *IJCAI*, pages 2657–2662, 2007.

[21] Krisztian Balog and Maarten de Rijke. Non-local evidence for expert finding. In *Proceeding of the 17th ACM conference on Information and knowledge management*, CIKM '08, pages 489–498, New York, NY, USA, 2008. ACM.

[22] Krisztian Balog, Maarten de Rijke, and Wouter Weerkamp. Bloggers as experts: feed distillation using expert retrieval models. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 753–754, New York, NY, USA, 2008. ACM.

[23] Krisztian Balog, Gilad Mishne, and Maarten de Rijke. Why are they excited?: identifying and explaining spikes in blog mood levels. In *EACL '06: Proceedings of the Eleventh Conference of the European Chapter of the Association for Computational Linguistics: Posters & Demonstrations*, pages 207–210, Morristown, NJ, USA, 2006. Association for Computational Linguistics.

[24] J.R. Baron, D.D. Lewis, and D.W. Oard. TREC-2006 legal track overview. In *The Fifteenth Text REtrieval Conference (TREC 2006) Proceedings*, 2006.

[25] Holger Bast, Alexandru Chitea, Fabian Suchanek, and Ingmar Weber. ESTER: efficient search on text, entities, and relations. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 671–678, New York, NY, USA, 2007. ACM.

[26] T. Beckers, P. Bellot, G. Demartini, L. Denoyer, C.M. De Vries, A. Doucet, K.N. Fachry, N. Fuhr, P. Gallinari, S. Geva, W.-C. Huang, T. Iofciu, J. Kamps, G. Kazai, M. Koolen, S. Kutty, M. Landoni, M. Lehtonen, V. Moriceau, R. Nayak, R. Nordlie, N. Pharo, E. San Juan, R. Schenkel, X. Tannier, M. Theobald, J.A. Thom, A. Trotman, and A.P. de Vries. Report on INEX 2009. *SIGIR Forum*, 44(1):38–56, 2010.

[27] Klaus Berberich, Srikanta Bedathur, Thomas Neumann, and Gerhard Weikum. A time machine for text search. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 519–526, New York, NY, USA, 2007. ACM.

[28] J. Bhogal, A. Macfarlane, and P. Smith. A review of ontology based query expansion. *Inf. Process. Manage.*, 43(4):866–886, 2007.

[29] Bodo Billerbeck, Gianluca Demartini, Claudiu S. Firan, Tereza Iofciu, and Ralf Krestel. Exploiting click-through data for entity retrieval. In Crestani et al. [52], pages 803–804.

[30] Bodo Billerbeck, Gianluca Demartini, Claudiu S. Firan, Tereza Iofciu, and Ralf Krestel. Ranking Entities Using Web Search Query Logs. In *ECDL*, Lecture Notes in Computer Science. Springer, 2010.

[31] Paolo Bouquet, Harry Halpin, Heiko Stoermer, and Giovanni Tummarello, editors. *Proceedings of the 1st international workshop on Identity and Reference on the Semantic Web (IRSW2008) at the 5th European Semantic Web Conference (ESWC 2008), Tenerife, Spain, June 2nd, 2008*, CEUR Workshop Proceedings. CEUR-WS.org, 2008.

[32] Paolo Bouquet, Heiko Stoermer, and Barbara Bazzanella. An Entity Name System (ENS) for the Semantic Web. In Sean Bechhofer, Manfred Hauswirth, Jörg Hoffmann, and Manolis Koubarakis, editors, *The Semantic Web: Research and Applications, 5th European Semantic Web Conference, ESWC 2008, Tenerife, Canary Islands, Spain, June 1-5, 2008, Proceedings*, volume 5021 of *Lecture Notes in Computer Science*, pages 258–272. Springer, 2008.

[33] Paolo Bouquet, Heiko Stoermer, Claudia Niederee, and Antonio Mana. Entity Name System: The Backbone of an Open and Scalable Web of Data. In *Proceedings of the IEEE International Conference on Semantic Computing, ICSC 2008*, CSS-ICSC, pages 554–561. IEEE Computer Society, 2008.

[34] Paolo Bouquet, Heiko Stoermer, Giovanni Tummarello, and Harry Halpin, editors. *Proceedings of the WWW2007 Workshop I^3: Identity, Identifiers, Identification, Entity-Centric Approaches to Information and Knowledge Management on the Web, Banff, Canada, May 8, 2007*, volume 249 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2007.

[35] Andrei Z. Broder. A Taxonomy of Web Search. *SIGIR Forum*, 36(2):3–10, 2002.

[36] David Carmel, Elad Yom-Tov, and Ian Soboroff. SIGIR workshop report: predicting query difficulty - methods and applications. *SIGIR Forum*, 39(2):25–28, 2005.

