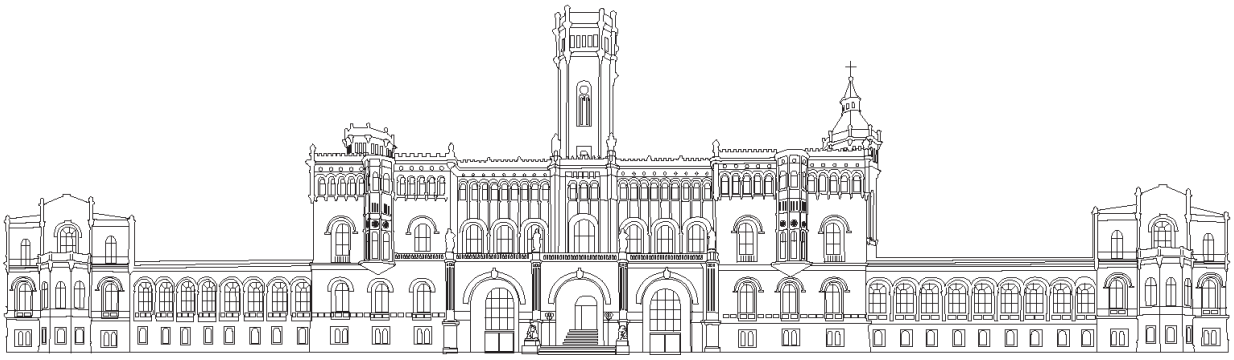

LEIBNIZ UNIVERSITÄT HANNOVER

FAKULTÄT FÜR ELEKTROTECHNIK UND INFORMATIK
FORSCHUNGSZENTRUM L3S

Ranking Archived Documents for Structured Queries on Semantic Layers

Master Thesis

submitted by

Vaibhav Kasturia

on 23.01.2018

First Examiner : Prof. Dr. Eirini Ntoutsi
Second Examiner : Prof. Dr. Wolfgang Nejdl
Supervisor : Dr. Pavlos Fafalios

This thesis is dedicated to my mother Sangeeta Kasturia

Declaration

I hereby confirm that I have prepared the presented Master Thesis without the help of third parties and only with the specified sources and tools. All passages that have been taken from the sources, either literally or in terms of content, have been identified as such. This thesis has not been submitted in the same or similar form to any other examining authority.

Hannover, the 23rd January 2018

Vaibhav Kasturia

Contents

List of Figures

List of Tables

Abstract

Archived collections of documents (like newspaper and web archives) serve as important information sources in a variety of disciplines, including Digital Humanities, Historical Science, and Journalism. However, the absence of efficient and meaningful exploration methods still remains a major hurdle in the way of turning them into usable sources of information. A semantic layer is an RDF graph that describes metadata and semantic information about a collection of archived documents, which in turn can be queried through a semantic query language (SPARQL). This allows running advanced queries by combining metadata of the documents (like publication date) and content-based semantic information (like entities mentioned in the documents). However, the results returned by such structured queries can be numerous and moreover they all equally match the query. In this thesis, we deal with this problem and formalize the task of *"ranking archived documents for structured queries on semantic graphs"*. We define several approaches to describe the aspects of relativeness of documents to entities and the temporal relations among the entities. Based on performance, we take the best approaches for these aspects and together with the aspect of timeliness of documents, define a probabilistic model that considers a combination of these aspects. Considering these aspects again, we lastly propose a stochastic model (a biased, personalized-like PageRank algorithm) for analyzing a graph defined by the query entities and scoring its nodes. The experimental results on a new evaluation dataset show the effectiveness of the proposed models and allow us to understand their limitations.

Acknowledgements

This thesis appears in its current form by taking the assistance and guidance from several people. I would, therefore, like to offer my sincere thanks and gratitude to all of them.

First, I would like to sincerely thank my examiners Prof. Dr. Eirini Ntoutsi and Prof. Dr. Wolfgang Nejdl for their substantial guidance and giving their valuable feedback in order to enhance the thesis.

Furthermore, I would like to express my highest gratitude towards my supervisor Dr. Pavlos Fafalios, L3S Research Center, for supervising my thesis. His advice was essential for the completion of my thesis. Our countless meetings were fruitful and productive, and it was a pleasure to be able to work with him. Dr. Fafalios supported me in every detail of theoretical and technical aspects till the very end. My special thanks goes to him.

Finally, I wish to thank my family and friends for their constant support and encouragement throughout my thesis.

Chapter 1

Introduction

Archiving is the process of preservation of records permanently or for long-terms on grounds of their enduring cultural, evidential or historical value. As the web continues to grow we have records holding significant value being constantly produced and consumed every day. However, due to transient nature of the Web, most of these records produced become either lost or unavailable after a short period of time. Archives capture portions of these record collections for sociologists, historians and other interested parties for whom these collections are immensely valuable.

Despite the increasing number of digital archives worldwide (like newspaper and web archives), the absence of efficient and meaningful exploration methods still remains a major bottleneck in the way of turning them into usable information sources [9]. In our previous work[19], we proposed building semantic profiles/layers that describe information about the contents of the archived documents and could be used as a method to solve the problem of archive exploration. Specifically, we used Semantic Web technologies as a base and constructed an RDF/S vocabulary that allowed for:

a) describing useful metadata information about each archived document.

b) annotating each document with entities and events detected in its textual contents.

c) enriching the detected entities/events[1] with more semantic information (like properties and related entities coming from the LOD).

d) publishing all this data on the Web in the standard RDF format (making thereby all

[1]From now on, when we say "entities" we refer to both entities, events and concepts.

this information directly accessible and exploitable by other systems and tools).

A Semantic Layer allows running advanced queries which combine metadata of the documents (like publication date) and content-based semantic information (like entities mentioned in the documents). We have identified the following information needs that a semantic layer is able to satisfy:

i) *Information Exploration.* Exploring documents about entities from the past in a more advanced and "exploratory" way, e.g., even if we do not know the entity names related to our information need. For example, we can find articles of a specific time period discussing about a specific category of entities (e.g., *politicians*) or about entities sharing some characteristics (e.g., *born in Germany between 1980 and 1990*).

ii) *Information Integration.* Exploring web archives by also integrating information from existing knowledge bases. For example, we can find articles discussing about some entities and for each entity to also retrieve and show some characteristics (e.g., an image or a description in a specific language). Cross-domain knowledge bases like DBpedia contain such properties for almost every popular entity. Moreover, we can directly integrate information coming from multiple web archives. For example, we can combine information from a news archive and a social media archive.

iii) *Information Inference.* Inferring knowledge by exploiting the contents of a web archive. For example, we can identify important time periods related to one or more entities. Vice-versa, we can find out the most popular entities of a specific type in a specific time period (e.g., most discussed *politicians* in articles of *2000*). Or we can understand the topic of a web page (e.g., find news articles related to *medicine*).

iv) *Robustness (in information change).* Exploring a web archive by automatically taking into account the change of entities over time. For example, the company *Accenture* was formerly known as *Andersen Consulting*, or the city *Saint Petersburg* was previously named *Leningrad*. Such temporal reference variants are common in the case of high impact events, new technologies, role changes, etc. We can find documents from the past about such entities without having to worry about their correct reference.

v) *Multilinguality.* Exploring documents about entities from the past independently of the document language (and thus of the language of the entity name). For instance, *abortion* is *Avortement* in French and *Schwangerschaftsabbruch* in German. We can find such documents about entities without having to worry about the document and

entity language.

vi) *Interoperability.* Facilitating exploitation of web archives by other systems. We can expose information about web archives in a standard and machine understandable format, that will always be available on the Web, and that will allow for easy information integration. This avoids downloading and parsing the entire web archive for identifying an interesting part of it related to a time period, some metadata values and/or some entities. For example, we can gather a corpus of articles of *2004* discussing about *Indian politicians*.

Such advanced information needs can be directly expressed through structured (SPARQL) queries or through user-friendly interactive interfaces which transparently transform user interactions to SPARQL queries (e.g., Faceted Search-like browsing interfaces [49]).

However, the results returned by such structured queries can be numerous and moreover they all equally match the query (there is no relevance ranking like in the case of keyword-based information retrieval). Thus, there arises the need for an effective method to rank the returned results for discovering and showing to the users the most important ones. For instance, when requesting articles from a news archive published within a specific time period and mentioning one or more query entities, important documents may be those whose main topic is about an important event related to the query entities during the requested time period. Thus, an effective ranking method should consider the different factors that affect the importance of documents to the query, while at the same time relying only on the data available in the semantic layer (there is no access to the full contents of the documents).

Although there is a plethora of works on ranking archived documents for keyword-based temporal queries and ranking in knowledge graphs, the problem of ranking such documents for the case of structured queries on knowledge graphs has not yet been recognized and studied. In this thesis, we address this gap by first introducing and formalizing this type of problem. Then, to cope with this problem, we propose two ranking models (a probabilistic one and a Random Walk-based one) which jointly consider the following aspects:

i) the *relativeness* of a document to the query entities.

ii) the *timeliness* of document's publication date.

iii) the temporal *relatedness* of the query entities to other entities mentioned in the documents.

The idea is to promote documents that mention the query entities many times, that have been published in important (for the query entities) time periods, and that mention many other entities co-occurring frequently with the query entities in important time periods. For example, in case we want to rank articles of 1990 discussing about *Nelson Mandela*, we want to favor articles that

i) discuss about *Nelson Mandela* as their main topic.

ii) have been published in important time periods for *Nelson Mandela* (e.g., February 1990 since during that period he was released from prison).

iii) mention other entities that seem to be important for *Nelson Mandela* during important time periods (e.g., *F. W. de Klerk* who was South Africa's State President in February 1990).

This ranking becomes more complicated when we need to rank articles discussing about multiple entities. Similar to the previous example, consider that we want to rank articles between 1993 to 1994 discussing about *Nelson Mandela* or *F. W. de Klerk*. In this case we would want to favor articles that

i) discuss both *Nelson Mandela* and *F. W. de Klerk* rather than just one of *Nelson Mandela* or *F. W. de Klerk* as their main topic.

ii) have been published in important time periods for both of them (e.g., *Nelson Mandela* becoming president in May 1994 was a more important time period for *Nelson Mandela* than *F. W. de Klerk*, but the Nobel Peace Prize jointly awarded to *Nelson Mandela* and *F. W. de Klerk* for their fight against Apartheid in December 1993 was an important time period for both *Nelson Mandela* and *F. W. de Klerk*).

iii) mention other entities that seem to be important for both *Nelson Mandela* and *F. W. de Klerk* rather than just one of *Nelson Mandela* or *F. W. de Klerk* (e.g. *African National Congress* is a more important entity for *Nelson Mandela* rather than *F. W. de Klerk* as *Nelson Mandela* was a key member of this political party. However, *Apartheid* can be considered an important entity for both *Nelson Mandela* and *F. W. de Klerk* as both struggled to eradicate it from South Africa).

In a nutshell, in this thesis we make the following contributions:

- We formulate and formalize the problem of ranking archived documents for structured queries over semantic graphs.

- Due to lack of evaluation datasets for this problem, we have created a new ground truth dataset for a news archive which we make publicly available.
- We define several approaches to describe the aspects of relativeness of documents to entities and the temporal relations among the entities. Based on performance, we take the best approaches for these aspects and together with the aspect of timeliness of documents, define a probabilistic model that considers a combination of these aspects. Considering these aspects again, we lastly propose a stochastic model (a biased, personalized-like PageRank algorithm) for analyzing a graph defined by the query entities and scoring its nodes.
- We present the results of an experimental evaluation which illustrate the effectiveness of the proposed models. We also analyze problematic cases for understanding when and why the models fail to provide good rankings.

The rest of the thesis is organized as follows: Chapter 2 defines the problem and describes a new evaluation dataset. Chapter 3 presents the required background and related literature. Chapter 4 introduces the probabilistic model. Chapter 5 introduces a stochastic model (Random Walk with Restart). Chapter 6 presents evaluation results. Finally, Chapter 7 concludes the report and discusses interesting directions for future research.

Chapter 2

Problem Definition and Evaluation Dataset

In this section, we formalize the problem of ranking documents returned by structured queries (SPARQL) and describe a new ground truth dataset for the problem at hand. First we introduce the required notions and notations.

2.1 Notions and Notations

Entities

In our problem, an *entity* is anything with a separate and meaningful existence that also has an identity expressed through a reference in a knowledge base (e.g., a Wikipedia/DBpedia URI). This does not only include persons, locations, organizations, etc., but also events (e.g., *US 2016 presidential election*) and more abstract concepts such as *democracy* or *abortion*. Let E be a finite set of entities, e.g., all Wikipedia entities, where each entity $e \in E$ is associated with a unique URI in the reference knowledge base.

Documents and extracted entities

Let D be a set of documents (e.g., a set of news articles) published within a set of time periods T_D of fixed granularity Δ (e.g., day). For a document $d \in D$, let $t_d \in T_D$ be the

time period of granularity Δ in which d was published, while for a time period $t \in T_D$, let $docs(t) \subseteq D$ be the set of all documents published within t, i.e., $docs(t) = \{d \in D \mid t_d = t\}$. Let also $ents(d) \subseteq E$ be all entities mentioned in d extracted using an entity linking system. Inversely, for an entity $e \in E$, let $docs(e) \subseteq D$ be all documents that mention e, i.e., $docs(e) = \{d \in D \mid e \in ents(d)\}$.

2.2 Problem Definition

Given a corpus of documents D, a set of entities $E_D \in E$ mentioned in documents of D, and a SPARQL query Q requesting documents from D published within a *time period* $T_Q \subseteq T_D$ and related to one or more query entities $E_Q \subseteq E_D$ with logical `AND` (mentioning all the query entities) or `OR` (mentioning at least one of the query entities) semantics, the problem is how to rank the documents $D_Q \subseteq D$ that match Q.

