2013

Mining Web Dynamics for Search

Na Dai
Lehigh University

Recommended Citation

Dai, Na, "Mining Web Dynamics for Search" (2013). *Theses and Dissertations.* Paper 1466.

MINING WEB DYNAMICS FOR SEARCH

by

Na Dai

A Dissertation

Presented to the Graduate and Research Committee

of Lehigh University

in Candidacy for the Degree of

Doctor of Philosophy

in

Computer Science

Lehigh University

May 2013

This dissertation is accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

(Date)

(Accepted Date)

Brian D. Davison
(Committee Chair)

Daniel Lopresti

Mooi Choo Chuah

Wei-Min Huang
(Department of Mathematics)

Acknowledgments

I am extremely grateful to my parents, Zhongqiu Dai and Zhihua Zhang, for fostering me from the very beginning. My father valued learning from labor and play, and so I was able to repair bicycles, do woodwork, cook for the family, and grow the flowers and fishes before ten-years old. Maybe only because of that, I usually can overcome difficulties by connecting comparable solutions in different domains from an early age. My parents also value knowledge. They made great sacrifices and gave me huge spiritual supports to let me finish my Ph.D. study. Compared with them, I am just like a spoiled child, addicting to my own research every day. I feel ashamed because I did not bring them enough care in return. Maybe only because of that, I feel what I can do now is to contribute my expertise to make the world better, even though only a trivial portion.

I am truthfully grateful to my advisor, Prof. Brian D. Davison, for his guidance, encouragement, and support throughout my Ph.D. study. He is the one who brings me to the research area of information retrieval and web search, and makes me love this area so deeply. He encourages me to conduct independent work and believe myself. But when I get stuck, he always pushed me back to the right direction promptly. I learn a lot from him, among which two critical things include (1) discovering problems and motivations from experimental results; and (2) visioning the future, thinking about what might influence this world. Without his guidance, I could not have finished this dissertation.

I am also grateful to my committee members, professors Daniel Lopresti, Mooi Choo Chuah, and Wei-Min Huang for their excellent guidance and suggestions.

I am very happy to have three collaborative works with Xiaoguang Qi (now at Microsoft) in our lab. While these works are not in the scope of this dissertation, I appreciate him since I learn a lot from him about his research experience especially at my early stage of Ph.D. studies. I appreciate Lan Nie (now at Google) for teaching me her ranking system prototype, based on which my first link analysis work was conducted.

I also appreciate Dr. Dmitry Pechyony and Dr. Rosie Jones for providing me the chance to intern at Akamai Technologies, extending my area to computational advertising.

Many thanks to my labmates Shruti Bhandari, Ovidiu Dan, Vinay Goel, Liangjie Hong, Lan Nie, Xiaoguang Qi, Jian Wang, YaoShuang Wang, Baoning Wu, Zhenzhen Xue, Xiong Xiong, Zaihan Yang, Dawei Yin for their helpful discussions. Many thanks for the helpful comments through numerus paper reviews and interactions with others during conferences.

My work has been financially supported by different sources which include the National Science Foundation under award IIS-0803605 and IIS-0545875. I appreciate Bryan Hodgson, Steve Lidie, Ovidiu Dan, and Liangjie Hong etc. for their great support on computational resources used in my research.

Contents

List of Tables

List of Figures

Abstract

Billions of web users collectively contribute to a dynamic web that preserves how information sources and descriptions change over time. This dynamic process sheds light on the quality of web content, and even indicates the temporal properties of information needs expressed via queries. However, existing commercial search engines typically utilize one crawl of web content (the latest) without considering the complementary information concealed in web dynamics. As a result, the generated rankings may be biased due to the deficiency of knowledge on page or hyperlink evolution, and the time-sensitive facet within search quality, e.g., freshness, has to be neglected. While previous research efforts have been focused on exploring the temporal dimension in retrieval process, few of them showed consistent improvements on large-scale real-world archival web corpus with a broad time span.