[37] William B. Cavnar and John M. Trenkle. N-gram-based text categorization. In *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175, 1994.

[38] Soumen Chakrabarti. Dynamic personalized pagerank in entity-relation graphs. In *WWW*, pages 571–580, 2007.

[39] Tao Cheng and Kevin Chen-Chuan Chang. Entity Search Engine: Towards Agile Best-Effort Information Integration over the Web. In *CIDR 2007, Third Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 7-10, 2007, Online Proceedings*, pages 108–113. www.crdrdb.org, 2007.

[40] Tao Cheng, Xifeng Yan, and Kevin Chen-Chuan Chang. EntityRank: Searching Entities Directly and Holistically. In *VLDB '07: Proceedings of the 33rd international conference on Very large data bases*, pages 387–398. VLDB Endowment, 2007.

[41] Tao Cheng, Xifeng Yan, and Kevin Chen-Chuan Chang. Supporting entity search: a large-scale prototype search engine. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, SIGMOD '07, pages 1144–1146, New York, NY, USA, 2007. ACM.

[42] Sergey Chernov, Gianluca Demartini, and Julien Gaugaz. L3s research center at trec 2006 enterprise track. In Ellen M. Voorhees and Lori P. Buckland, editors, *TREC*, volume Special Publication 500-272. National Institute of Standards and Technology (NIST), 2006.

[43] Sergey Chernov, Gianluca Demartini, Eelco Herder, Michal Kopycki, and Wolfgang Nejdl. Evaluating Personal Information Management Using an Activity Logs Enriched Desktop Dataset. In *Proceedings of 3rd Personal Information Management Workshop (PIM 2008)*, 2008.

[44] Sergey Chernov, Pavel Serdyukov, Paul-Alexandru Chirita, Gianluca Demartini, and Wolfgang Nejdl. Building a desktop search test-bed. In Giambattista Amati, Claudio Carpineto, and Giovanni Romano, editors, *ECIR*, volume 4425 of *Lecture Notes in Computer Science*, pages 686–690. Springer, 2007.

[45] P.A. Chirita, C.S. Firan, and W. Nejdl. Summarizing Local Context to Personalize Global Web Search. In *CIKM*, pages 287–296, 2006.

[46] Paul Alexandru Chirita, Claudiu S. Firan, and Wolfgang Nejdl. Personalized Query Expansion for the Web. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 7–14, New York, NY, USA, 2007. ACM.

[47] P.S.P. Cowpertwait and A.V. Metcalfe. *Introductory Time Series with R*. Springer London, Limited, 2009.

[48] N. Craswell and D. Hawking. Overview of the TREC-2004 Web Track. *The Thirteenth Text REtrieval Conference (TREC 2004)*, 2004.

[49] N. Craswell, D. Hawking, A. Vercoustre, and P. Wilkins. P@noptic Expert: Searching for Experts not just for Documents. *Ausweb, 2001*, 2001.

[50] Nick Craswell, Gianluca Demartini, Julien Gaugaz, and Tereza Iofciu. L3S at INEX 2008: Retrieving Entities Using Structured Information. In Geva et al. [89], pages 253–263.

[51] Nick Craswell, Stephen Robertson, Hugo Zaragoza, and Michael Taylor. Relevance weighting for query independent evidence. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 416–423, New York, NY, USA, 2005. ACM.

[52] Fabio Crestani, Stéphane Marchand-Maillet, Hsin-Hsi Chen, Efthimis N. Efthimiadis, and Jacques Savoy, editors. *Proceeding of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2010, Geneva, Switzerland, July 19-23, 2010*. ACM, 2010.

[53] S. Deerwester, S.T. Dumais, G.W. Furnas, T.K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407, 1990.

[54] Gianluca Demartini. Finding experts using wikipedia. In Anna V. Zhdanova, Lyndon J. B. Nixon, Malgorzata Mochol, and John G. Breslin, editors, *FEWS*, volume 290 of *CEUR Workshop Proceedings*, pages 33–41. CEUR-WS.org, 2007.

[55] Gianluca Demartini. Leveraging semantic technologies for enterprise search. In Aparna S. Varde and Jian Pei, editors, *PIKM*, pages 25–32. ACM, 2007.

[56] Gianluca Demartini. Comparing people in the enterprise. In José Cordeiro and Joaquim Filipe, editors, *ICEIS (2)*, pages 455–458, 2008.

[57] Gianluca Demartini. How Many Experts? - A New Task for Enterprise Search Evaluation. In *Workshop on Novel Methodologies for Evaluation in Information Retrieval at the 30th European Conference on IR Research, ECIR 2008*, pages 39–43, 2008.