Listing 2.1 shows an example SPARQL query requesting documents published in 1990 discussing about the entities *Nelson Mandela* and *Frederik Willem de Klerk* (logical `AND` semantics), while the query in Listing 2.2 requests articles of 1990 discussing about *state presidents of South Africa* (logical `OR` semantics). Our objective is to rank the results returned by such SPARQL queries.

2.3 Evaluation Dataset

Due to the lack of benchmark datasets for our problem, and to enable empirical evaluations, we have created a new ground truth dataset. We used the New York Times (NYT) annotated corpus [44] as the underlying document collection. The corpus contains over 1.8 million articles published by NYT between 1987 and 2007. We used Babelfy [39] for extracting DBpedia entities from each article, using a configuration proposed by the Babelfy developers[1]. We tested this configuration in the AIDA/CONLL-Test B ground truth dataset [26] and got the following evaluation scores: micro precision: 0.818, micro recall: 0.684, micro F1: 0.745. Based on these annotations, we constructed a semantic layer following the process described in [19]. Then, we created 24 SPARQL queries, each

[1]The configuration is available at: `https://github.com/dice-group/gerbil/blob/master/src/main/java/org/aksw/gerbil/annotator/impl/babelfy/BabelfyAnnotator.java`

```
SELECT DISTINCT ?article WHERE {
  ?article dc:date ?date FILTER(year(?date) = 1990) .
  ?article oae:mentions ?entity1, ?entity2 .
  ?entity1 oae:hasMatchedURI  dbr:Nelson_Mandela .
  ?entity2 oae:hasMatchedURI  dbr:F._W._de_Klerk }
```

Listing 2.1: SPARQL query for retrieving articles of 1990 discussing about *Nelson Mandela* and *Frederik Willem de Klerk* (logical `AND` semantics).

```
SELECT DISTINCT ?article WHERE {
  SERVICE <http://dbpedia.org/sparql> {
    ?p dc:subject dbc:State_Presidents_of_South_Africa> }
  ?article dc:date ?date FILTER(year(?date) = 1990) .
  ?article oae:mentions ?entity .
  ?entity oae:hasMatchedURI  ?p }
```

Listing 2.2: SPARQL query for retrieving articles of 1990 discussing about *state presidents of South Africa* (logical `OR` semantics).

one requesting articles published in a specific time period and mentioning one or more entities. The queries are grouped into 4 categories:

- **Single-entity queries (Q1-Q6):** 6 queries requesting articles related to 1 entity (e.g., articles of 1990 discussing about *Nelson Mandela*).
- **Multiple-entity `AND` queries (Q7-Q12):** 6 queries requesting articles related to 2 or more entities with logical `AND` semantics (e.g., articles of 1990 discussing about *Nelson Mandela* and *F.W. de Klerk*).
- **Multiple-entity `OR` queries (Q13-Q18):** 6 queries requesting articles related to 2 or more entities with logical `OR` semantics (e.g., articles of 1990 discussing about *Nelson Mandela* or *F.W. de Klerk*).
- **Category queries (Q19-Q24):** 6 queries requesting articles related to entities belonging to a DBpedia category (e.g., articles of 1990 discussing about *presidents of South Africa*[2]).

We manually evaluated all the results returned by these queries (773 results totally) using

[2]Entities that have the value `<http://dbpedia.org/resource/Category:Presidents_of_South_Africa>` in their subject property (`<http://purl.org/dc/terms/subject>`).

a graded relevance scale (from 0 to 3), following the criteria described below:

- **Score 0**: The document has almost nothing to do with the query entities.
- **Score 1**: The topic of the document is **not** about the query entities, however the query entities are related to the document context.
- **Score 2**: The topic of the document is **not** about the query entities, however the query entities are important for the document context.
- **Score 3**: The topic of the document is about the query entities and discusses something important about them.

Table 2.1 contains the list of information needs and the significant events that we identified from Wikipedia and the other criteria that we used for evaluating our ground truth dataset. Table 2.2 shows the number of results per relevance score for each of the queries. The semantic layer, the SPARQL queries, and the relevance scores (together with explanations for the provided scores) are publicly available[3].

Table 2.1: List of information needs and the significant events identified from Wikipedia and other criteria used in the evaluation.

#	Information Need	Significant Events/Other Evaluation Criteria
1	Find articles between 6-15/2/1990 about Nelson Mandela	• Mandela's release from Victor Vester Prison on 11 February 1990
2	Find articles between 15/2/1987 and 30/4/1987 about Fidel Castro	• Fidel Castro's increasing support for Soviet Union and Socialism • Deteriorating relations of Cuba with the United States
3	Find articles between 15/10/1988 and 31/12/1988 about the rock band Beatles	• Induction in Rock and Roll Hall of Fame in 1988
4	Find articles between 1/7/1991 and 30/9/1991 about the video game company Nintendo	• Release of game console Nintendo NES in North America on 23 August 1991
5	Find articles between 1-10/1990 and 31/10/1991 about United Airlines	• Buyout of parent of United Airlines • United Airlines placing largest ever order for jets at that time with Boeing • United Airlines winning deal for starting routes from five cities in United States to London
6	Find articles of January-February 1991 about Alzheimers Disease	• Research on genes to study causes of Alzheimers disease • Articles underlying other aspects of the Alzheimers disease
7	Find articles between 10/11/1988 and 18/12/1988 discussing about former US President George H.W. Bush and former Soviet Union President Mikhail Gorbachev	• Gorbachev's visit to the US to meet Bush and Reagan • Assurance of Soviet troop reduction to consolidate diplomacy with US by Mikhail Gorbachev
8	Find articles between 15/07/1987 and 31/07/1991 about former Indian Prime Minister Rajiv Gandhi and the Tamil Tigers, the militant organization involved in the Sri Lankan Civil War	• Signing of India-Sri Lanka Accord bringing truce to Sri Lankan Civil War on 29 July 1987 • Assassination of Rajiv Gandhi on 21 May 1991

Continued on next page

[3]Available at: `https://www.dropbox.com/s/dzkgvwik3akfk5r/evaluation_dataset.zip?dl=0`

Table 2.1 – continued from previous page

#	Information Need	Significant Events/Other Evaluation Criteria
9	Find articles between 11/11/1989 and 10/2/1990 discussing about German reunification and the Berlin Wall	• Fall of the Berlin Wall on 9 November 1989
10	Find articles between 1987 and 1991 about Hollywood actors Robert de Niro and Sean Connery	• Release of the Academy Award winning film The Untouchables on 3 June 1987
11	Find articles between March-June 1987 about US President Ronald Reagan and Iran Supreme Leader Ayatollah Khomeini	• Iran-contra affair in which the Reagan administration supplied the profits obtained from selling arms to Khomeini regime to the Nicaraguan rebels
12	Find articles between January-May 1990 about State of Israel and Hezbollah	• Israeli air raids against Hezbollah group • Hezbollah leader threatening not to release hostages unless US Senate revoked declaration of Jerusalem as capital of Israel • Israel deciding not to release Shiite prisoners as demanded by the Hezbollah group for American hostages release and wanting in return its three Israeli soldiers captured by the Hezbollah group
13	Find articles between 1/4/1989 and 31/12/1989 about King Fahd of Saudi Arabia or Sheikh Zayed, Founder of UAE	• Both kings played significant role in Taif Agreement to end Lebanon Civil War negotiated in Saudi Arabia on 22 October 1989 and ratified in Lebanese Parliament on 5 November 1989
14	Find articles between 1/7/1989 and 1/12/1989 about Pablo Escobar, Colombian Drug Lord and Head of the Medellin Cartel or Colombian Presidents	• Assassination of Colombian Presidential Candidate Luis Carlos Galan on 18 August 1989 • Assassination attempt of Colombian President César Gaviria (Avianca Airlines Flight 203) on 27 November 1989
15	Find articles between 1989 and 1991 discussing about the wrestler Hulk Hogan or WrestleMania Event	• Wrestlemania V on 2 April 1989: Hulk Hogan won the title defeating Randy Savage • Wrestlemania VI on 1 April 1990 in which Hulk Hogan lost the title to The Ultimate Warrior • Wrestlemania VII on 24 March 1991 in which Hulk Hogan won the title defeating Sgt. Slaughter • Hulk Hogan's involvement in Steroid Scandal during the time period
16	Find articles between 1990 and 1991 about Formula One drivers Aryton Senna, Nelson Piquet or Alain Prost	• Aryton Senna winning the 1990 Formula One World Champion title from Alain Prost on 21 October 1990
17	Find articles of September 1989 about Tennis players Steffi Graf or Martina Navratilova	• Graf and Sabatini losing in doubles semi-final of the US Open against Navratilova and Mandlikova • Steffi Graf defeating Martina Navratilova in the US Open finals
18	Find articles between June and December 1989 about UK PM Margaret Thatcher or UK Deputy PM Geoffrey Howe	• Geoffrey Howe's visit to Hong Kong and announcing decision of Margaret Thatcher not to allow Hong Kong residents holding British passport to emigrate to UK after ceding the colony to China • Margaret Thatcher reshuffling her Cabinet after defeat in European Parliament elections and naming Geoffrey Howe as Deputy Prime Minister
19	Find articles between 1/1/1990 and 10/4/1990 mentioning Civilian Astronauts of NASA	• Importance given to mentioning more famous astronauts than others • Importance given to articles mentioning many astronauts • Importance given to articles discussing about an astronaut or a significant event involving an astronaut rather than articles just mentioning an astronaut
20	Find articles between 1/4/1990 and 1/12/1991 mentioning Indian sportspersons who received the Arjuna Award	• Importance given to mentioning more famous sportspersons than others • Importance given to articles mentioning many sportspersons • Importance given to articles discussing about a sportsperson more than articles just mentioning a sportsperson

Continued on next page

Table 2.1 – continued from previous page

#	Information Need	Significant Events/Other Evaluation Criteria
21	Find articles between 1/4/1991 and 1/11/1991 mentioning Williams Formula One drivers	• Importance given to mentioning more famous drivers than others • Importance given to articles mentioning many drivers • Importance given to articles discussing about a driver more than articles just mentioning a driver (e.g. in a result of a race)
22	Find articles between 1-30/6/1987 discussing about famous Indian personalities who got awarded the Bharat Ratna	• Importance given to mentioning more important personalities than others • Importance given to articles mentioning many personalities • Importance given to articles discussing about a personality more than articles just mentioning a personality (e.g. a film of an actor awarded the Bharat Ratna award)
23	Find articles of January 1990 about aviation accidents and incidents of 1990	• Crash of Avianca Airlines 52 from Bogota to New York
24	Find articles between 1/2/1988 and 1/3/1988 discussing about cognitive disorders	• Importance given to articles describing symptoms, causes, treatments and preventive measures of cognitive disorders

Table 2.2: No. of results per relevance score for each query of the evaluation dataset.

Query	Total number of results	Score 0 results	Score 1 results	Score 2 results	Score 3 results
1	65	13	13	4	35
2	28	20	5	1	2
3	28	24	2	0	2
4	23	15	3	1	4
5	38	18	6	7	7
6	24	16	4	1	3
7	29	13	6	8	2
8	61	10	27	15	9
9	42	14	10	10	8
10	28	17	4	6	1
11	27	13	10	4	0
12	24	13	2	8	1
13	30	21	1	4	4
14	37	15	8	8	6
15	27	18	5	2	2
16	23	5	14	2	2
17	21	13	2	2	4
18	25	6	4	9	6
19	41	24	4	8	5
20	22	11	8	3	0
21	29	16	5	5	3
22	47	33	7	3	4
23	31	18	3	8	2
24	23	14	3	3	3
1-6	206	106	33	14	53
7-12	211	80	59	51	21
13-18	163	78	34	27	24
19-24	193	116	30	30	17
Overall (1-24)	773	380	156	122	115

Chapter 3

Background and Related Literature

3.1 Semantic Layer

A Semantic Layer is an RDF repository (RDF graph) of structured data about a collection of archived documents [19]. Structured data includes not only metadata information about a document (like publication date), but also *entity annotations*, i.e., disambiguated entities mentioned in each document extracted using an entity linking system [45]. Figure 3.1 shows an example of (a part of) a Semantic Layer describing metadata and annotation information for a news article of New York Times. We notice that the document was published on January 6, 2012 and mentions the entity name "Federer" which probably corresponds to the known tennis player Roger Federer (with confidence score 0.9). A detailed description of the data models that we used to describe archived documents is provided in Appendix A.