We investigate how to utilize the changes of web pages and hyperlinks to improve search quality, in terms of freshness and relevance of search results. Three applications that I have focused on are: (1) document representation, in which the anchortext (short descriptive text associated with hyperlinks) importance is estimated by considering its

historical status; (2) web authority estimation, in which web freshness is quantified and utilized for controlling the authority propagation; and (3) learning to rank, in which freshness and relevance are optimized simultaneously in an adaptive way depending on query type. The contributions of this thesis are: (1) incorporate web dynamics information into critical components within search infrastructure in a principled way; and (2) empirically verify the proposed methods by conducting experiments based on (or depending on) a large-scale real-world archival web corpus, and demonstrated their superiority over existing state-of-the-art.

Chapter 1

Introduction and Outline

1.1 Introduction

Billions of web users collectively contribute to a dynamic web that preserves the traces of web content creators and reflects humans' daily lives [11, 12, 33, 57]. Representative examples include social network sites, microblogs, wikis, video sharing sites, mashups, folksonomies etc. In addition to these Web 2.0 application features, the whole web demonstrates certain dynamic and collaborative evolution patterns. The incoming links pointing to SIGIR 2011 (a conference) home page increases faster than SIGIR 2009 for now (August 2011). Dr. Brian Davison's home page (Lehigh) in 2008 introduced him as an assistant professor, but now that field has been updated to "associate professor". News events, depending on their significance, draw web users' attention in real time—query volumes and the news page incoming links mutually increase. Therefore, the creation, updates and removal of web pages and hyperlinks shed light on the quality of web content, reflect how web users interpret changes in information sources over time, and even indicate the temporal properties of queries. We thus naturally ask how to utilize such complementary information to improve web search.

Web search, more specifically, retrieving web documents in the scope of this thesis, aims to truly satisfy users' information needs expressed through queries.[1] Information seekers usually pay much attention to results at top positions, and the quality of lower rankings becomes less important. To select a small group of documents that mostly satisfy users' information needs is challenging especially given the huge pool of available information on the web. Therefore, web dynamics provides complementary information that helps further differentiate the web pages sharing similar topics (e.g., SIGIR 2011 and SIGIR 2009 home pages), and so enhance the rankings only generated via content-based matching.

Unfortunately, conventional belief is that existing commercial search engines typically utilize one crawl of web content (the latest) without considering the complementary information concealed in web dynamics. Therefore, the generated rankings may be biased due to the deficiency of knowledge on page or hyperlink evolution, and the time-sensitive facet within search quality, e.g., freshness, has been neglected. While previous research efforts have focused on exploring the temporal dimension in the retrieval process, few of them evaluated their methodologies on a portion of real web with long history, and showed

[1] This thesis focuses on general ranking. Personalized ranking and search in social media are not in the scope of our study.

consistent superiority over the competitors that do not take web dynamics into account.

In this thesis, we consider the research question of how to utilize the changes of web pages and hyperlinks to improve search quality, in terms of freshness and relevance of search results. Based on search engine infrastructure [98], we propose to incorporate knowledge from web dynamics into three search components: document representation—incorporating the anchor text trends extracted from the changes of in-coming links to better quantify anchor text importance, web authority estimation—incorporating web freshness inferred from user maintenance activities into a semi-Markov model to demote stale but otherwise authoritative pages, and learning to rank system—optimizing freshness and relevance by considering query differences on temporal characteristics. These three components interweave with each other. The first and second ones respectively discuss extracting dynamic and static ranking signals that incorporate temporal information, while the last one presents how temporal aspects of queries can influence the design of a learning to rank system framework. To evaluate our proposed methods, comparable experiments are conducted based on a large-scale archival web corpus collected by the Internet Archive[2] from January 2000 to December 2007.

Of course, the importance of web dynamics can be extended beyond web search. Advertisement rankers have to consider the problem of balancing between multiple optimization criteria; related search and auto-complete suggestions must provide users with fresh and relevant alternatives to their queries; vertical search [49] ranking and triggering can be

[2] http://www.archive.org

affected by temporal changes; profiling people and their reputation in social networks can be enhanced by using their historical status and connections.