[58] Gianluca Demartini. ARES: A Retrieval Engine based on Sentiments - Sentiment-based Search Result Annotation and Diversification. In *33rd European Conference on Information Retrieval (ECIR 2011)*, 2011.

[59] Gianluca Demartini, Paul-Alexandru Chirita, Ingo Brunkhorst, and Wolfgang Nejdl. Ranking categories for web search. In Macdonald et al. [120], pages 564–569.

[60] Gianluca Demartini, Arjen P. de Vries, Tereza Iofciu, and Jianhan Zhu. Overview of the inex 2008 entity ranking track. In Geva et al. [89], pages 243–252.

[61] Gianluca Demartini, Ludovic Denoye, Antoine Doucet, Khairun Nisa Fachry, Patrick Gallinar, Shlomo Geva, Darren Wei Che Huang, Tereza Iofciu, Jaap Kamps, Gabriella Kazai, Marijn Koolen, Monica Landoni, Ragnar Nordlie, Nils Pharo, Ralf Schenkel, Martin Theobald, Andrew Trotman, Arjen P. de Vries, Alan Woodley, and Jianhan Zhu. Report on inex 2008. *SIGIR Forum*, 43(1):17–36, 2009.

[62] Gianluca Demartini, Claudiu S. Firan, Mihai Georgescu, Tereza Iofciu, Ralf Krestel, and Wolfgang Nejdl. An architecture for finding entities on the web. In Edgar

Chávez, Elizabeth Furtado, and Alberto L. Morán, editors, *LA-WEB/CLIHC*, pages 230–237. IEEE Computer Society, 2009.

[63] Gianluca Demartini, Claudiu S. Firan, and Tereza Iofciu. L3s at inex 2007: Query expansion for entity ranking using a highly accurate ontology. In Norbert Fuhr, Jaap Kamps, Mounia Lalmas, and Andrew Trotman, editors, *INEX*, volume 4862 of *Lecture Notes in Computer Science*, pages 252–263. Springer, 2007.

[64] Gianluca Demartini, Claudiu S. Firan, Tereza Iofciu, Ralf Krestel, and Wolfgang Nejdl. A Model for Ranking Entities and Its Application to Wikipedia. In *LA-WEB '08: Proceedings of the 2008 Latin American Web Conference*, pages 29–38, Washington, DC, USA, 2008. IEEE Computer Society.

[65] Gianluca Demartini, Claudiu S. Firan, Tereza Iofciu, Ralf Krestel, and Wolfgang Nejdl. Why Finding Entities in Wikipedia is Difficult, Sometimes. *Inf. Retr.*, 13(4), 2010.

[66] Gianluca Demartini, Claudiu S. Firan, Tereza Iofciu, and Wolfgang Nejdl. Semantically enhanced entity ranking. In James Bailey, David Maier, Klaus-Dieter Schewe, Bernhard Thalheim, and Xiaoyang Sean Wang, editors, *WISE*, volume 5175 of *Lecture Notes in Computer Science*, pages 176–188. Springer, 2008.

[67] Gianluca Demartini, Julien Gaugaz, and Wolfgang Nejdl. A vector space model for ranking entities and its application to expert search. In Mohand Boughanem, Catherine Berrut, Josiane Mothe, and Chantal Soulé-Dupuy, editors, *ECIR*, volume 5478 of *Lecture Notes in Computer Science*, pages 189–201. Springer, 2009.

[68] Gianluca Demartini, Tereza Iofciu, and Arjen P. de Vries. Overview of the inex 2009 entity ranking track. In *INEX*, 2009.

[69] Gianluca Demartini, Malik Muhammad Saad Missen, Roi Blanco, and Hugo Zaragoza. Entity summarization of news articles. In Crestani et al. [52], pages 795–796.

[70] Gianluca Demartini, Malik Muhammad Saad Missen, Roi Blanco, and Hugo Zaragoza. TAER: Time-Aware Entity Retrieval - Exploiting the Past to find Relevant Entities in News Articles. In *The 19th ACM International Conference on Information and Knowledge Management (CIKM 2010)*, 2010.

[71] Gianluca Demartini and Claudia Niederée. Finding Experts on the Semantic Desktop. In *Personal Identification and Collaborations: Knowledge Mediation and Extraction (PICKME 2008) Workshop at ISWC 2008*, 2008.

[72] Gianluca Demartini and Stefan Siersdorfer. Dear Search Engine: What's your opinion about...? - Sentiment Analysis for Semantic Enrichment of Web Search Results. In *Semantic Search 2010 Workshop located at the 19th Int. World Wide Web Conference WWW2010*, 2010.

[73] Gianluca Demartini, Stefan Siersdorfer, Sergiu Chelaru, and Wolfgang Nejdl. Analyzing Political Trends in the Blogosphere. In *Fifth International AAAI Conference on Weblogs and Social Media (ICWSM 2011)*, 2011.