A semantic layer allows running advanced (entity-centric) queries that can also directly integrate information from other knowledge bases (like DBpedia). Listing 3.1 shows an example of a SPARQL query that can be answered by a semantic layer over a collection of old news articles. The query requests articles of 1989 mentioning New York lawyers born in Brooklyn. By accessing DBpedia at query-execution time, the query retrieves the entities that satisfy the query as well as additional information (in particular the birth date of each lawyer). We notice that, by exploiting the entity URIs and type/category information, we can find documents about entities even if we do not know the names of

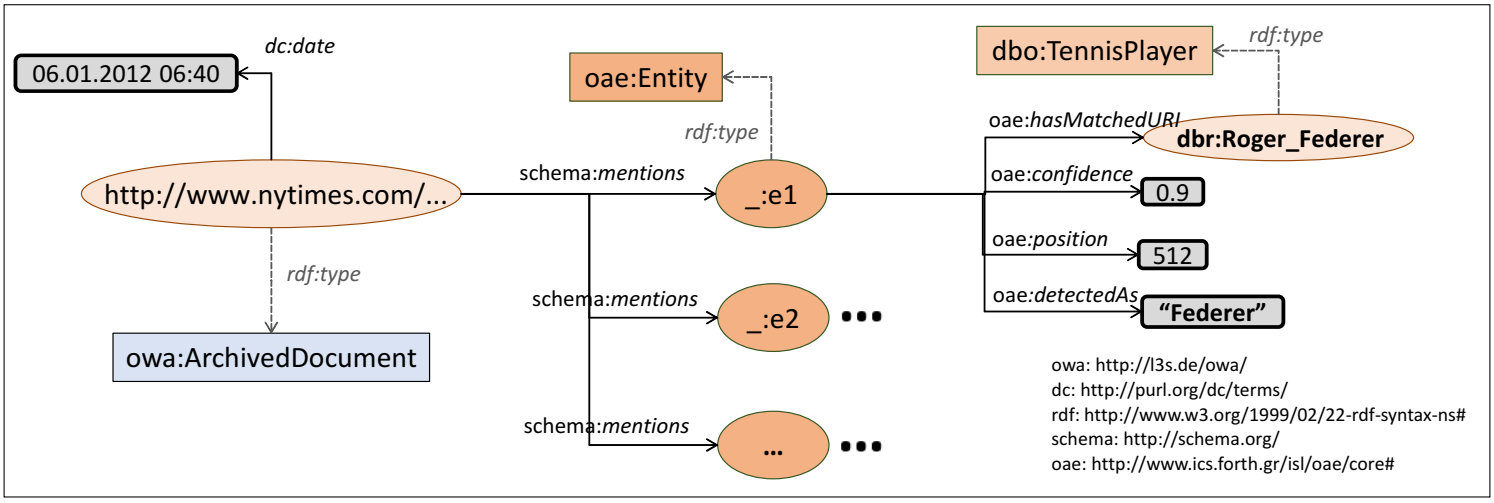

Figure 3.1: A part of a Semantic Layer describing metadata and entity annotations for a news article.

the entities or without needing to specify a long list of all the entity names. Besides, by exploiting the expressive power of SPARQL, we can aggregate information related to documents and entities using operators like `COUNT` and `GROUP BY`.

As shown in our previous work[19], a semantic layer can answer information needs that existing keyword-based systems (like Google news) are not able to sufficiently satisfy. Such advanced (but also common) information needs can be directly expressed through SPARQL queries or by exploiting a user-friendly interface that transforms user interactions to SPARQL queries, like Sparklis [21] or SemFacet [4]. Nevertheless, the results returned by such queries can be numerous and, moreover, they all equally match the submitted query since there is no relevance ranking like in classic keyword-based searching. In this work, we study how we can provide a ranking for the returned documents based on their *"importance"* to the query entities in the requested time period.

3.2 Related Work

Our objective is to rank a set of documents returned by submitting a structured (SPARQL) query on a semantic layer (like the query in Listing 3.1). Below, we first report works on highly-related research areas (*time-aware document ranking* and *ranking in knowledge graphs*) and then we discuss the commonalities and differences of these research areas to the problem we tackle in this thesis.

```
SELECT ?article ?title ?date ?nylawyer ?bdate ?abstr WHERE {
  SERVICE <http://dbpedia.org/sparql> {
    ?nylawyer dc:subject dbc:New_York_lawyers ; dbo:birthPlace dbr:Brooklyn .
    OPTIONAL { ?nylawyer dbo:birthDate ?bdate } }
  ?article dc:date ?date FILTER(year(?date) = 1989) .
  ?article schema:mentions ?entity .
  ?entity oae:hasMatchedURI ?nylawyer .
  ?article dc:title ?title
} ORDER BY ?nylawyer
```

Listing 3.1: An example of a SPARQL query over a Semantic Layer of a collection of old news articles. The query requests articles of 1989 discussing about New York lawyers born in Brooklyn.

Time-aware Document Ranking

The impact of temporal information on information retrieval has received a large share of attention in the last decade. The surveys in [10] and [33] provide a comprehensive categorization and overview of temporal IR approaches and related applications. As regards time-aware *ranking*, existing works are classified into two different types based on two main notions of relevance with respect to time [10, 33]: 1) recency-based ranking, and 2) time-dependent ranking. Since recency-based ranking methods promote documents that are recently created or updated (this preference of freshness is common in general web searching), below we discuss only time-dependent ranking methods which are useful when searching archived collections of documents.

Jin et al. [30] proposed a ranking algorithm to sort results by applying a linear interpolation of text similarity, temporal information, and page importance (based on PageRank), where temporal similarity is the ranking score of temporal relevance based on the set of intersection conditions between the temporal query and the temporal expressions found in the web page. Arikan et al. [5] and Berberich et al. [7] introduced approaches that integrate temporal expressions extracted from the documents into language modeling frameworks. Metzler et al. [37] proposed a time-dependent ranking model to adjust the document scores based on an analysis of web query logs and a set of document fields to estimate the time of both the query and the document. The work by Perkio et al.[11] automatically detects topical trends and their importance over time within a news corpus using a statistic topic model and a simple variant of TF-IDF. These trends are then used as the basis for temporally adaptive rankings. Dakka et al. [12] consider the publication time of documents

to identify the important time intervals that are likely to be of interest to an implicit temporal query. Then, time was incorporated into language models to assign an estimated relevance value to each time period. Aji et al. [1] proposed a term weighting model that uses the revision history analysis of a document to redefine the importance of terms which is then incorporated into BM25 and statistical language models. Kanhabua and Nørvag [32] proposed a time-sensitive ranking model based on learning-to-rank techniques for explicit temporal queries. To learn the ranking model, temporal and entity-based features are applied.

Regarding more recent works, Singh et al. [47] introduced the notion of *Historical Query Intents* and modeled it as a search result diversification task which intends to present the most relevant results from a topic-temporal space. For retrieving and ranking historical documents like news articles, the authors propose a novel retrieval algorithm, called HistDiv, which jointly considers the dimensions of aspect and time. Expedition [46] is a time-aware search system for scholars which allows users to search articles in a news collection by entering free-text queries and choosing from four retrieval models: Temporal Relevance, Temporal Diversity, Topical Diversity, and Historical Diversity. Tempas [28] is a search system for web archives that exploits a social bookmarking service (Delicious) for temporally searching an archive by indexing tags and time. The new version of Tempas [29] exploits temporal link graphs and the corresponding anchor texts. The authors show how temporal anchor texts can be effective in answering queries beyond purely navigational intents, like finding the most central web pages of an entity in a given time period.

Difference of our case

Similar to the above works, our objective is to rank documents for a user information need. However, our case has the following three distinctive characteristics:

i) The full contents of the documents are not available and there are no term-based indexes on top of them. We have access only on the RDF triples existing in the semantic layer, i.e., on metadata about the documents and on the entities mentioned in the documents. Moreover, these entities have been extracted using automated entity linking systems and thus are prone to disambiguation errors.

ii) The information needs are expressed through structured SPARQL queries, not keywords. These SPARQL queries request documents of a specific time period mentioning one or more specific entities, while these query entities are specified

through URIs which means that there is no ambiguity about them.

iii) We already know the documents that match the query, however there are no relevance scores, i.e., all documents *equally* match the query. Our objective is to identify (and rank higher) the documents that discuss important information about the entities given in the query.

Ranking in Knowledge Graphs

There is a plethora of works on ranking entities, concepts and resources in knowledge graphs. The majority of these works exploit the structure of the graph and apply some variation of a popular link analysis algorithm (like PageRank). The survey in [42] formalizes and contextualizes the problem of ranking in the Web of Data and provides an analysis and contrast of the similarities, differences and applicability of the different approaches. Below we discuss some of these works.

Dali et. al.[13] propose a query independent Learning to Rank(LTR) approach for RDF entity search. They use RDF graph extracted features, search engine based features and centrality-based features and compare them to target features. To obtain the ground truth for the target features they use human relevance judgements. Latifi and Nematbaksh[35] use the same approach as that proposed by Dali et. al.[13] but suggest the use of the Information Content(IC) feature to reduce ranking time of the system. Butt et. al.[8] introduced DWRank, a two-staged bi-directional graph walk ranking algorithm for concepts in ontologies. DWRank characterises two features of a concept in an ontology to determine its rank in a corpus, the authoritativeness of the ontology in which it is defined called AuthorityScore and the centrality of the concept to the ontology within which it is defined called HubScore. DWRank then uses a Learning to Rank approach to learn the feature weights for the two ranking strategies.

OntologyRank algorithm by Ding et al. introduced in [15] and mentioned further in [16] finds use in the Semantic web search engine *Swoogle*. It identifies Semantic Web Ontologies(SWOs) in Semantic Web Documents(SWDs) and further ranks terms in an Ontology based on their popularity. AKTiveRank[2] by Alani et. al. is another ontology ranking approach which relies on their importance to a given query and uses the semantic web search engine *Swoogle*[16] to get a list of ontologies that need to be ranked. SemRank[3] designed by Anwanyu et. al. ranks relationships instead of entities. The final ranking is calculated based on the predictability or gain of information of a semantic association, the

degree of similarity of a keyword or property occurring in a semantic association and the amount of differences between the properties in the original schema and the properties that compose a path. Concept-And-Relation-Ranking(CARRank) algorithm proposed by Wu et. al.[51] identifies important relations and concepts in an ontology. CARRank follows a convergant iterative process in which the weights of relations and importance of concepts reinforce one another in an iterative manner and achieves a similar converging speed as PageRank-like algorithms.

Regarding PageRank based approaches, PopRank[40] assigns weights to links among Web Objects depending on the relationship types between objects. This system extracts a subset of the graph based on domain and then assigns link weights to the subgraph using Ranking Lists made by domain experts. ReConRank[27] performs dynamic ranking and only analyses the result data that matches the user query. It considers the ratio of all the links received as in-links in order to reduce the number of iterations and allows users to fine tune the weights according to their choice. It goodness relies on the relationships between resources and their provenance and faces challenges like extraction of topical graph and increase in query time due to dynamic ranking. Harth et. al.[24] use a variation of the PageRank algorithm for ranking in which they assign weights to all links in the graphs based on authority or provenance of data source and calculate PageRank for the whole graph even before the query is entered. This approach is in complete contrast to ReConRank[27] approach. RareRank[50] is another PageRank-like algorithm where the random component in the PageRank model is replaced by a more deterministic component based on the domain of search in order to reduce randomness in a graph. DBPediaRanker[38] tries to first dereference and explore all nodes in the DBPedia graph belonging to the same domain given a query. Then by checking whether a strong relation exists between two resources, it creates a contextualized weighted graph with the weights of links between two resources based on the similarity between nodes. YAGO-NAGA introduced in Kasneci et. al.[34] and extended to include keyword based search in Elbassouni et. al.[17] is a semantic search system based on Language Model that performs ranking based on the notions of Informativeness, Compactness and Confidence. The computation is done through a PageRank like algorithm and it makes use of the provenance of information. DING[14] is another adaptation of the PageRank algorithm which calculates rank in three steps. First it calculates the global dataset rank, then the entity rank and finally the global ranking which is a combination of the of both the global dataset rank and the entity rank. Fafalios and Tzitzikas[20] integrate classical Web with the Web of Linked Data to provide users with semantic context and thus help users save time in exploratory search scenarios. For the top-100 results from BING

search engine for keyword search query, the system detects the entities in the snippets of the results. For the top-k entities derived from a PageRank like algorithm, the system then using the information available on the LOD Cloud tries to show the user how the top detected entities are related.

Considering link-based approaches other than PageRank adaptations, NOC-ORDER[22] introduced by Graves et. al. ranks nodes in an RDF graph based on centrality feature. The algorithm is an adaptation of *All-Pairs Shortest Path* algorithm for RDF graph and tries to rank nodes based on the connectedness and distance of each node to the other nodes.

An interesting related line of research tackles the problem of *ad-hoc object retrieval* [41, 48]. In this problem, the input is a keyword query and the output is one or more resources (entity URIs) that satisfy the corresponding information need. To tackle this kind of problem, Pound et. al.[41] proposed an adaptation of TF-IDF, while Tonon et. al.[48] combined an inverted index with entity graph traversal. Pound et. al.[41] also proposed an evaluation protocol and tested a number of metrics for their stability and discriminating power. In the same context, the SemSearch challenge [23] focused on finding the entity identifier of a specific entity described by a user query, while the TREC entity track [6] studied two related search tasks: i) finding all entities related to a given entity, and ii) finding entities with common properties given some examples.

Difference of our case

This thesis is also focused on ranking in knowledge graphs similar to the above mentioned works. However, the following differences distinguish this thesis from the above works:

i) In all these works, the result is a ranked list of resources (like entities, properties or triples) from an RDF data collection in response to a keyword query. The work by Anwanyu et. al.[3] ranks relationships, the works by Ding et. al.[15, 16] and Alani et. al.[2] focus on ranking ontologies and the works by Wu et. al.[51] and Butt et. al.[8] rank concepts within an ontology. Thus, in contrast to our problem, the units of retrieval are resources in general (like entity URIs), not archived documents.

ii) These works operate over knowledge graphs like DBpedia, where several entities are described through properties and associations with other entities. A semantic layer

is a special kind of a knowledge graph that represents metadata and annotation information about a set of unstructured documents like news articles. Given a non-ambiguous SPARQL query and its result (i.e., the documents that match the query), our aim is to identify those documents that discuss important information about the query entities in the requested time-period, by exploiting only the contents of the semantic layer.

Chapter 4

Probabilistic Modeling

The question is: *"what makes an archived document important given a time period and one or more query entities?"* We have identified the following aspects:

- the *relativeness* of a document with respect to the query entities (the document should talk about the query entities, ideally as its main topic).
- the *timeliness* of a document with respect to its publication date (the document should have been published in a time period which is important for the query entities).
- the *relatedness* of a document with respect to its reference to other entities (the document should discuss the relation of the query entities with other entities that are important for the query entities in important time periods).

The idea is to promote documents that: i) mention the query entities many times in their contents (because then, the topic of the document may be about these entities), ii) have been published in important (for the query entities) time periods, and iii) mention many other entities that co-occur frequently with the query entities in important time periods. For example, in case we want to rank articles of 1990 discussing about *Nelson Mandela*, we want to favor articles that i) discuss about *Nelson Mandela* as their main topic, ii) have been published in important (for *Nelson Mandela*) time periods (e.g., February of 1990 since during that period he was released from prison), and iii) mention other entities that seem to be important for Nelson Mandela during important time periods (e.g., *Frederik Willem de Klerk* who was South Africa's State President in 1990).