We next introduce the perspectives from which we incorporate web dynamics into these three search components in Section 1.2.

1.2 Outline

In this thesis, we propose to utilize the changes of web pages and hyperlinks to improve three search components: document representation—the creation time of anchor text is taken into consideration when representing target page content; web authority estimation—page and hyperlink maintenance activities are incorporated to mitigate the problem that traditional link-based ranking algorithms usually favor old pages; and, learning to rank systems—the temporal characteristics of information needs are incorporated into separate ranker training to better optimize freshness and relevance simultaneously.

Our key contributions are as follows.

- We propose to incorporate web dynamics into document representation, web authority estimation and learning to rank systems respectively. For document representation, we are the first that consider temporal contexts of anchor text for weighing anchor text importance. For web authority estimation, we are the first that utilize web maintenance activities to bias the behaviors of random surfer models. For learning to rank systems, we are the first that optimize freshness and relevance by considering the temporal characteristics of queries.

- We empirically verify the proposed methods by conducting experiments based on (or depending on) a large-scale real-world archival web corpus, and demonstrate their superiority over existing state-of-the-art.

For search engine engineers, our work unravels how temporal factors can be helpful in designing an on-line search service. For academic search engine researchers, it studies more effective ways of enhancing anchor-based retrieval models, estimating web authorities, and optimizing multiple objectives in learning to rank.

Our work operates on the environment of an archival web corpus. Each page and hyperlink associates its past maintenance activities, i.e., the time point on which it was created, updated, and/or removal is known. While the accuracy of such information strongly depends on the crawl strategies in practice (See Section **??** for details.), we in this thesis assume that the information we utilized is entirely accurate, targeting at improving different search components.

1.2.1 Using historical anchor text to enhance document representation

Anchor text has been recognized as useful complementary information for describing the content of target web pages [46]. Typical ways of estimating its importance depend on the anchor text popularity and link structures. Inferring such information from one web snapshot may suffer from the deficiency that a single web snapshot is not able to capture the variability of link structure. It has been shown that web pages disappear at a rate of 0.25-0.5% per week [57]. Local link structures with sudden changes might indicate link spam. Therefore, the influence of transitory links/pages and spam links may result in inaccurate estimation of anchor text importance.

We propose a novel temporal anchor text weighting method to incorporate the trends of anchor text creation over time, which combines historical weights of anchor text by propagating anchor text weights among snapshots over the time axis. In this way, anchor text importance can be estimated based on a more stable status— via proximity-based density kernel functions mapping onto multiple nearby local web graphs on the time axis.

The detailed contributions are as follows.

- We propose a novel temporal anchor text weighting method to incorporate the trends of anchor text creation over time; and

- We conduct experiments on a real-world web corpus, and demonstrate that propagating historical anchor text weights through time can achieve significant and consistent ranking improvements over several representative variants that do not take historical anchor text into account.

1.2.2 Quantifying web freshness for estimating page authorities

In-coming links reflect the popularity of web pages from the perspective of other pages (via link structures). Traditional link based algorithms, such as PageRank [98] and HITS [78], only consider one snapshot of the web graph without considering when hyperlinks were created, updated, and removed. In this way, the authority scores are biased toward old pages given that these pages have more time to attract in-links pointing to them, and so the authoritative but stale pages may achieve higher rankings. From users' viewpoint, failing to promote fresh search results can negatively affect the user experience, and make the search engine appear stale.

We propose a temporal web link-based ranking scheme, which incorporates features from historical author activities. We quantify web page freshness over time from page and in-link activity, and design a web surfer model that incorporates web freshness, based on a temporal web graph composed of multiple web snapshots at different time points. It includes authority propagation among snapshots, enabling link structures at distinct time points to influence each other when estimating web page authority. In this way, fresh web pages tend to attract more authority flows even if they have fewer in-coming links.

The detailed contributions are as follows.