[74] Kerstin Denecke. Are sentiwordnet scores suited for multi-domain sentiment classification? In Bill Grosky, Frédéric Andrès, and Pit Pichappan, editors, *ICDIM*, pages 33–38. IEEE, 2009.

[75] Ludovic Denoyer and Patrick Gallinari. The Wikipedia XML corpus. *SIGIR Forum*, 40(1):64–69, 2006.

[76] Fernando Diaz. Integration of news content into web results. In *WSDM '09: Proceedings of the Second ACM International Conference on Web Search and Data Mining*, pages 182–191, New York, NY, USA, 2009. ACM.

[77] S.T. Dumais. Improving the retrieval of information from external sources. *Behavior Research Methods, Instruments and Computers*, 23(2):229–236, 1991.

[78] Erik Elgersma and Maarten de Rijke. Personal vs non-personal blogs: initial classification experiments. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 723–724, New York, NY, USA, 2008. ACM.

[79] A. Esuli and F. Sebastiani. SentiWordNet: A publicly available lexical resource for opinion mining. In *Proceedings of LREC*, volume 6. Citeseer, 2006.

[80] Oren Etzioni, Michele Banko, Stephen Soderland, and Daniel S. Weld. Open information extraction from the web. *Commun. ACM*, 51(12):68–74, 2008.

[81] Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. Unsupervised named-entity extraction from the web: an experimental study. *Artif. Intell.*, 165(1):91–134, 2005.

[82] R. Fagin, R. Kumar, K.S. McCurley, J. Novak, D. Sivakumar, J.A. Tomlin, and D.P. Williamson. Searching the workplace web. *Proceedings of the 12th international conference on World Wide Web*, pages 366–375, 2003.

[83] Hui Fang and ChengXiang Zhai. Probabilistic models for expert finding. In Giambattista Amati, Claudio Carpineto, and Giovanni Romano, editors, *Advances in Information Retrieval*, volume 4425 of *Lecture Notes in Computer Science*, pages 418–430. Springer Berlin / Heidelberg, 2007.

[84] Christiane Fellbaum. *WordNet: An Electronic Lexical Database (Language, Speech, and Communication)*. The MIT Press, May 1998.

[85] R. Forsythe, F. Nelson, G.R. Neumann, and J. Wright. Anatomy of an experimental political stock market. *The American Economic Review*, pages 1142–1161, 1992.

[86] Julien Gaugaz and Gianluca Demartini. Entity identifiers for lineage preservation. In Bouquet et al. [31].

[87] Julien Gaugaz, Jakub Zakrzewski, Gianluca Demartini, and Wolfgang Nejdl. How to trace and revise identities. In Lora Aroyo, Paolo Traverso, Fabio Ciravegna, Philipp Cimiano, Tom Heath, Eero Hyvönen, Riichiro Mizoguchi, Eyal Oren, Marta Sabou, and Elena Paslaru Bontas Simperl, editors, *ESWC*, volume 5554 of *Lecture Notes in Computer Science*, pages 414–428. Springer, 2009.

[88] Georgina Gaughan and Alan F. Smeaton. Finding new news: Novelty detection in broadcast news. In Gary Geunbae Lee, Akio Yamada, Helen Meng, and Sung-Hyon Myaeng, editors, *AIRS*, volume 3689 of *Lecture Notes in Computer Science*, pages 583–588. Springer, 2005.

[89] Shlomo Geva, Jaap Kamps, and Andrew Trotman, editors. *Advances in Focused Retrieval, 7th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2008, Dagstuhl Castle, Germany, December 15-18, 2008. Revised and Selected Papers*, volume 5631 of *Lecture Notes in Computer Science*. Springer, 2009.

[90] Tom Gruber. Ontology. In Ling Liu and M. Tamer Özsu, editors, *Encyclopedia of Database Systems*, pages 1963–1965. Springer US, 2009.

[91] Donna Harman. Overview of the TREC 2002 Novelty Track. In *The Eleventh Text Retrieval Conference Proceedings (TREC 2002)*. NIST, 2002. Special Publication.

[92] D. Hawking. Challenges in enterprise search. *Proceedings of the Australasian Database Conference ADC2004, pages*, pages 15–26, 2004.

[93] Tom Heath and Enrico Motta. Revyu: Linking reviews and ratings into the Web of Data. *J. Web Sem.*, 6(4):266–273, 2008.

[94] Ming-Hung Hsu, Ming-Feng Tsai, and Hsin-Hsi Chen. Query Expansion with ConceptNet and WordNet: An Intrinsic Comparison. In Hwee Tou Ng, Mun-Kew Leong, Min-Yen Kan, and Donghong Ji, editors, *Information Retrieval Technology, Third Asia Information Retrieval Symposium, AIRS 2006, Singapore, October 16-18, 2006, Proceedings*, volume 4182 of *Lecture Notes in Computer Science*, pages 1–13. Springer, 2006.