4.1 Relativeness

We consider that if the query entities are mentioned multiple times within a document, the document should receive a high score since the document's topic may be about these entities. We use three different approaches to define relativeness: the first approach considers just the entity frequency without the position of the entities inside the document, the second and third approach take the position of the entities inside the documents as well into consideration but differ in terms of the decrease of the importance score of the entities with increasing position. The second approach is based on exponential decay of the importance score of the entities with position, whereas the third approach is based on linear decay of the importance score of the entities with position.

4.1.1 Entity Frequency Based (without Entity Position)

The term frequency (in our case entity frequency) is a classic numerical statistic that is intended to reflect how important a word (entity) is to a document in a collection or corpus [36].

We first define a *relativeness* score of a document $d \in D_Q$ based on the *frequency* of the query entities in d. First, let $count(e, d)$ be the number of occurrences of e in document d. For the case of `AND` semantics ("$\wedge$"), the score is defined as:

$$score^f_\wedge(d, E_Q) = \frac{\sum_{e \in E_Q} count(e, d)}{\sum_{e' \in ents(d)} count(e', d)} \tag{4.1}$$

Notice that the score of a document will be 1 if it contains the query entities and no other entity. For the case of `OR` semantics ("$\vee$"), we can also consider the number of query entities mentioned in the document (since a document does not probably contain all the query entities as in the case of `AND` semantics). In this case, the *relativeness* score can be defined as follows:

$$score^f_\vee(d, E_Q) = \frac{\sum_{e \in E_Q} count(e, d)}{\sum_{e' \in ents(d)} count(e', d)} \cdot \frac{|ents(d) \cap E_Q|}{|E_Q|} \tag{4.2}$$

where $\frac{|ents(d) \cap E_Q|}{|E_Q|}$ is the percentage of query entities discussed in the document. The score of a document will be 1 if it contains all the query entities and no other entity. This formula favors documents mentioning multiple times many of the query entities.

Now, the probability of a retrieved document $d \in D_Q$ given only the query entities can be defined as:

$$P(d|E_Q) = \frac{score^f(d, E_Q)}{\sum_{d' \in D_Q} score^f(d', E_Q)} \tag{4.3}$$

4.1.2 Exponential Decay with Entity Position

A document should be considered more important than another document if query entities in the the first document are more at the beginning of the document and the query entities in the second document are more in the middle or end of the document. We consider the position of entities in the documents and modify the *relativeness* score of the documents for `AND` semantics as follows:

$$score^f_{\wedge}(d, E_Q) = \frac{\sum_{e \in E_Q} (\sum_{p \in P_{e,d}} \exp(-a \cdot p))}{\sum_{e' \in ents(d)} (\sum_{p \in P_{e',d}} \exp(-a \cdot p))} \tag{4.4}$$

For the case of `OR` semantics, we can also consider the number of query entities mentioned in the document.

$$score^f_{\vee}(d, E_Q) = \frac{\sum_{e \in E_Q} (\sum_{p \in P_{e,d}} \exp(-a \cdot p))}{\sum_{e' \in ents(d)} (\sum_{p \in P_{e',d}} \exp(-a \cdot p))} \cdot \frac{|ents(d) \cap E_Q|}{|E_Q|} \tag{4.5}$$

In 4.4 and 4.5, a denotes the rate factor of the negative exponential function, $P_{e,d}$ denotes the set of all positions of an entity e in a document $d, d \in D_Q$. The modeling of the importance score as a negative exponential function is based on the notion that the attention of the user tends to decrease rapidly as he moves across the document. For example, a historian looking for an important document related to an entity may not read the complete document if he does not find the entity and the content he is looking for at the beginning of the document. The rate factor of the negative exponential function decides the rate of decay of the negative exponential function.

4.1.3 Linear Decrease with Entity Position

In this approach, we model the *relativeness* score as a linearly decreasing function with the entity position. Since we do not know about the length of the document, we consider that the importance converges to zero at the last position at which any entity is detected

in a document. In this approach, we define the *relativeness* score of the documents for AND semantics as follows:

$$score^{f}_{\wedge}(d, E_Q) = \frac{\sum_{e \in E_Q} (\sum_{p \in P_{e,d}} (1 - \frac{p}{maxP_d}))}{\sum_{e' \in ents(d)} (\sum_{p \in P_{e',d}} (1 - \frac{p}{maxP_d}))} \tag{4.6}$$

For the case of OR semantics, similar to the first and second approach for calculating relativeness, we can also consider the number of query entities mentioned in the document.

$$score^{f}_{\vee}(d, E_Q) = \frac{\sum_{e \in E_Q} (\sum_{p \in P_{e,d}} (1 - \frac{p}{maxP_d}))}{\sum_{e' \in ents(d)} (\sum_{p \in P_{e',d}} (1 - \frac{p}{maxP_d}))} \cdot \frac{|ents(d) \cap E_Q|}{|E_Q|} \tag{4.7}$$

Similar to the last approach, in 4.6 and 4.7, $P_{e,d}$ denotes the set of all positions of an entity e in a document $d, d \in D_Q$. P_d denotes the set of all positions of all entities in a document d, that is, $ents(d)$ where $d \in D_Q$. P_d is just the superset of all $P_{e,d}$. $maxP_d$ denotes the maximum element of the set P_d. The modeling of *relativeness* as a linearly decreasing function would mean that the attention of the user decreases slowly as he reads the document and reduces to zero only when he has almost reached the end of the document. This approach contrasts with the previous approach as it assumes that a user looking for an important document related to an entity reads the complete document even if he does not find the entity and the content he is looking for at the beginning of the document.

4.2 Timeliness

We consider that a time period $t \in T_Q$ is important for the entities in E_Q, if there is a relatively large number of documents in D_Q discussing about these entities during t. For example, many articles about *Nelson Mandela* were published the period 11-13 of February 1990 because in February 11 *Nelson Mandela* was released from prison. Thus, articles published during that period should be promoted since they are probably related to this important event of *Nelson Mandela*'s life.

For the case of AND semantics, we define the following importance score of a *time period*

$t \in T_Q$:

$$score^t_{\wedge}(t) = \frac{|docs(t) \cap D_Q|}{|D_Q|} \tag{4.8}$$

This scoring formula favors time periods in which there is a large number of documents discussing about the query entities.

For the case of `OR` semantics, in a time period t there may be a large number of documents discussing only for one of the query entities, while in another time period t' there may be a smaller number of documents discussing though for many of the query entities. For also taking into account the number of query entities discussed in documents of a specific time period, we consider the following formula:

$$score^t_{\vee}(t) = \frac{|docs(t) \cap D_Q|}{|D_Q|} \cdot N(E_Q, t) \tag{4.9}$$

where, $N(E_Q, t)$ is the average percentage of query entities discussed in articles of t, i.e.:

$$N(E_Q, t) = \frac{\sum_{d \in docs(t) \cap D_Q} \frac{|ents(d) \cap E_Q|}{|E_Q|}}{|docs(t) \cap D_Q|} \tag{4.10}$$

Now, the probability of a retrieved document $d \in D_Q$ given only its publication date t_d can be defined as:

$$P(d|t_d) = \frac{score^t(t_d)}{\sum_{d' \in D_Q} score^t(t_{d'})} \tag{4.11}$$

4.3 Relatedness

Entities that are co-mentioned frequently with the query entities in important time periods are probably important for them. For example, *Apartheid* was an important concept related to *Nelson Mandela* during 1990, thus articles discussing for both *Apartheid* and *Nelson Mandela* should be promoted. However, there may be also some general entities (e.g., *South Africa* in our example) that co-occur with the query entities in almost all documents (independently of the time period). Thus, we should also avoid over-emphasizing documents mentioning such "common" entities. We define *relatedness* using three approaches: the first approach considers the query entities and related entities without considering the position of these entities inside the documents, the second and the third approach consider the position of the query entities and the related entities inside

the documents as well. The second approach and third approach differ, however, in the way of assigning importance score based on the difference in position between the query entity and the related entity.

4.3.1 Entity Frequency Based (without Entity Position)

For the case of `AND` semantics, we consider the following *relatedness* score for an entity $e \in E_D \setminus E_Q$:

$$\begin{aligned} score^r_{\wedge}(e) &= idf_{\wedge}(e) \cdot \sum_{t \in T_Q} (score^t_{\wedge}(t) \cdot \frac{|docs(t) \cap D_Q \cap docs(e)|}{|docs(t) \cap D_Q|}) \\ &= idf_{\wedge}(e) \cdot \sum_{t \in T_Q} \frac{|docs(t) \cap D_Q \cap docs(e)|}{|D_Q|} \end{aligned} \tag{4.12}$$

where $idf_{\wedge}(e)$ is the inverse document frequency of entity e in the set of documents discussing about the query entities in the entire corpus, which can be defined as follows:

$$idf_{\wedge}(e) = 1 - \frac{|docs(e) \cap (\cap_{e' \in E_Q} docs(e'))|}{|\cap_{e' \in E_Q} docs(e')|} \tag{4.13}$$

The formula considers the percentage of documents in which the entity co-occurs with the query entities in important time periods.

For the case of `OR` semantics, the above formula does not consider the number of different query entities discussed in documents together with the entity e. To also handle this aspect, we consider the following *relatedness* score for the case of `OR` semantics:

$$\begin{aligned} score^r_{\vee}(e) &= idf_{\vee}(e)\ N(E_Q, e) \sum_{t \in T_Q} (score^t_{\vee}(t) \frac{|docs(t) \cap D_Q \cap docs(e)|}{|docs(t) \cap D_Q|}) \\ &= idf_{\vee}(e)\ N(E_Q, e) \sum_{t \in T_Q} (N(E_Q, t) \frac{|docs(t) \cap D_Q \cap docs(e)|}{|D_Q|}) \end{aligned} \tag{4.14}$$

where $N(E_Q, e)$ is the average percentage of query entities discussed in articles together with entity e, i.e.:

$$N(E_Q, e) = \frac{\sum_{d \in docs(e) \cap D_Q} \frac{|ents(d) \cap E_Q|}{|E_Q|}}{|docs(e) \cap D_Q|} \tag{4.15}$$

Now the inverse document frequency $idf_{\vee}(e)$ includes documents mentioning at least one of the query entities, i.e.:

$$idf_{\vee}(e) = 1 - \frac{|docs(e) \cap (\cup_{e' \in E_Q} docs(e'))|}{|\cup_{e' \in E_Q} docs(e')|} \tag{4.16}$$

This formula favors related entities that i) co-occur frequently with many of the query entities, ii) are discussed in documents published in important (for the query entities) time periods.

Now, the probability of a document $d \in D_Q$ given only other entities mentioned in the retrieved documents (E_{D_Q}) can be defined as:

$$P(d|E_{D_Q}) = \frac{\sum_{e \in ents(d) \setminus E_Q} score^r(e)}{\sum_{d' \in D_Q} \sum_{e' \in ents(d') \setminus E_Q} score^r(e')} \tag{4.17}$$

4.3.2 Closest Distance between Entities

In this approach, when we model the *relatedness* score of a document, we also keep in mind the difference in position between the query entities and the related entities. We define a measure called *proximity* score for each related entity which is based on the position difference of the related entity to the query entity.

Consider a document in which the red lines denote the positions of a query entity and blue lines denote the positions of a related entity along the length of the document as shown in Figure 4.1.

We calculate the closest distance by checking each red line(position of the query entity) in the document. If the red line that we are currently at, is preceded by a blue line and succeeded by a red line (or no line), or preceded by another red line (or no line) and succeeded by a blue line, then we take $Dist(e, e')$ as the the absolute difference in position between current red line and the blue line that precedes or succeeds the current red line. However, if the red line that we are currently at is preceded and succeeded by blue lines, then we take $Dist(e, e')$ as the average of the absolute difference in position between the current red line and the each blue line that precedes and succeeds the current red line. In the last case, if we are currently at a red line which is preceded and succeeded by red lines as well, we do nothing and move to check the next red line. The $avgDist(e, e')$ obtained at the end is just an average of all the $Dist(e, e')$ in the document d.

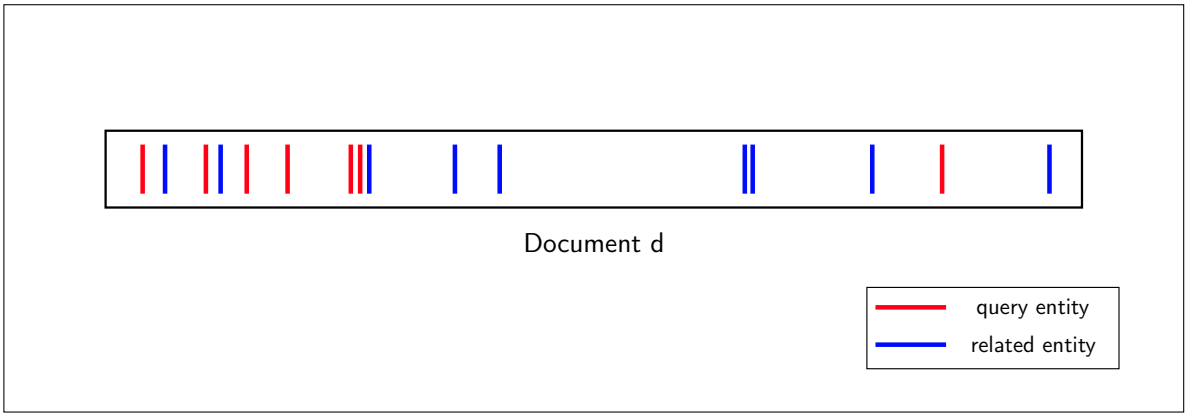

Figure 4.1: Positions of a query entity and a related entity inside a document d visualized across its length.