- We propose a novel method to quantify web freshness from authors' maintenance activities on web content over time, from the perspectives of page freshness and in-link freshness;
- We design a novel method to incorporate web freshness into authority propagation to favor fresh pages;
- We explore a series of proximity-based density kernel functions to model authority propagation among web snapshots; and
- We conduct experiments on a real-world archival web data set and show the superiority of our approach on ranking performance in terms of both relevance and

freshness.

1.2.3 Optimizing freshness and relevance for learning ranking models

Freshness of results is important to modern search. Failing to recognize the temporal aspect of a query can negatively affect the user experience, and make the search engine appear stale. While freshness and relevance can be closely related for some topics (e.g., news queries), they are more independent in others (e.g., time insensitive queries). Therefore, optimizing one criterion does not necessarily improve the other, and can even do harm in some cases.

We propose a machine-learning framework for simultaneously optimizing freshness and relevance, in which the trade-off is automatically adaptive to temporal characteristics of the query. This supervised framework leverages the temporal profile of queries (inferred from pseudo-feedback documents) along with the other ranking features to improve both freshness and relevance of search results.

The detailed contributions are as follows.

- We propose a novel extension to an existing learning to rank framework to optimize for both freshness and relevance;
- We introduce a new loss function that emphasizes certain query-document pairs for better optimization;
- We investigate the correlation between freshness and relevance and compare it across temporal and non-temporal queries; and

- We introduce hybrid NDCG, a new variant of NDCG [71] that considers both freshness and relevance labels in evaluation.

We next introduce how we organize this dissertation in Section 1.3.

1.3 Overall Layout

This thesis is organized as follows.

In Chapter 2, we introduce the background of the thesis and the motivation for my thesis research. The thesis background includes search engine infrastructure; basic retrieval models and field retrieval models; document representation for generating dynamic ranking signals; web authority estimation for generating static ranking signals; and learning to rank systems for combining dynamic and static ranking signals. Motivated from the appropriate parts of these backgrounds, I present my thesis work in the following chapters of this thesis.

In Chapter 3, I focus on the document representation, exploring the ways how anchor text complements document content to improve search relevance. I introduce one approach which differentiates anchor text according to the time points on which their associated hyperlinks were created. It tries to incorporate the variance of link structure and anchor text weights into document representation.

In Chapter 4, I focus on web authority estimation. Two approaches are proposed. One of them aims to overcome the problem that previous approaches unfairly favor old pages. We incorporate web freshness inferred from web content maintenance activities into controlling authority flow distribution, referred to as "*T-Fresh*". The other one utilizes the correlation between different types of web freshness as a confidence indicator of web page freshness scores, referred to as "*C-Fresh*".

In Chapter 5, I focus on improving learning to rank systems. The motivation comes from the conjecture that search quality facets (i.e., freshness and relevance in our work) correlate with each other in the way depending on queries' (temporal) characteristics, and so we design a learning to rank system which optimizes ranking objectives simultaneously depending on query types. I call this system prototype "CS-DAC".

In Chapter 5.4, I introduce the evaluation platform on which I will show the superiority of the proposed system prototype "CS-DAC". Here, we will describe (1) how we collect queries used in ranking evaluation; (2) how we collect groundtruth on freshness and relevance of search results; and (3) the metrics for evaluating ranking performance.

Based on such platform, I next report the ranking performance of "CS-DAC" and compare it with the existing state-of-arts in Chapter 5.5. I also highlight important findings that demonstrate the unique properties of our approaches.

In Chapter 6, I conclude the impacts and limitations of my thesis work. I also summarize the important findings inferred from deeper analysis of experimental results, suggesting their usability in different scenarios. I end by envisioning the future directions of my thesis work.