[95] Guoping Hu, Jingjing Liu, Hang Li, Yunbo Cao, Jian-Yun Nie, and Jianfeng Gao. A supervised learning approach to entity search. In Hwee Ng, Mun-Kew Leong, Min-Yen Kan, and Donghong Ji, editors, *Information Retrieval Technology*, volume 4182 of *Lecture Notes in Computer Science*, pages 54–66. Springer Berlin / Heidelberg, 2006.

[96] Tereza Iofciu and Gianluca Demartini. Time based Tag Recommendation using Direct and Extended Users Sets. In Folke Eisterlehner, Andreas Hotho, and Robert Jschke, editors, *ECML PKDD Discovery Challenge 2009 (DC09)*, volume 497 of *CEUR-WS.org*, September 2009.

[97] Tereza Iofciu, Gianluca Demartini, Nick Craswell, and Arjen P. de Vries. ReFER: effective Relevance Feedback for Entity Ranking. In *33rd European Conference on Information Retrieval (ECIR 2011)*, 2011.

[98] Lifeng Jia, Clement T. Yu, and Wei Zhang. UIC at TREC 208 Blog Track. In *TREC*, 2008.

[99] Jiepu Jiang, Wei Lu, Xianqian Rong, and Yangyan Gao. Adapting Expert Search Models to Rank Entities. In Geva et al. [89].

[100] Thorsten Joachims. Making large-scale support vector machine learning practical. In *Advances in kernel methods: support vector learning*, pages 169–184, Cambridge, MA, USA, 1999. MIT Press.

[101] Massimiliano Ciaramita Jordi Atserias, Hugo Zaragoza and Giuseppe Attardi. Semantically Annotated Snapshot of the English Wikipedia. In Nicoletta Calzolari, Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odjik, Stelios Piperidis, and Daniel Tapias, editors, *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, may 2008. European Language Resources Association (ELRA). http://www.lrec-conf.org/proceedings/lrec2008/.

[102] Rianne Kaptein and Jaap Kamps. Finding Entities in Wikipedia using Links and Categories. In Geva et al. [89].

[103] Rianne Kaptein, Pavel Serdyukov, Arjen P. de Vries, and Jaap Kamps. Entity ranking using wikipedia as a pivot. In Jimmy Huang, Nick Koudas, Gareth J. F. Jones, Xindong Wu, Kevyn Collins-Thompson, and Aijun An, editors, *CIKM*, pages 69–78. ACM, 2010.

[104] Christian Kohlschütter, Peter Fankhauser, and Wolfgang Nejdl. Boilerplate detection using shallow text features. In *WSDM '10: Proceedings of the third ACM international conference on Web search and data mining*, pages 441–450, New York, NY, USA, 2010.

[105] Ralf Krestel, Gianluca Demartini, and Eelco Herder. Visual Interfaces for Stimulating Exploratory Search. In *ACM/IEEE Joint Conference on Digital Libraries (JCDL 2011)*, 2011.

[106] Ravi Kumar and Andrew Tomkins. A characterization of online search behavior. *IEEE Data Eng. Bull.*, 32(2):3–11, 2009.

[107] S. Le Cessie and JC Van Houwelingen. Ridge estimators in logistic regression. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 41(1):191–201, 1992.

[108] Yeha Lee, Seung-Hoon Na, Jungi Kim, Sang-Hyob Nam, Hun-Young Jung, and Jong-Hyeok Lee. KLE at TREC 2008 Blog Track: Blog Post and Feed Retrieval. In *TREC*, 2008.

[109] Jure Leskovec, Lars Backstrom, and Jon Kleinberg. Meme-tracking and the dynamics of the news cycle. In *KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 497–506, New York, NY, USA, 2009. ACM.

[110] H. Li, Y. Cao, J. Xu, Y. Hu, S. Li, and D. Meyerzon. A new approach to intranet search based on information extraction. *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 460–468, 2005.

[111] Xiaoyan Li and W. Bruce Croft. Time-based language models. In *CIKM '03: Proceedings of the twelfth international conference on Information and knowledge management*, pages 469–475, New York, NY, USA, 2003. ACM.

[112] Xiaoyan Li and W. Bruce Croft. Novelty detection based on sentence level patterns. In *CIKM '05: Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 744–751, New York, NY, USA, 2005. ACM.

[113] Xiaoyan Li and W. Bruce Croft. Improving novelty detection for general topics using sentence level information patterns. In *CIKM '06: Proceedings of the 15th ACM international conference on Information and knowledge management*, pages 238–247, New York, NY, USA, 2006. ACM.