4.3.3 Average Distance between Entities

The modeling of the *relatedness* score considering average distance between entities is similar to the previous approach in that we consider the difference in position between the query entities and the related entities in this approach as well. We make use of the *proximity* score measure in this approach also but it changes from the previous approach as the way we calculate $avgDist(e, e')$ in this approach changes.

Consider the Figure 4.1 which depicts a document in which the red lines denote the positions of a query entity and blue lines denote the positions of a related entity along the length of the document, again. We take $Dist(e, e')$ as the average of the sum of the absolute difference in position from a query entity e to each related entity e', that is, the average of the sum of the absolute difference in position from a red line to each blue line. Subsequently, we obtain $avgDist(e, e')$ by computing the average of all $Dist(e, e')$.

Note that the length of the document is irrelevant for calculating $avgDist(e, e')$ in both Sections 4.3.2 and 4.3.3. It does not make a difference if we do not know the end of the document d as depicted in Figure 4.1. We now proceed to define the *proximity* score of a related entity e w.r.t. a query entity e' in a document d as the inverse of the average distance between the related entity e and a query entity e', in 4.18 for both the approaches described in Sections 4.3.2 and 4.3.3 as:

$$proximityScore(e, e') = \frac{1}{avgDist(e, e')} \tag{4.18}$$

The *proximity* score of a related entity e for a document d for `AND` and `OR` semantics is

defined as follows:

$$proximityScore_{\wedge}(e,d) = \frac{\sum_{e' \in E_Q} proximityScore(e,e')}{|E_Q|} \tag{4.19}$$

$$proximityScore_{\vee}(e,d) = \sum_{e' \in E_Q \cap ents(d)} \frac{proximityScore(e,e')}{|E_Q|} \cdot \frac{|ents(d) \cap E_Q|}{|E_Q|} \tag{4.20}$$

Finally, we incorporate 4.19 and 4.20 into our relatedness score for an entity $e \in E_D \setminus E_Q$ in a document d for `AND` and `OR` semantics as below.

We consider the following *relatedness* score for an entity $e \in E_D \setminus E_Q$ in a document d for the case of `AND` semantics:

$$\begin{aligned} score^{r}_{\wedge}(e,d) &= idf_{\wedge}(e) \cdot proximityScore_{\wedge}(e,d) \cdot \sum_{t \in T_Q} (score^{t}_{\wedge}(t) \cdot \frac{|docs(t) \cap D_Q \cap docs(e)|}{|docs(t) \cap D_Q|}) \\ &= idf_{\wedge}(e,d) \cdot proximityScore_{\wedge}(e,d) \cdot \sum_{t \in T_Q} (\frac{|docs(t) \cap D_Q \cap docs(e)|}{|D_Q|}) \end{aligned} \tag{4.21}$$

where $idf_{\wedge}(e)$ is the inverse document frequency of entity e in the set of documents discussing about the query entities in the entire corpus, which can be defined in the same way as in 4.13 in Section 4.3.1:

$$idf_{\wedge}(e) = 1 - \frac{|docs(e) \cap (\cap_{e' \in E_Q} docs(e'))|}{|\cap_{e' \in E_Q} docs(e')|} \tag{4.22}$$

The formula considers the percentage of documents in which the entity co-occurs with the query entities in important time periods.

For the case of `OR` semantics, the above formula does not consider the number of different query entities discussed in documents together with the entity e. To also handle this aspect, we consider the following *relatedness* score for the case of `OR` semantics:

$$score^r_{\vee}(e,d) = idf_{\vee}(e)\ N(E_Q,e) \cdot proximityScore_{\vee}(e,d) \cdot \sum_{t \in T_Q} (score^t_{\vee}(t) \cdot \frac{|docs(t) \cap D_Q \cap docs(e)|}{|docs(t) \cap D_Q|})$$
$$= idf_{\vee}(e,d)\ N(E_Q,e) \cdot proximityScore_{\vee}(e,d) \cdot \sum_{t \in T_Q} (N(E_Q,t) \cdot \frac{|docs(t) \cap D_Q \cap docs(e)|}{|D_Q|}) \quad (4.23)$$

where $N(E_Q, e)$ is the average percentage of query entities discussed in articles together with entity e, i.e.:

$$N(E_Q,e) = \frac{\sum_{d \in docs(e) \cap D_Q} \frac{|ents(d) \cap E_Q|}{|E_Q|}}{|docs(e) \cap D_Q|} \quad (4.24)$$

Now the inverse document frequency $idf_{\vee}(e)$, similar to 4.16 in Section 4.3.1 includes documents mentioning at least one of the query entities, i.e.:

$$idf_{\vee}(e) = 1 - \frac{|docs(e) \cap (\cup_{e' \in E_Q} docs(e'))|}{|\cup_{e' \in E_Q} docs(e')|} \quad (4.25)$$

The above formula favors related entities that: i) co-occur frequently with many of the query entities, ii) are discussed in documents published in important (for the query entities) time periods, and iii) are at a close distance to the query entities inside the documents.

Now, the probability of a document $d \in D_Q$ given only other entities mentioned in the retrieved documents (E_{D_Q}) for both the approaches described in Sections 4.3.2 and 4.3.2 can be defined as:

$$P(d|E_{D_Q}) = \frac{\sum_{e \in ents(d) \setminus E_Q} score^r(e,d)}{\sum_{d' \in D_Q} \sum_{e' \in ents(d') \setminus E_Q} score^r(e',d)} \quad (4.26)$$

4.4 Joining the Models

We can now combine the different models in a single probability score:

$$P(d|E_Q, t_d, E_{D_Q}) = \frac{P(d|E_Q)P(d|T_Q)P(d|E_{D_Q})}{\sum_{d' \in D_Q} P(d'|E_Q)P(d'|T_Q)P(d'|E_{D_Q})} \quad (4.27)$$

where the denominator can be ignored as it does not influence the ranking. Note that for combining the different models we can just perform a simple multiplication of the *relativeness*, *timeliness* and *relatedness* models since these models are independent of each other. A similar justification holds for the combination of any two of the *relativeness*, *timeliness* and *relatedness* models at a time.

Chapter 5

Stochastic Modeling

In this chapter, we model the problem as a *random walker* on the graph (*Markov chain*) defined by the query-entities E_Q, the returned documents D_Q, and the entities mentioned in the documents E_{D_Q}. Then, we propose a biased (personalized-like) PageRank algorithm for analyzing the graph and scoring its nodes.

5.1 Transition Graph

The walker starts from a query-entity and can move either to a document mentioning the entity or to a related entity (co-occurring with the query-entity in at least one document). From a document, the walker can move to an entity mentioned in it, while from a no query-entity, the walker can only move to a document mentioning that entity.

In the case of logical `AND` semantics, the query-entities should be connected with all documents, while in the case of `OR` semantics, each query-entity should be connected with at least one document. Figure 5.1 shows an example of a transition graph for the case of `OR` semantics. In this example, the query-entities are two (the black nodes), while we notice that three of the documents mention both query-entities (d_2, d_3 and d_4).

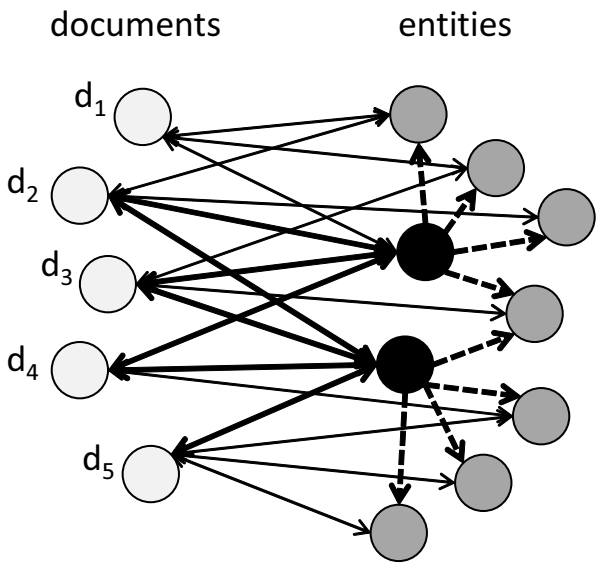

Figure 5.1: An example of the considered transition graph for the case of logical `OR` semantics (with the query-entities being the black nodes).

5.2 Transition Probabilities

From query-entities to documents or related entities

When the walker lies at a query-entity $e \in E_Q$, he can either move to a document d mentioning the entity or to a related entity e'. When moving to a document, we consider its *relativeness* and *timeliness* scores as introduced in the previous section. Specifically, the weight of the edge from a query-entity e to a document d is defined as:

$$weight(e \to d) = \frac{score^f(d, E_Q) \cdot score^t(t_d)}{\sum_{d' \in docs(e) \cap D_Q} (score^f(d', E_Q) \cdot score^t(t_{d'}))} \tag{5.1}$$

For moving from a query-entity e to a related entity e', we consider the *relativeness* score of e':

$$weight(e \to e') = \frac{score^r(e')}{\sum_{e'' \in out(e)} (score^r(e''))} \tag{5.2}$$

where $out(e)$ is the set of nodes connected to e through outgoing edges starting from e.

Based on the above, the weight of the edge from a query-entity e to a connected node n (that can be either a document or a related entity) is defined as:

$$weight(e \to n) = \begin{cases} p_1 \cdot weight(e \to d) & n \in D_Q \\ (1 - p_1) \cdot weight(e \to e') & n \in E_{D_Q} \end{cases} \tag{5.3}$$

where $p_1 \in [0, 1]$ is the probability that the walker selects to move to a document node.

From documents to entities

For moving from a document d to an entity e mentioned in d, it is more likely that the walker moves to an entity that is mentioned many times within its contents. Thus, we simply consider the normalized frequency of e in d. Specifically:

$$weight(d \to e) = \frac{count(e, d)}{\sum_{e' \in ents(d)}(count(e', d))} \tag{5.4}$$

From no query-entities to documents

Likewise, when moving from a no query-entity e to a document d, it is more likely that the walker moves to a document that mentions e many times. Thus, we again consider the normalized frequency of e in d. Specifically:

$$weight(e \to d) = \frac{count(e, d)}{\sum_{d' \in docs(e) \cap D_Q}(count(e, d')} \tag{5.5}$$

5.3 Stochastic Analysis (Random Walk with Restart)

For analyzing the transition graph, we follow a PageRank-like algorithm. The walker starts from the query-entities and can either follow an edge or perform a "restart", i.e., jump to a query-entity and start again the traversal . Formally, the score of a graph node n is defined as:

$$r(n) = d \cdot Jump(n) + (1 - d) \cdot \sum_{n' \in in(n)} (weight(n' \to n) \cdot r(n')) \tag{5.6}$$

where $d \in [0, 1]$ is the probability that the walker performs a restart, $Jump(n)$ is the probability the walker to restart by jumping to node n, $in(n)$ is the set of nodes connected to n through incoming edges, and $weight(n \to n')$ (as defined in Formulas 5.1-5.5) is the probability that the walker visits n when being at node n' connected to n (there should be an edge from n' to n).

As regards the *restart*, we allow equiprobable jumps only to query entities, i.e.:

$$Jump(n) = \begin{cases} 1/|E_Q| & n \in E_Q \\ 0 & n \notin E_Q \end{cases} \tag{5.7}$$

Regarding the initial scores of the graph nodes, we assign the same score to the query entities ($1/|E_Q|$), while the score of all other entities is zero. Finally, the algorithm should iteratively run to convergence.

Chapter 6

Evaluation

We evaluated the performance of the proposed models by exploiting the ground truth dataset described in Section 2.3 and using two evaluation measures: Normalized Discounted Cumulative gain (NDCG) at positions 5, 10, full list, and precision at 5 and 10 (precision for full list is the same in all cases). The complete NDCG and precision tables are publicly available[1]. For precision, we consider a document as relevant if its relevance score is either 2 or 3 and irrelevant if it is either 0 or 1. As regards *timeliness*, we considered *day* granularity. We also tested the case of random rankings of the results. In this case, we computed 10 different random lists for each query and considered the average NDCG and precision scores.

6.1 Effectiveness of Probabilistic Model

We first tested the performance of the different approaches of the *relativeness* and the *relatedness* models described in Chapter 4 to see which of the approaches works best for each model. Based on that, we selected the best performing model for *relativeness* and *relatedness* and checked the performance of the *timeliness* model as well the combination of these models.

[1] Available at: `https://www.dropbox.com/s/uxl14kod8csum4h/Evaluation_Tables.zip?dl=0`

6.1.1 Relativeness and Relatedness

In Chapter 4, we had modelled *relativeness* using three approaches: entity frequency based which did not take into consideration the entity position, exponential decay with entity position and linear decrease with entity position. Similarly, *relatedness* had been modelled using three approaches: entity frequency based without entity position consideration, average distance between entities and closest distance between entities. For the second approach of *relativeness* model, namely the exponential decay with entity position based approach, we chose rate factor values as 10^{-3}, 5×10^{-4} and 10^{-5}. The smaller we keep the rate factor, the slower will be the exponential decay. The description of the different *relativeness* and *relatedness* models is as follows:

i) $[A_1]$: Relativeness Score based on entity frequency (without entity position)

ii) $[A_2]$: Relativeness Score with exponential decay keeping rate factor a as 10^{-3}

iii) $[A_3]$: Relativeness Score with exponential decay keeping rate factor a as 5×10^{-4}

iv) $[A_4]$: Relativeness Score with exponential decay keeping rate factor a as 10^{-5}

v) $[A_5]$: Relativeness Score considering linear decrease with position

vi) $[C_1]$: Relatedness Score based on entity frequency (without entity position)

vii) $[C_2]$: Relatedness Score considering closest distance between entities

viii) $[C_3]$: Relatedness Score considering average distance between entities

Table 6.1 shows the average NDCG and precision scores of the *relativeness* and *relatedness* models for all queries. We have also shown the average NDCG and precision scores of the *relativeness* and *relatedness* models for each query type in Tables 6.2, 6.3, 6.4 and 6.5.