Chapter 2

Background and Motivation

2.1 Introduction

The main goal of search service is to improve users' search experience by generating appropriate web document rankings to satisfy users' information needs. To achieve this, main efforts focus on the search quality of results, as interpreted through specific ranking criteria. Representative ones include relevance, diversity [1, 63, 112], efficiency [118, 120, 119], and freshness [44, **?**, 50, 51]. Neglecting any one of them can negatively affect user experience. Diversity reflects the richness of information contained in search results. Failing to generate diverse results could make search engines provide duplicate answers. Efficiency reflects the complexity of ranking models. It is known that more complicated models may generate better document rankings at the price of spending more time, and too much waiting time may hurt user experience. Freshness quantifies how fresh the search results are. Failing to generate fresh results makes search engines look stale.

To leverage all search criteria within one ranking list is not a trivial task. The first problem is to correctly clarify the definition of each criterion, which sometimes even vary with query types. Take freshness as one example. Freshness can be interpreted in different ways. For certain temporal queries such as breaking news, freshness is more meaningful when the actual page content reflects new information. Whereas, for non-temporal (time-insensitive) queries, it makes more sense to interpret freshness as the recency of page maintenance with respect to the time point of generating ranking lists (suppose web pages contain such information). These two interpretations for freshness may be correlated to some extent but are not the same, considering the pages updated recently tend to record fresh information. One may notice that both explanations of freshness can influence user search experience.

Given a clear definition of each facet, the next problem is to optimize ranking by considering the balance among multiple search facets. It is not trivial since different ranking criteria may correlate with each other, depending on query types. Take the relationship between freshness and relevance as an example. For certain temporal queries such as breaking news, relevance and freshness are highly correlated. As a result, a ranker optimized for returning fresh documents may produce satisfactory results. However, for queries that are not usually time-sensitive (e.g., “facebook”, “machine learning”), paying too much attention to freshness may significantly hurt ranking effectiveness in terms of relevance. As a result, a ranker that optimizes either freshness or relevance only may not be flexible enough to deal with the temporal characteristics of queries effectively.

While improving search quality is from users' viewpoint, engineers may focus on improving each individual search modules for satisfying users' search experience from system's viewpoint. In the following sections, we focus on search systems and their main components. We start by introducing the high-level search engine infrastructure, and then move to each individual component. Here, the main search components include document relevance estimation, web authority estimation, and learning to rank (i.e., using machine learning techniques for ranking documents). We next review how prior work incorporate temporal information of web content, hyperlinks, and queries into improving search quality for each component.

2.2 Search Engine Infrastructure

Brin and Page [22] in 1998 introduced the high-level architecture of Google, as a prototype of modern search engine systems. It is distinguished from traditional retrieval systems by two main points: (1) hyperlinks between web pages are utilized to differentiate pages complementing content-based matching; and (2) anchor text is used to enhance the content of target web pages.

Nowadays, a web search system mainly includes a crawler, an indexer, a ranker learning module, and a query processor. A crawler is a means of providing up-to-date copies of web pages by following hyperlinks according to a certain strategy (e.g., breadth-first). This process aims to support index for page content for fast searches. Usually, web pages

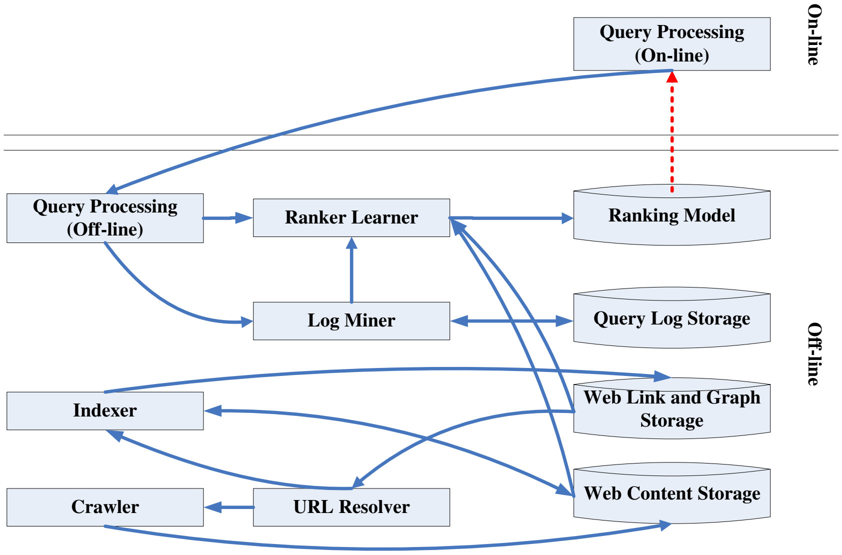


Figure 2.1: High-level web search system architecture.