[114] Xiaoyan Li and W. Bruce Croft. An information-pattern-based approach to novelty detection. *Inf. Process. Manage.*, 44(3):1159–1188, 2008.

[115] Yang Liu, Xiangji Huang, Aijun An, and Xiaohui Yu. Arsa: a sentiment-aware model for predicting sales performance using blogs. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 607–614, New York, NY, USA, 2007. ACM.

[116] C. Macdonald and I. Ounis. Voting for Candidates: Adapting Data Fusion Techniques for an Expert Search Task. In *CIKM*, pages 387–396, 2006.

[117] C. Macdonald and I. Ounis. Using Relevance Feedback in Expert Search. In *ECIR*, pages 431–443, 2007.

[118] Craig Macdonald, David Hannah, and Iadh Ounis. High quality expertise evidence for expert search. In *ECIR*, pages 283–295, 2008.

[119] Craig Macdonald and Iadh Ounis. Expertise drift and query expansion in expert search. In *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 341–350, New York, NY, USA, 2007. ACM.

[120] Craig Macdonald, Iadh Ounis, Vassilis Plachouras, Ian Ruthven, and Ryen W. White, editors. *Advances in Information Retrieval , 30th European Conference on IR Research, ECIR 2008, Glasgow, UK, March 30-April 3, 2008. Proceedings*, volume 4956 of *Lecture Notes in Computer Science*. Springer, 2008.

[121] C.D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[122] A. McLean, A.M. Vercoustre, and M. Wu. Enterprise PeopleFinder: Combining Evidence from Web Pages and Corporate Data. *Proceedings of Australian Document Computing Symposium*, 2003.

[123] Frank McSherry and Marc Najork. Computing information retrieval performance measures efficiently in the presence of tied scores. In *ECIR'08: Proceedings of the IR research, 30th European conference on Advances in information retrieval*, pages 414–421, Berlin, Heidelberg, 2008. Springer-Verlag.

[124] Enrico Minack, Gianluca Demartini, and Wolfgang Nejdl. Current Approaches to Search Result Diversification. In *Proceedings of First International Workshop on Living Web, Collocated with the 8th International Semantic Web Conference (ISWC-2009), Washington D.C., USA*. CEUR-WS, October 2009.

[125] Enrico Minack, Raluca Paiu, Stefania Costache, Gianluca Demartini, Julien Gaugaz, Ekaterini Ioannou, Paul-Alexandru Chirita, and Wolfgang Nejdl. Leveraging personal metadata for desktop search: The beagle^{++} system. *J. Web Sem.*, 8(1):37–54, 2010.

[126] Gilad Mishne and Maarten de Rijke. Deriving wishlists from blogs show us your blog, and we'll tell you what books to buy. In *WWW*, pages 925–926, 2006.

[127] JA Nelder and R. Mead. A simplex method for function minimization. *The computer journal*, 7(4):308, 1965.

[128] Brendan O'Connor, Ramnath Balasubramanyan, Bryan R. Routledge, and Noah A. Smith. From tweets to polls: Linking text sentiment to public opinion time series. In *Proceedings of the International AAAI Conference on Weblogs and Social Media*, 2010.

[129] Deanna Osman, John Yearwood, and Peter Vamplew. Automated opinion detection: Implications of the level of agreement between human raters. *Inf. Process. Manage.*, 46(3):331–342, 2010.

[130] I. Ounis, C. Macdonald, and I. Soboroff. Overview of the TREC 2008 Blog Track. In *TREC*, 2008.

[131] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999.

[132] Themis Palpanas, Junaid Chaudhry, Periklis Andritsos, and Yannis Velegrakis. Entity Data Management in OKKAM. In *DEXA '08: Proceedings of the 2008 19th International Conference on Database and Expert Systems Application*, pages 729–733, Washington, DC, USA, 2008. IEEE Computer Society.

[133] Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the ACL*, 2005.

[134] George Papadakis, Gianluca Demartini, Philipp Kärger, and Peter Fankhauser. The Missing Links: Discovering Hidden Same-as Links among a Billion of Triples. In *iiWAS*, 2010.

[135] Jovan Pehcevski, Anne-Marie Vercoustre, and James A. Thom. Exploiting Locality of Wikipedia Links in Entity Ranking. In Craig Macdonald, Iadh Ounis, Vassilis Plachouras, Ian Ruthven, and Ryen W. White, editors, *Advances in Information Retrieval , 30th European Conference on IR Research, ECIR 2008, Glasgow, UK, March 30-April 3, 2008. Proceedings*, volume 4956 of *Lecture Notes in Computer Science*, pages 258–269. Springer, 2008.