Table 6.1: Average NDCG and Precision of the relativeness and relatedness models for all queries (Q1-Q24).

Measure	$[A_1]$	$[A_2]$	$[A_3]$	$[A_4]$	$[A_5]$	$[C_1]$	$[C_2]$	$[C_3]$
NDCG@5	0.48	**0.50**	0.49	0.46	0.48	**0.42**	0.28	0.18
NDCG@10	0.52	**0.56**	0.53	0.52	0.53	**0.50**	0.35	0.23
NDCG@all	0.79	**0.80**	**0.80**	0.79	0.79	**0.77**	0.70	0.64
P@5	0.44	**0.52**	0.51	0.48	**0.52**	**0.47**	0.29	0.20
P@10	0.38	**0.43**	0.40	0.41	0.40	**0.44**	0.29	0.22

Table 6.2: Average NDCG and Precision of the relativeness and relatedness models for single-entity queries (Q1-Q6).

Measure	$[A_1]$	$[A_2]$	$[A_3]$	$[A_4]$	$[A_5]$	$[C_1]$	$[C_2]$	$[C_3]$
NDCG@5	0.66	**0.69**	0.68	0.69	0.67	**0.40**	0.16	0.04
NDCG@10	0.69	**0.76**	0.73	0.71	0.74	**0.51**	0.25	0.08
NDCG@all	0.88	**0.90**	0.89	0.88	0.89	**0.75**	0.64	0.55
P@5	0.57	**0.60**	**0.60**	**0.60**	**0.60**	**0.50**	0.17	0.03
P@10	0.40	**0.47**	0.45	0.42	0.43	**0.45**	0.20	0.8

Table 6.3: Average NDCG and Precision of the relativeness and relatedness models for multiple-entity AND queries (Q7-Q12).

Measure	$[A_1]$	$[A_2]$	$[A_3]$	$[A_4]$	$[A_5]$	$[C_1]$	$[C_2]$	$[C_3]$
NDCG@5	0.34	**0.41**	0.36	0.36	0.39	**0.31**	0.19	0.15
NDCG@10	0.43	0.42	0.41	**0.46**	0.43	**0.40**	0.24	0.20
NDCG@all	**0.76**	**0.76**	0.75	**0.76**	**0.76**	**0.75**	0.67	0.66
P@5	0.43	0.53	0.50	0.47	**0.57**	**0.30**	0.17	0.13
P@10	0.50	0.47	0.43	**0.52**	0.45	**0.42**	0.23	0.22

Table 6.4: Average NDCG and Precision of the relativeness and relatedness models for multiple-entity OR queries (Q13-Q18).

Measure	[A1]	[A2]	[A3]	[A4]	[A5]	[C1]	[C2]	[C3]
NDCG@5	**0.68**	0.61	0.63	0.59	0.60	**0.50**	0.42	0.25
NDCG@10	**0.69**	0.63	0.61	0.64	0.62	**0.61**	0.51	0.30
NDCG@all	**0.87**	0.83	0.84	0.84	0.83	**0.81**	0.76	0.67
P@5	0.60	0.60	**0.63**	0.57	0.60	**0.57**	0.47	0.30
P@10	0.42	0.40	0.38	**0.43**	0.42	**0.50**	0.42	0.28

Table 6.5: Average NDCG and Precision of the relativeness and relatedness models for category queries (Q19-Q24).

Measure	[A1]	[A2]	[A3]	[A4]	[A5]	[C1]	[C2]	[C3]
NDCG@5	0.22	**0.31**	0.28	0.20	0.26	**0.46**	0.33	0.26
NDCG@10	0.28	**0.42**	0.37	0.29	0.35	**0.48**	0.39	0.34
NDCG@all	0.66	**0.71**	0.70	0.66	0.69	**0.77**	0.71	0.68
P@5	0.17	**0.33**	0.30	0.27	0.30	**0.50**	0.37	0.33
P@10	0.20	**0.37**	0.33	0.27	0.32	**0.40**	0.32	0.30

Score Description for Relativeness

For *relativeness*, it is noticed that when we consider all the queries, relativeness with exponential decay and relativeness with linear decrease seem to perform better than relativeness which is just entity frequency based. An exception here can be noticed at NDCG@5 where the score of relativeness with exponential decay keeping rate factor as 10^{-5} is lesser than relativeness which is just entity frequency based. For single-entity queries, relativeness with exponential decay and relativeness with linear decrease always perform better than relativeness which is just entity frequency based. The same appears to be true for the case of category queries, with an exception at NDCG@10 where the score of relativeness with exponential decay keeping rate factor as 5×10^{-4} is lesser than plain entity frequency based relativeness. The performance in case of multiple-entity AND queries is

almost the same if not better for relativeness with exponential decay and relativeness with linear decrease over plain entity frequency based relativeness. It is interesting that for multiple-entity OR queries, the NDCG scores for plain entity frequency based relativeness are better than relativeness with the other two approaches but the precision scores are lower than the best scores.

In general, we conclude that the position of the entities in the document does play a role in deciding the importance of a document as usually when a document has an entity as the main topic, it is also the case that the entity occurs more at the beginning of the document rather than at the end. Hence, it makes sense not to assign equal importance to two same entities at different positions. The best performance for all queries is observed with the relativeness model with exponential decay and keeping rate factor 10^{-3}. Since this happens at the biggest value among all rate factors, it can be inferred that the user attention decreases rapidly as he moves across the document.

Score Description for Relatedness

Regarding *relatedness*, we observe that relatedness which is just entity frequency based consistently outperforms relatedness which considers closest distance between entities and relatedness which considers average distance between entities. Similarly, relatedness with closest distance between entities consistently outperforms relatedness with average distance between entities. This holds true for all the queries as well as each query type.

One of the problems that we identified while modeling *relatedness* considering position was disambiguation errors and multiple-entity detection at the same position by the entity-linking system. This caused problem especially when there was just one query entity occurrence inside the document and that had been multiply disambiguated. As an example, consider the Figure 6.1 which shows a part of a NYT article retrieved for the query entity *President of Colombia*. In this article, the word "President" in the highlighted text gets mapped by the entity linking system to the entity *President of Colombia* when it considers the name of the president immediately following it. At the same time the word "President" also gets mapped to the entity *President of the United States* when it just considers this single word due to the high popularity of the word "President" to be associated with the US President. In such a case, assuming that there is no other occurrence of the entity *President of the United States* inside the document, the average distance between the query entity *President of Colombia* and the related entity *President*

of the United States becomes zero, thereby making the proximity score as infinity. This problem was resolved by not considering other occurrences at the same position as with the query entity.

Colombia Turns Drug War Into a Long Chase

BOGOTA, Colombia, Nov. 4– The Colombian Government's war on drugs has essentially turned into a chase for Pablo Escobar Gaviria and Jose Gonzalo Rodriguez Gacha, regarded as the two most powerful and violent leaders of South America's illegal drug operations.

These two bosses of the Medellin drug ring are said to have become an obsession with President Virgilio Barco Vargas, who believes that they have not fled Colombia, are hiding out under the protection of a network of informants and that they are orchestrating the violence that has traumatized the country in recent weeks.

Figure 6.1: A part of a sample NYT article retrieved for the query entity *Colombian President*

Now consider the Figure 6.2 as an example of an article returned for the query entity *Shaquille O'Neal*. In this article, suppose that the highlighted word "Los Angeles" gets mapped to the entity *Los Angeles* and the highlighted word "Los Angeles Lakers" gets mapped multiply to both the entities, the city of *Los Angeles* and the basketball team *Los Angeles Lakers*, by the entity-linking system as it considers different combination of words while detecting entities. In such a case, considering just these two detections for the entity *Los Angeles*, the detection of the entity *Los Angeles* for the word "Los Angeles Lakers" will result in incorrect values of distance when calculating distance between the query entity *Shaquille O'Neal* and the related entity *Los Angeles* for both the closest distance and the average distance between entities relatedness approach. Here the detection of the entity *Los Angeles* for the word "Los Angeles Lakers" does not make much sense. This is a possible reason why our approaches considering position do not work so well and this is an underlying problem of the entity-linking system which is not easily resolved.

6.1.2 Overall Results combining the Probabilistic Models

We evaluated our combined probabilistic model considering the approaches for *relativeness* and *relatedness* which gave the best results, i.e., relativeness with exponential decay keeping rate factor as 10^{-3} and relatedness which is just plain entity frequency based. Timeliness

Why a $120 Million Contract Makes Sense for Lakers

Los Angeles, CA, Aug. 23– To be successful in professional basketball these days, you have to master the fundamentals: shooting, passing and marketing. Mostly marketing.

The Los Angeles Lakers of the National Basketball Association stunned the sports world recently with the richest contract ever signed: a 7-year, $120 million deal with a 24-year-old center, Shaquille O'Neal. A basketball franchise is, at bottom, a business, so how can the team afford to pay a single player – even a 7 foot 1 inch, 320-pound star – that much and still make ends meet?

Figure 6.2: A part of a sample NYT article retrieved for the query entity *Shaquille O'Neal.*

in the Table 6.6 as well as the subsequent tables is denoted by [B]. We use plain entity frequency based relativeness as a baseline model for our problem since it considers entity frequency which is a classic numerical statistic [36]. Table 6.6 shows the average NDCG and precision scores for all queries. We notice that joining all three models provides the best results. The improvement of the top-5 results compared to plain entity frequency based relativeness is about 23% in NDCG and 36% in precision, which is statistically significant (paired t-test, $p \leq 0.05$). Further, the improvement of the top-10 results compared to plain entity frequency based relativeness is about 17% in NDCG and 12% in precision, of which the improvement in NDCG is statistically significant (paired t-test, $p \leq 0.05$). The results also show that a combination of *relativeness* and *relatedness* performs very well, outperforming the joined models for P@10. This means that considering other entities that co-occur frequently with the query entities in important time periods as well as the importance of entities based on position has a positive effect on the ranking.

Table 6.6: Average NDCG and Precision of the probabilistic models for all queries (Q1-Q24).

Measure	$[A_2]$	[B]	$[C_1]$	$[A_2][B]$	$[A_2][C_1]$	$[B][C_1]$	$[A_2][B][C_1]$
NDCG@5	0.50	0.27	0.42	0.54	0.55	0.44	**0.59**
NDCG@10	0.56	0.36	0.50	0.58	**0.61**	0.52	**0.61**
NDCG@all	0.80	0.69	0.77	0.81	**0.82**	0.76	**0.82**
P@5	0.52	0.28	0.47	0.55	0.56	0.45	**0.60**
P@10	0.43	0.30	0.44	0.42	**0.46**	0.41	0.43

6.1.3 Detailed results per query type

Tables 6.7-6.10 show the results per query type.

Table 6.7: Average NDCG and Precision of the probabilistic models for single-entity queries (Q1-Q6).

Measure	[A_2]	[B]	[C_1]	[A_2][B]	[A_2][C_1]	[B][C_1]	[A_2][B][C_1]
NDCG@5	0.69	0.30	0.40	**0.78**	0.72	0.45	0.76
NDCG@10	0.76	0.38	0.51	**0.81**	**0.81**	0.52	0.77
NDCG@all	**0.90**	0.67	0.75	**0.90**	**0.90**	0.72	**0.90**
P@5	0.60	0.23	0.50	**0.70**	0.63	0.40	0.67
P@10	0.47	0.27	0.45	0.48	**0.50**	0.38	0.47

Table 6.8: Average NDCG and Precision of the probabilistic models for multiple-entity AND queries (Q7-Q12).

Measure	[A_2]	[B]	[C_1]	[A_2][B]	[A_2][C_1]	[B][C_1]	[A_2][B][C_1]
NDCG@5	0.41	0.28	0.31	0.46	0.43	0.38	**0.49**
NDCG@10	0.42	0.33	0.40	0.47	0.45	0.46	**0.49**
NDCG@all	0.76	0.72	0.75	0.78	0.77	0.77	**0.79**
P@5	0.53	0.33	0.30	0.53	0.53	0.43	**0.57**
P@10	0.47	0.32	0.42	0.43	**0.48**	0.47	0.45

Table 6.9: Average NDCG and Precision of the probabilistic models for multiple-entity OR queries (Q13-Q18).

Measure	[A_2]	[B]	[C_1]	[A_2][B]	[A_2][C_1]	[B][C_1]	[A_2][B][C_1]
NDCG@5	0.61	0.24	0.50	0.59	**0.63**	0.46	0.61
NDCG@10	0.63	0.36	0.61	0.62	**0.66**	0.54	0.64
NDCG@all	0.83	0.69	0.81	0.82	**0.84**	0.78	**0.84**
P@5	**0.60**	0.27	0.57	**0.60**	**0.60**	0.47	**0.60**
P@10	0.40	0.30	**0.50**	0.40	0.44	0.42	0.42

Table 6.10: Average NDCG and Precision of the probabilistic models for category queries (Q19-Q24).

Measure	[A_2]	[B]	[C_1]	[A_2][B]	[A_2][C_1]	[B][C_1]	[A_2][B][C_1]
NDCG@5	0.31	0.26	0.46	0.32	0.42	**0.48**	**0.48**
NDCG@10	0.42	0.36	0.48	0.43	0.52	**0.54**	0.53
NDCG@all	0.71	0.69	**0.77**	0.72	0.76	**0.77**	**0.77**
P@5	0.33	0.27	0.50	0.37	0.47	0.50	**0.57**
P@10	0.37	0.30	0.40	0.35	**0.43**	0.38	0.40

Regarding *single entity queries* (Table 6.7), we observe that combining *relativeness* with *timeliness* and combining *relativeness* with *relatedness* has better performance with the case of joining all models. This means that a combination of all the three models seems to have a minor negative effect on the rankings. In this query type, query 4 has the best performance on the joined model with NDCG@5 = 1.0 and NDCG@10 = 0.97. On the contrary, query 3 in the combined model has the worst performance in this category with NDCG@5 = 0.57 and NDCG@10 = 0.57 (this query returns several articles which contain just a mention of Beatles, but are totally irrelevant to the rock band, which may confuse our modeling).