are first crawled by a set of distributed crawlers, and then are stored onto the servers responsible for storing web content. The list of URLs to crawl is generated from the URL server, and sent to individual crawlers. Each page is assigned one docID.

An indexer parses web pages, recording term occurrence, fonts, positions, which are used to generate a partially sorted inverted index. The indexer also functions to parse the out-going links within web pages and their associated anchor text, and build up the web link graph. The URL resolver processes the parsed out-going links, and normalizes their format to match the docIDs in existing systems. This process generates the inverted indexes for all terms in the vocabulary, according to page content. It also records the connectivity (via hyperlinks) and descriptive context (via anchor text) between web pages.

These outputs facilitate the system to extract ranking signals characterizing document and/or query properties for better estimating document relevancy..

A ranker learning module trains rankers by using multiple ranking signals, including but not limited to those from link structures (e.g., PageRank) and page content (e.g., relevance scores generated by using traditional retrieval models). This step is typically done off-line. First, a set of training query-document pairs are selected. Second, how relevant a document is to a given query is judged by human editors. Third, ranking signals associated with each query-document pair are extracted. Fourth, rankers are trained by using relevance judgements and the ranking features associated with query-document pairs. It outputs ranking models for generating the rankings of new queries.

Given a new query (submitted by some search engine users), a query processer first modifies it into the format that fits the search system. Such operations include query parsing and stemming, query normalization, query rewriting and expansion, etc. This process may also involve profiling queries' characteristics, e.g., temporal characteristics. Such query related information is then used to compute dynamic (e.g., content-based) ranking features for web documents. Selecting the most suitable ranking model, the system finally generates the document rankings for that query. It was worthwhile pointing out that users' activities on search engines are recorded into query logs and are analyzed to enrich the preference of individual users, interpret users' intent, and infer document relevancy. Unfortunately, query log mining and analysis are out of the scope of this thesis.

Given this high-level system organization, we now consider how to incorporate web

Table 2.1: Notations in retrieval models (Sections 2.3 and 2.4).

Notation	Meaning
$tf(w,d)$	the frequency of term w in document d
df_w	the document frequency of term w
N	the total number of documents in the corpus
$\mathbf{q}$	the term importance vector representing query q
$\mathbf{d}$	the term importance vector representing document d
$\|d\|$	the document length
$avdl$	the average of document length over all documents
$tf(w,\mathcal{C})$	the frequency of term w in the corpus
$\|\mathcal{C}\|$	the total number of terms in the corpus
$tf(w,fi,d)$	the frequency of term w in field fi of document d
$\|d(fi)\|$	the length of field fi of document d
$avdl(fi)$	the average length of field fi over all documents

dynamics into the modules of estimating anchor text importance, computing web authorities, and training ranking models respectively. In the remainder of this chapter, we review previous research work on improving each of these modules. We start by introducing the general background of each individual module, and then present how prior work utilized web dynamics to improve these modules.

2.3 Basic Retrieval Models

Traditional information retrieval studies content-based ranking signals for search systems, which captures the semantic matchability between queries and documents. We start by introducing some notations in Table 2.1.