[136] Desislava Petkova and W. Bruce Croft. Hierarchical language models for expert finding in enterprise corpora. *Tools with Artificial Intelligence, IEEE International Conference on*, 0:599–608, 2006.

[137] Desislava Petkova and W. Bruce Croft. Proximity-based document representation for named entity retrieval. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, CIKM '07, pages 731–740, New York, NY, USA, 2007. ACM.

[138] Yves Raimond, Christopher Sutton, and Mark Sandler. Automatic Interlinking of Music Datasets on the Semantic Web. In *Linked Data on the Web (LDOW2008)*, 2008.

[139] S. E. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 232–241, New York, NY, USA, 1994. Springer-Verlag New York, Inc.

[140] Stephen Robertson and Hugo Zaragoza. The Probabilistic Relevance Framework: BM25 and Beyond. *Found. Trends Inf. Retr.*, 3(4):333–389, 2009.

[141] Henning Rode, Djoerd Hiemstra, Arjen P. de Vries, and Pavel Serdyukov. Efficient XML and Entity Retrieval with PF/Tijah: CWI and University of Twente at INEX'08. In Geva et al. [89].

[142] Henning Rode, Pavel Serdyukov, and Djoerd Hiemstra. 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, New York, NY, USA, 2008. ACM.

[143] T. Rölleke, T. Tsikrika, and G. Kazai. A general matrix framework for modelling Information Retrieval. *Information Processing and Management*, 42(1):4–30, 2006.

[144] G. Salton, A. Wong, and CS Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.

[145] Ralf Schenkel, Fabian M. Suchanek, and Gjergji Kasneci. YAWN: A Semantically Annotated Wikipedia XML Corpus. In Alfons Kemper, Harald Schöning, Thomas Rose, Matthias Jarke, Thomas Seidl, Christoph Quix, and Christoph Brochhaus, editors, *Datenbanksysteme in Business, Technologie und Web (BTW 2007), 12. Fachtagung des GI-Fachbereichs "Datenbanken und Informationssysteme" (DBIS), Proceedings, 7.-9. März 2007, Aachen, Germany*, volume 103 of *LNI*, pages 277–291. GI, 2007.

[146] Giovanni Semeraro, Marco Degemmis, Pasquale Lops, and Pierpaolo Basile. Combining learning and word sense disambiguation for intelligent user profiling. In *IJCAI'07: Proceedings of the 20th international joint conference on Artifical intelligence*, pages 2856–2861, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

[147] P. Serdyukov and A. de Vries. Delft University at the TREC 2009 Entity track: ranking Wikipedia entities. In *TREC*, 2009.

[148] P. Serdyukov and D. Hiemstra. Being omnipresent to be almighty: The importance of the global web evidence for organizational expert finding. In *Proceedings of the SIGIR 2008 Workshop on Future Challenges in Expertise Retrieval (fCHER)*, pages 17–24, 2008.

[149] Pavel Serdyukov and Djoerd Hiemstra. Modeling documents as mixtures of persons for expert finding. In Macdonald et al. [120], pages 309–320.

[150] Pavel Serdyukov, Henning Rode, and Djoerd Hiemstra. Modeling multi-step relevance propagation for expert finding. In *Proceeding of the 17th ACM conference on Information and knowledge management*, CIKM '08, pages 1133–1142, New York, NY, USA, 2008. ACM.

[151] P. Sinha and A.K. Bansal. Hierarchical Bayes Prediction for the 2008 US Presidential Election. *The Journal of Prediction Markets*, 2(3):47–59, 2008.

[152] Ian Soboroff. Overview of the TREC 2004 Novelty Track. In *The Thirteenth Text Retrieval Conference Proceedings (TREC 2004)*. NIST, 2004. Special Publication.

[153] Ian Soboroff and Donna Harman. Overview of the TREC 2003 Novelty Track. In *The Twelfth Text Retrieval Conference Proceedings (TREC 2003)*, pages 38–53. NIST, 2003. Special Publication.

[154] Ian Soboroff and Donna Harman. Novelty detection: the trec experience. In *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 105–112, Morristown, NJ, USA, 2005. Association for Computational Linguistics.

[155] Rodolfo Stecher, Gianluca Demartini, and Claudia Niederée. Social recommendations of content and metadata. In Gabriele Kotsis, David Taniar, Eric Pardede, and Ismail Khalil Ibrahim, editors, *iiWAS*, pages 92–97. ACM, 2008.

[156] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In Carey L. Williamson, Mary Ellen Zurko, Peter F. Patel-Schneider, and Prashant J. Shenoy, editors, *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007*, pages 697–706. ACM, 2007.

[157] J.A. Thom, J. Pehcevski, and A.M. Vercoustre. Use of Wikipedia categories in entity ranking. In *Proceedings of 12th Australasian Document Computing Symposium (ADCS07)*, pages 56–63, 2007.