As regards *multiple-entity AND queries* (Table 6.8), we observe that joining all models seems to perform better than all other models. The only exception here occurs at P@10 where a combination of *relativeness* and *relatedness* gives a better performance than the joined model. We also notice that the performance of the joined models in this query type is less when compared to the single-entity case. In this query type, the NDCG@5 score of all queries ranges from 0.33 (for query 10) to 0.60 (for query 9). Similarly, the NDCG@10 score of all queries ranges from 0.36 (for query 10) to 0.57 (for query 11). Query 10 seems to have a low performance because it contains several articles mentioning films or containing a name mention of Robert De Niro and Sean Connery without being important for them. A lot of such articles are about all the upcoming films or films released containing names of other big Hollywood stars as well which in our case negatively impacts the ranking.

Regarding *multiple-entity OR queries*, (Table 6.9) the combination of *relativeness* and *relatedness* seems to perform best (except at P@10 where *timeliness* has the best performance). In this query type, query 16 has the best performance on the joined model with NDCG@5 = 0.95 and NDCG@10 = 0.93. On the contrary, query 14 in the combined

model has the worst performance in this category with NDCG@5 = 0.24 and NDCG@10 = 0.23 (this query returns several articles which contain an incorrect mapping to Pablo Escobar).

As regards *category queries* (Table 6.10), the performance of the joined models as well as the *relatedness* and *timeliness* combination is very well and almost the same. This means that *relativeness* in this case could be having a minor negative impact on our ranking. For this query type, query 20 has the best performance with NDCG@5 = 1.0 and NDCG@10 = 0.92 and query 24 has the worst performance with NDCG@5 = 0.12 and NDCG@10 = 0.22. By examining the results of query 24, we see that this query returns several articles mentioning Alzheimer's disease but are actually not about the causes, symptoms or treatment of the disease. Rather these articles seem to be more about persons and just mention them having the disease which is not very useful. Query 23 has the next worst performance with NDCG@5 = 0.31 and NDCG@10 = 0.32. Examination of these articles revealed that several unimportant articles also got returned about the crash of Avianca airlines 52 along with important articles. The unimportant articles contain history of Avianca airlines, history of air crashes, TV broadcast about the crash, as well as there are several articles which are a list of articles but these articles fail to provide actual details of the crash. Although the occurrence of such unimportant articles along with important articles related to the crash may boost timeliness due to increase of articles on a particular day, still they fail to improve the performance of the joined model.

6.2 Effectiveness of Stochastic Model

We run experiments for different values of d probability (probability to perform a restart, cf. Equation 5.6) and different values of p_1 probability (probability to move to a document node when being at a query-entity node, cf. Equation 5.3). For the restart probability, we tested the following 5 cases: $d = 0.0, 0.2, 0.4, 0.6, 0.8$. Note that testing $d = 1.0$ does not make sense, since the walker will never reach document nodes. Regarding the p_1 probability, we tested 6 cases ($p_1 = 0.0, 0.2, 0.4, 0.6, 0.8, 1.0$). Empirically, $d = 0.2$ provides the best results independently of the p_1 value. Thus, all the results reported below correspond to $d = 0.2$. Moreover, in all experiments, we set the number of iterations to 30.

Tables 6.11-6.14 show the results per query type and for different values of p_1 probability. For the first three types of queries, we observe that $p_1 = 1.0$ provides the best results and that, as the probability gets lower, the results get worse. Using $p_1 = 1.0$, the walker

can move only to document nodes. This means that for small values of p_1 probability, the algorithm overemphasizes the association between the entities resulting in worst performance. Thus, the results show that reaching documents through related entities affects negatively the rankings for these types of queries.

Compared to results of the probabilistic modeling, we see that: i) the performance of the probabilistic model is much better than the stochastic model for single-entity queries, while the stochastic model outperforms the best probabilistic model for multiple-entity `AND` queries, and ii) the best probabilistic model outperforms the stochastic model for multiple-entity `OR` queries. Regarding (ii), this failure of the stochastic model is probably due to the fact that the graph in the case of `OR` queries is not as well-connected as in the case of `AND` queries (for `OR` queries, the graph may contain disconnected components, e.g., in cases where there are no documents mentioning all query entities).

Table 6.11: Average NDCG and Precision of the stochastic model for single entity queries (Q1-Q6).

Measure	p_1=0.0	p_1=0.2	p_1=0.4	p_1=0.6	p_1=0.8	p_1=1.0
NDCG@5	0.09	0.27	0.48	0.63	0.65	**0.67**
NDCG@10	0.18	0.35	0.54	0.69	0.72	**0.74**
NDCG@all	0.60	0.68	0.77	0.85	**0.87**	**0.87**
P@5	0.07	0.23	0.43	0.47	0.53	**0.60**
P@10	0.12	0.23	0.33	0.37	0.42	**0.43**

Table 6.12: Average NDCG and Precision of the stochastic model for multiple-entity AND queries (Q7-Q12).

Measure	p_1=0.0	p_1=0.2	p_1=0.4	p_1=0.6	p_1=0.8	p_1=1.0
NDCG@5	0.29	0.29	0.32	0.31	0.37	**0.48**
NDCG@10	0.36	0.36	0.40	0.45	0.47	**0.52**
NDCG@all	0.72	0.73	0.74	0.75	0.78	**0.80**
P@5	0.27	0.27	0.27	0.30	0.37	**0.57**
P@10	0.30	0.30	0.38	0.45	0.47	**0.50**

Table 6.13: Average NDCG and Precision of the stochastic model for multiple-entity OR queries (Q13-Q18).

Measure	p_1=0.0	p_1=0.2	p_1=0.4	p_1=0.6	p_1=0.8	p_1=1.0
NDCG@5	0.20	0.30	0.32	0.38	0.45	**0.59**
NDCG@10	0.27	0.38	0.42	0.51	0.52	**0.65**
NDCG@all	0.66	0.71	0.72	0.75	0.76	**0.84**
P@5	0.27	0.33	0.33	0.37	0.47	**0.53**
P@10	0.30	0.33	0.35	0.42	0.42	**0.45**

Table 6.14: Average NDCG and Precision of the stochastic model for category queries (Q19-Q24).

Measure	p_1=0.0	p_1=0.2	p_1=0.4	p_1=0.6	p_1=0.8	p_1=1.0
NDCG@5	0.43	0.47	**0.53**	0.50	0.46	0.23
NDCG@10	0.48	0.54	**0.55**	0.52	0.54	0.36
NDCG@all	0.74	0.77	**0.79**	0.77	0.77	0.68
P@5	0.57	0.57	**0.60**	**0.60**	**0.60**	0.23
P@10	0.42	**0.43**	0.40	0.40	**0.43**	0.28

However, for the category queries (Table 6.14), we see that $p_1 = 1.0$ provides the worst results, while $p_1 = 0.4$ performs better. This means that, for this type of queries, considering the associations of the query entities with other entities mentioned in the retrieved documents affects positively the results. However, at the same time, we see that we should not give too much emphasis to this (by giving very low value to p_1). Notice that this is in correspondence to the evaluation results of the probabilistic models for the same type of queries (cf. Section 6.1). Moreover, we notice that the stochastic model for $p_1 = 0.4$ performs better than the best probabilistic model, providing better rankings and more relevant results in the top positions.

6.3 Synopsis of Evaluation Results

Based on the evaluation results, we can conclude that:

- Considering position of entities inside a document has a positive effect on the *relativeness* score. The same, however, cannot be said for *relatedness* as disambiguation errors or multiple detection of entities at the same position by the entity linking system limit the performance of the *relatedness* score considering position of entities.
- *Category queries* is a special case of logical `OR` semantics, where the number of query entities can be very large and this makes the results more susceptible to disambiguation errors of the used entity linking system. In all the other three types of queries, the number of query entities is less (in our experiments limited to a maximum of three query entities). Thereby, for this type of queries one should select a model which considers the associations between the query-entities and other entities mentioned in the returned documents, since this limits the negative impact of this problem. Finally, a stochastic model with $p_1 = 0.4$ outperforms the best probabilistic model (relatedness).
- For *single entity queries*, a model which considers both relativeness with exponential decay and timeliness should be selected, as relatedness seems to have a bit of a negative affect on the rankings in this case. For *multiple-entity AND queries*, one may select the stochastic model which seems to perform better than the probabilistic models, however for *multiple-entity OR queries* where the graph is not so well connected, one may opt for a probabilistic model.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

We have formalized the problem of ranking archived documents for structured (SPARQL) queries on semantic layers, i.e., on RDF graphs describing metadata and annotation information about the documents. For coping with this problem, we define three aspects which make an archived document important and those are: i) the *relativeness* of the documents to the query entities, ii) the *timeliness* of the documents, and iii) the *relatedness* of other entities mentioned in the documents to the query entities. We then propose several approaches for the aspects of *relativeness* and *relatedness*. Finally, we propose two ranking models (a probabilistic one and a stochastic one) which jointly consider all the three aspects. To evaluate our approach, and due to lack of evaluation datasets for the problem at hand, we carefully created a new ground truth dataset which we make publicly available for fostering further research on similar problems. The results show that considering position of entities inside a document has a positive effect on the *relativeness* score. The same, however, cannot be said for *relatedness* as disambiguation errors or multiple detection of entities at the same position by the entity linking system limit the performance of the *relatedness* score considering position of entities. Further, we observe that the probabilistic model combining all aspects can identify important - for all the query entities - documents, achieving high NDCG and precision scores and outperforming a classic, frequency-based baseline model. The results obtained from the stochastic model for *category queries* make it evident that *relatedness*, i.e., considering the temporal association of the query entities with other entities mentioned in the results, has a high positive impact on the ranking and

can limit the negative effect caused by disambiguation errors of entity linking.

7.2 Future Work

There are several directions of research which could be undertaken in the future. Some of these are listed below.

Ranking on other Web Archives

In this thesis, we provided ranking models for the New York Times(NYT) corpus (a non- versioned news archive). The NYT corpus is small in size and being a well known newspaper, the articles it contains are of high quality without grammatical errors, well structured and can be assumed to be factually correct. A challenging task now would be to study the applicability of similar models on more "classical" web archives (e.g. *Occupy Movement 2011/2012 collection*[1]) which are bigger where much more noisy (and probably spam) data exists, while documents also contain multiple similar or identical versions. It would also be interesting to see whether these models could be applied to Social Media Archives (e.g. a collection of tweets). Unlike NYT articles, tweets are very small in size and sometimes have links mentioned inside them. A tweet for an entity may not even have any other entity mentioned in it. Moreover, tweets could also be spam or noise and one cannot rely on the factual correctness of the tweets.

Ranking with other Models

We could propose and test other ranking models (e.g., Spreading Activation, Learning to Rank) for the NYT corpus or other web archives and check the performance of these models against our current ranking models.

Building User-friendly Interfaces

The end-users of web archives are usually historians, sociologists and journalists. These people are not expected to be proficient with SPARQL. An interesting direction would

[1] `https://archive-it.org/collections/2950`

be to build a user-friendly interactive interface which transparently transforms user interactions to SPARQL queries(e.g., a faceted browsing interface) allowing users to easily and effectively explore digital archives.

Incorporating Time-Aware Entity Linking in Semantic Layers

It is clear that the quality of the entity annotations in the semantic layers plays a crucial role in the performance of the ranking models. Disambiguation errors are a major cause for limiting the performance of our ranking models. A lot of these disambiguation errors get caused because the existing state-of-the-art entity linking systems fail to explicitly consider the time aspect and in particular the temporality of an entity's prior probability (popularity) and embedding (semantic network). As an example, consider the word "*Ronaldo*" inside the text snippet "*Ronaldo scored a goal for Real Madrid*". If this snippet is present in an article from 2002, then it might probably refer to the Brazilian footballer *Ronaldo Luís Nazário de Lima*. However, an entity linking system which is not time-aware and using the latest DBpedia descriptions may link this to the Portuguese soccer player *Cristiano Ronaldo* due to the high popularity of the word to be currently associated with the Portuguese footballer. This problem becomes bigger when annotating, query logs or social media blogs, the texts being short with limited context. Joao[31] presented an entity linking modeling with time-aware entity priors and word embeddings. A future direction could be to generate semantic layers with time-aware entity linking and to check the performance of the ranking models with the newly generated layers.

Appendix A

Data Models used for describing Archived Documents

The *Open Web Archive* data model is an RDF/S data model that we introduced our previous work[19] to describe the semantic information and metadata about the documents of a web archive. The *Open Web Archive* data model[1], is depicted in Figure A.1. We re-use elements from many other existing data models and define 2 new classes and 3 new properties. An archived document is represented using the class `owa:ArchivedDocument`. Further, the archived document may or may not be linked with some versions (i.e., instances of `owa:VersionedDocument`). Versions pages for billions of web sites can be found on the Internet Archive. The New York Times corpus [44] on which we have applied our ranking models does not contain versions.

An archived document with three main elements: i) metadata information, like format(mime type), date of capture/publication, and document title, ii) links to other documents (web pages), archived or not, and iii) set of annotations. Terms from the Dublin Core Metadata Initiative[2] were used to describe some of the metadata. Annotations were described by exploiting the Open Annotation Data Model [43] and the Open Named Entity Extraction (NEE) Model [18].

The Open Annotation Data Model contains an RDF-based framework specification for creating associations (annotations) between related resources, while the Open NEE Model is an extension that allows describing the result of an entity extraction process. An

[1]Specification publicly available at: `http://l3s.de/owa/`

[2]`http://dublincore.org/`

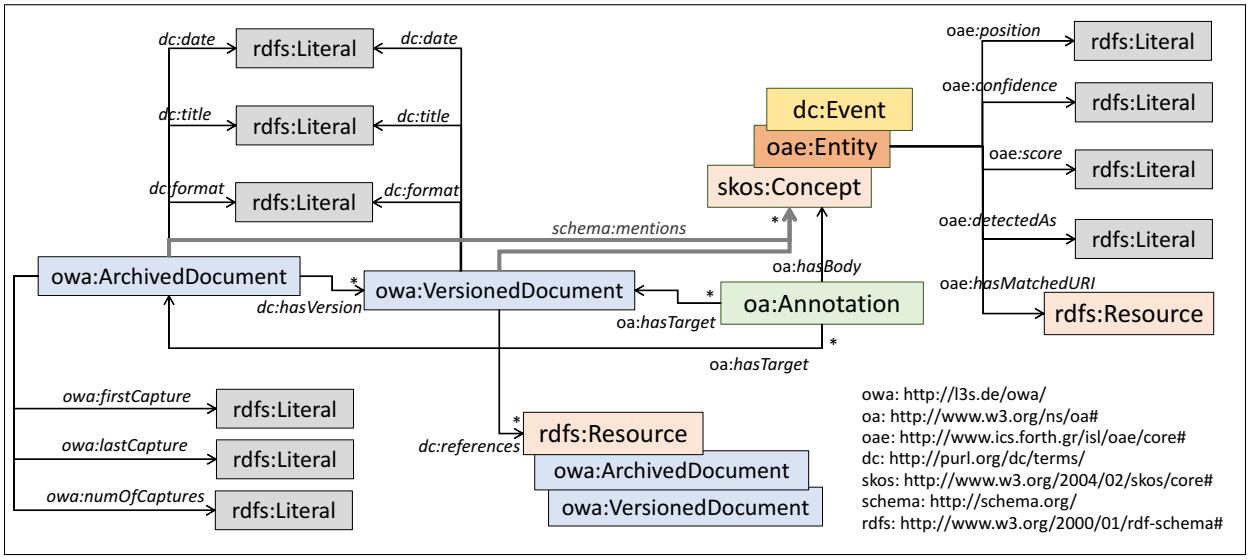

Figure A.1: The *Open Web Archive* data model.

annotation has a *target*, which is an archived document in our case, and a *body* which is a concept, entity or event mentioned in the document. An archived document can be directly related with an entity, concept or event by exploiting the property "`mentions`" of schema.org[3] for reducing the number of derived triples. A concept, entity or event can be associated with information like its name, a confidence score, its position in the document, and a resource (URI). The URI enables to retrieve additional information from the Linked Open Data (LOD) cloud [25] (like properties, relations with other entities, etc.).

[3] `http://schema.org/mentions`

Bibliography

[1] Ablimit Aji, Yu Wang, Eugene Agichtein, and Evgeniy Gabrilovich. "Using the past to score the present: Extending term weighting models through revision history analysis". In: *19th ACM international conference on Information and knowledge management*. ACM. 2010, pp. 629–638.

[2] Harith Alani, Christopher Brewster, and Nigel Shadbolt. "Ranking ontologies with AKTiveRank". In: *International Semantic Web Conference*. Springer. 2006, pp. 1–15.

[3] Kemafor Anyanwu, Angela Maduko, and Amit Sheth. "SemRank: ranking complex relationship search results on the semantic web". In: *Proceedings of the 14th international conference on World Wide Web*. ACM. 2005, pp. 117–127.

[4] Marcelo Arenas, Bernardo Cuenca Grau, Evgeny Kharlamov, Sarunas Marciuska, Dmitriy Zheleznyakov, and Ernesto Jimenez-Ruiz. "SemFacet: semantic faceted search over yago". In: *23rd International Conference on World Wide Web*. ACM. 2014.

[5] Irem Arikan, Srikanta Bedathur, and Klaus Berberich. "Time will tell: Leveraging Temporal Expressions in IR". In: *In WSDM*. ACM. 2009.

[6] Krisztian Balog, Pavel Serdyukov, and Arjen P De Vries. "Overview of the trec 2010 entity track". In: *In TREC 2010*. 2010.

[7] Klaus Berberich, Srikanta Bedathur, Omar Alonso, and Gerhard Weikum. "A Language Modeling Approach for Temporal Information Needs". In: *European Conference on Information Retrieval*. Vol. 10. Springer. 2010, pp. 13–25.

[8] Anila Sahar Butt, Armin Haller, and Lexing Xie. "DWRank: Learning concept ranking for ontology search". In: *Semantic Web* 7.4 (2016), pp. 447–461.

[9] Karen Calhoun. *Exploring digital libraries: foundations, practice, prospects*. Facet Publishing, 2014.

[10] Ricardo Campos, Gaël Dias, Alípio M Jorge, and Adam Jatowt. "Survey of temporal information retrieval and related applications". In: *ACM Computing Surveys (CSUR)* 47.2 (2015), p. 15.

[11] Mario Cataldi, Luigi Di Caro, and Claudio Schifanella. "Emerging topic detection on twitter based on temporal and social terms evaluation". In: *Proceedings of the tenth international workshop on multimedia data mining*. ACM. 2010, p. 4.

[12] Wisam Dakka, Luis Gravano, and Panagiotis Ipeirotis. "Answering general time-sensitive queries". In: *IEEE Transactions on Knowledge and Data Engineering* 24.2 (2012), pp. 220–235.

[13] Lorand Dali, Blaž Fortuna, Thanh Tran Duc, and Dunja Mladenić. "Query-independent learning to rank for rdf entity search". In: *Extended Semantic Web Conference*. Springer. 2012, pp. 484–498.

[14] Renaud Delbru, Nickolai Toupikov, Michele Catasta, Giovanni Tummarello, and Stefan Decker. "Hierarchical link analysis for ranking web data". In: *Extended Semantic Web Conference*. Springer. 2010, pp. 225–239.

[15] Li Ding, Tim Finin, Anupam Joshi, Rong Pan, R Scott Cost, Yun Peng, Pavan Reddivari, VC Doshi, and Joel Sachs. "Swoogle: A semantic web search and metadata engine". In: *Proc. 13th ACM Conf. on Information and Knowledge Management*. Vol. 304. Citeseer. 2004, pp. 10–1145.

[16] Li Ding, Rong Pan, Tim Finin, Anupam Joshi, Yun Peng, and Pranam Kolari. "Finding and ranking knowledge on the semantic web". In: *International Semantic Web Conference*. Springer. 2005, pp. 156–170.

[17] Shady Elbassuoni, Maya Ramanath, Ralf Schenkel, Marcin Sydow, and Gerhard Weikum. "Language-model-based ranking for queries on RDF-graphs". In: *18th ACM conference on Information and knowledge management*. ACM. 2009.

[18] P. Fafalios, M. Baritakis, and Y. Tzitzikas. "Exploiting Linked Data for Open and Configurable Named Entity Extraction". In: *International Journal on Artificial Intelligence Tools* 24.02 (2015).

[19] P. Fafalios, H. Holzmann, V. Kasturia, and W. Nejdl. "Building and Querying Semantic Layers for Web Archives". In: *ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'17)*. Toronto, Ontario, Canada, 2017.

[20] Pavlos Fafalios and Yannis Tzitzikas. "Post-analysis of keyword-based search results using entity mining, linked data, and link analysis at query time". In: *Semantic Computing (ICSC), 2014 IEEE International Conference on*. IEEE. 2014.

[21] Sébastien Ferré. "Sparklis: a sparql endpoint explorer for expressive question answering". In: *ISWC Posters & Demonstrations Track*. 2014.

[22] Alvaro Graves, Sibel Adali, and Jim Hendler. "A method to rank nodes in an RDF graph". In: *Proceedings of the 2007 International Conference on Posters and Demonstrations-Volume 401*. CEUR-WS. org. 2008, pp. 84–85.

[23] Harry Halpin, Daniel M Herzig, Peter Mika, Roi Blanco, Jeffrey Pound, Henry S Thompson, and Thanh Tran Duc. "Evaluating ad-hoc object retrieval". In: *International Workshop on Evaluation of Semantic Technologies, IWEST*. Citeseer. 2010.

[24] Andreas Harth, Sheila Kinsella, and Stefan Decker. "Using naming authority to rank data and ontologies for web search". In: *International Semantic Web Conference*. Springer. 2009, pp. 277–292.

[25] Tom Heath and Christian Bizer. "Linked data: Evolving the web into a global data space". In: *Synthesis lectures on the semantic web: theory and technology* 1.1 (2011), pp. 1–136.

[26] Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. "Robust disambiguation of named entities in text". In: *Conference on Empirical Methods in Natural Language Processing*. 2011.

[27] Aidan Hogan, Stefan Decker, and Andreas Harth. "Reconrank: A scalable ranking method for semantic web data with context". In: *International Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS 2006)*. 2006.

[28] Helge Holzmann and Avishek Anand. "Tempas: Temporal Archive Search Based on Tags". In: *International Conference on World Wide Web*. 2016.

[29] Helge Holzmann, Wolfgang Nejdl, and Avishek Anand. "Exploring Web Archives Through Temporal Anchor Texts". In: *Proceedings of the 2017 ACM on Web Science Conference*. ACM. 2017, pp. 289–298.

[30] Peiquan Jin, Jianlong Lian, Xujian Zhao, and Shouhong Wan. "Tise: A temporal search engine for web contents". In: *Intelligent Information Technology Application, 2008. IITA'08. Second International Symposium on*. Vol. 3. IEEE. 2008, pp. 220–224.

[31] Renato Stoffalette Joao. "Time-Aware Entity Linking". In: *International Semantic Web Conference*. 2017.

[32] Nattiya Kanhabua and Kjetil Nørvåg. "Learning to rank search results for time-sensitive queries". In: *21st ACM international conference on Information and knowledge management*. ACM. 2012, pp. 2463–2466.

[33] Nattiya Kanhabua, Roi Blanco, Kjetil Nørvåg, et al. "Temporal information retrieval". In: *Foundations and Trends in Information Retrieval* 9.2 (2015), pp. 91–208.

[34] Gjergji Kasneci, Fabian M Suchanek, Georgiana Ifrim, Maya Ramanath, and Gerhard Weikum. "Naga: Searching and ranking knowledge". In: *IEEE 24th International Conference on Data Engineering, 2008*. IEEE. 2008, pp. 953–962.

[35] Sara Latifi and Mohammadali Nematbakhsh. "Query-independent learning to rank RDF entity results of SPARQL queries". In: *4th International eConference on Computer and Knowledge Engineering (ICCKE)*. IEEE. 2014.

[36] Jure Leskovec, Anand Rajaraman, and Jeffrey David Ullman. *Mining of massive datasets*. Cambridge University Press, 2014.

[37] Donald Metzler, Rosie Jones, Fuchun Peng, and Ruiqiang Zhang. "Improving search relevance for implicitly temporal queries". In: *32nd international ACM SIGIR conference on Research and development in information retrieval*. ACM. 2009, pp. 700–701.

[38] Roberto Mirizzi, Azzurra Ragone, Tommaso Di Noia, and Eugenio Di Sciascio. "Ranking the linked data: the case of dbpedia". In: *International Conference on Web Engineering*. Springer. 2010, pp. 337–354.

[39] Andrea Moro, Alessandro Raganato, and Roberto Navigli. "Entity linking meets word sense disambiguation: a unified approach". In: *Transactions of the Association for Computational Linguistics* 2 (2014).

[40] Zaiqing Nie, Yuanzhi Zhang, Ji-Rong Wen, and Wei-Ying Ma. "Object-level ranking: bringing order to web objects". In: *Proceedings of the 14th international conference on World Wide Web*. ACM. 2005, pp. 567–574.

[41] Jeffrey Pound, Peter Mika, and Hugo Zaragoza. "Ad-hoc object retrieval in the web of data". In: *19th international conference on World wide web*. ACM. 2010.

[42] Antonio J Roa-Valverde and Miguel-Angel Sicilia. "A survey of approaches for ranking on the web of data". In: *Information Retrieval* 17.4 (2014), pp. 295–325.

[43] Robert Sanderson, Paolo Ciccarese, Herbert Van de Sompel, Shannon Bradshaw, Dan Brickley, Leyla Jael Garc a Castro, Timothy Clark, Timothy Cole, Phil Desenne, Anna Gerber, et al. "Open annotation data model". In: *W3C community draft* (2013).

[44] Evan Sandhaus. "The new york times annotated corpus". In: *Linguistic Data Consortium, Philadelphia* 6.12 (2008).

[45] Wei Shen, Jianyong Wang, and Jiawei Han. "Entity linking with a knowledge base: Issues, techniques, and solutions". In: *IEEE Transactions on Knowledge and Data Engineering* 27.2 (2015), pp. 443–460.

[46] Jaspreet Singh, Wolfgang Nejdl, and Avishek Anand. "Expedition: A Time-Aware Exploratory Search System Designed for Scholars". In: *SIGIR conference on Research and Development in Information Retrieval*. 2016.

[47] Jaspreet Singh, Wolfgang Nejdl, and Avishek Anand. "History by diversity: Helping historians search news archives". In: *ACM Conference on Human Information Interaction and Retrieval*. 2016.

[48] Alberto Tonon, Gianluca Demartini, and Philippe Cudré-Mauroux. "Combining inverted indices and structured search for ad-hoc object retrieval". In: *ACM SIGIR*. ACM. 2012, pp. 125–134.

[49] Yannis Tzitzikas, Nikos Manolis, and Panagiotis Papadakos. "Faceted exploration of RDF/S datasets: a survey". In: *Journal of Intelligent Information Systems* (2016), pp. 1–36.

[50] Wang Wei. "Semantic Search: Bringing Semantic Web Technologies to Information Retrieval". PhD thesis. School of Computer Science, The University of Nottingham, 2009.

[51] Gang Wu, Juanzi Li, Ling Feng, and Kehong Wang. "Identifying potentially important concepts and relations in an ontology". In: *International Semantic Web Conference*. Springer. 2008, pp. 33–49.