[158] Theodora Tsikrika, Pavel Serdyukov, Henning Rode, Thijs Westerveld, Robin Aly, Djoerd Hiemstra, and Arjen P. Vries. Structured Document Retrieval, Multimedia Retrieval, and Entity Ranking Using PF/Tijah. In *Focused Access to XML Documents: 6th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2007 Dagstuhl Castle, Germany, December 17-19, 2007. Selected Papers*, pages 306–320, Berlin, Heidelberg, 2008. Springer-Verlag.

[159] Anne-Marie Vercoustre, Jovan Pehcevski, and Vladimir Naumovski. Topic Difficulty Prediction in Entity Ranking. In *Advances in Focused Retrieval: 7th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2008, Dagstuhl Castle, Germany, December 15-18, 2008. Revised and Selected Papers*, pages 280–291, Berlin, Heidelberg, 2009. Springer-Verlag.

[160] Anne-Marie Vercoustre, Jovan Pehcevski, and James Thom. Using wikipedia categories and links in entity ranking. In Norbert Fuhr, Jaap Kamps, Mounia Lalmas, and Andrew Trotman, editors, *Focused Access to XML Documents*, volume 4862 of *Lecture Notes in Computer Science*, pages 321–335. Springer Berlin / Heidelberg, 2008.

[161] Anne-Marie Vercoustre, James A. Thom, and Jovan Pehcevski. Entity ranking in wikipedia. In *SAC '08: Proceedings of the 2008 ACM symposium on Applied computing*, pages 1101–1106, New York, NY, USA, 2008. ACM.

[162] Ellen M. Voorhees. On Expanding Query Vectors with Lexically Related Words. In *TREC*, pages 223–232, 1993.

[163] Ellen M. Voorhees. Query Expansion using lexical-semantic Relations. In *SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 61–69, New York, NY, USA, 1994. Springer-Verlag New York, Inc.

[164] Arjen P. Vries, Anne-Marie Vercoustre, James A. Thom, Nick Craswell, and Mounia Lalmas. Overview of the INEX 2007 Entity Ranking Track. In *Focused Access to XML Documents: 6th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2007 Dagstuhl Castle, Germany, December 17-19, 2007. Selected Papers*, pages 245–251, Berlin, Heidelberg, 2007. Springer-Verlag.

[165] William Webber, Alistair Moffat, Justin Zobel, and Tetsuya Sakai. Precision-at-ten Considered Redundant. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 695–696, New York, NY, USA, 2008. ACM.

[166] W.W.S. Wei. *Time series analysis: univariate and multivariate methods*. Addison-Wesley, 2006.

[167] J. Wolfers and E. Zitzewitz. Prediction markets. *Journal of Economic Perspectives*, pages 107–126, 2004.

[168] Emine Yilmaz and Javed A. Aslam. 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, NY, USA, 2006. ACM.

[169] Emine Yilmaz, Evangelos Kanoulas, and Javed A. Aslam. A simple and efficient sampling method for estimating AP and NDCG. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 603–610, New York, NY, USA, 2008. ACM.

[170] Hugo Zaragoza, Henning Rode, Peter Mika, Jordi Atserias, Massimiliano Ciaramita, and Giuseppe Attardi. Ranking very many typed entities on wikipedia. In *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 1015–1018, New York, NY, USA, 2007. ACM.

[171] Yi Zhang and Flora S. Tsai. Combining named entities and tags for novel sentence detection. In *ESAIR '09: Proceedings of the WSDM '09 Workshop on Exploiting*

Semantic Annotations in Information Retrieval, pages 30–34, New York, NY, USA, 2009. ACM.

[172] H. Zhu, S. Raghavan, S. Vaithyanathan, and A. Löser. Navigating the intranet with high precision. *Proceedings of the 16th international conference on World Wide Web*, pages 491–500, 2007.

[173] Jianhan Zhu, Arjen P. de Vries, Gianluca Demartini, and Tereza Iofciu. Evaluating Relation Retrieval for Entities and Experts. In *Future Challenges in Expertise Retrieval (fCHER 2008), SIGIR 2008 Workshop*, 2008.

[174] Cäcilia Zirn, Vivi Nastase, and Michael Strube. Distinguishing between Instances and Classes in the Wikipedia Taxonomy. In Sean Bechhofer, Manfred Hauswirth, Jörg Hoffmann, and Manolis Koubarakis, editors, *The Semantic Web: Research and Applications, 5th European Semantic Web Conference, ESWC 2008, Tenerife, Canary Islands, Spain, June 1-5, 2008, Proceedings*, volume 5021 of *Lecture Notes in Computer Science*, pages 376–387. Springer, 2008.