2.3.1 Similarity-based models

Boolean retrieval models

The idea of automatic retrieving from stored knowledge dates back to 1945 [27]. Early IR systems are boolean systems, in which users express their information needs through complex combinations of the operators, such as "AND" and "OR". While some users may appreciate control in expressing their information needs in the retrieval process, for many users, it is difficult to form the most accurate query to represent their needs. In addition, such systems usually generate document rankings by factors unrelated to relevance, such as the creation date and the alphabetical order of author names. Compared with boolean systems, ranked retrieval systems demonstrate their superiority in the sense that documents are ranked by their relevance, potentially better serving users' needs through further differentiation between documents. It is especially important for modern web search given the huge information pools on the web.[1]

Vector space model (VSM)

For ranked retrieval systems, the question comes to estimating document relevancy for a given query. One of the earliest and representative statistical retrieval model is Vector Space Model (VSM) [111]. It represents each document or query as a vector of terms, with a weight indicating the importance of each term with respect to the document or query

[1] Previous work has well recognized that document relevancy to the query is an important factor influencing users' satisfaction [88].

individually. To generate document rankings, two critical problems are: (1) estimating individual term importance for documents or queries; and (2) quantifying the similarity between each pair of queries and documents.

To solve the first problem, VSM operates on the bag-of-words model, i.e., every document and/or query is represented as a set of terms which are independent of each other (i.e., not considering term context). The term weighting strategy captures two heuristics: (1) documents that have more query term occurrence tend to be more relevant; and (2) if the total number of documents that contains a target term is large, the term importance should be deemphasized since such terms are less discriminative. These two points can be further captured by *term frequency* (TF) and *inverse document frequency* (IDF). Previous work called this weighting strategy TF-IDF term weighting, defined as:

$$TF\text{-}IDF(w,d) = tf(w,d) \times \log \frac{N}{1+df_w} \tag{2.1}$$

To solve the second problem, the Vector Space Model operates on document similarity theory, that is, the matchability between query and document is measured by the angle of their term weight vectors, more specifically, *cosine similarity*, defined as:

$$\cos\theta_{q,d} = \frac{\mathbf{d}\cdot\mathbf{q}}{\|\mathbf{d}\|\cdot\|\mathbf{q}\|} \tag{2.2}$$

In summary, similarity-based models such as the Vector Space Model generate document rankings based on the similarity between documents and the query. Such similarity is computed based on the term vectors representing the query and the documents respectively.

2.3.2 Statistical retrieval models

More recent research focuses on interpreting document relevancy in a probabilistic manner, among which an important family of probabilistic retrieval models follows the *Probabilistic Ranking Principle* (PRP) [108, 73, 105]. The short description of PRP is that if retrieved documents are ordered by decreasing probability of relevance on the data available, then the retrieval system's effectiveness is the best that can be obtained from the data. The relevance estimation of one document is independent of other documents.

Two representative branches of probabilistic retrieval models are BM25 [107] and language models [88]. BM25 incorporates one hidden binary variable (referred to as "Eliteness") associated with each term-document pair, and the estimation of document relevancy directly depends on these "Eliteness" variables. The model assumes the term frequency follows a 2-Possion distribution, which results in a non-linear term frequency component in the term weighting function. It is defined as:

$$BM(q,d) = \sum_{w\in q} \underbrace{\frac{tf(w,d)}{k_1((1-b)+b\frac{|d|}{avdl})+tf(w,d)}}_{\text{TF component}} \log \underbrace{\frac{N-df_w+0.5}{N+0.5}}_{\text{IDF component}} \tag{2.3}$$

where k_1 and b are free parameters. Equation 2.3 demonstrates that BM25 is composed of term and document frequency components, which is consistent with the spirit of tf-idf term weighting strategy in Section 2.3.1. Note that more complex BM25 versions differ on the IDF component (i.e., using Robertson-Spark-Jones model [109]) which further relies on relevance feedback.

For language models, the documents are ranked by the probability of generating query terms. Language models assume that each document (viewed as a bag of words) draws from a multinomial distribution over terms, and so the main research efforts focus on how to infer such a distribution from each document for better retrieval. To achieve this, one natural way is the query likelihood model, defined as:

$$p(q|d) = \prod_{w \in q} p(w|d) = \prod_{w \in q} \frac{tf(w,d)}{|d|} =_{rank} \sum_{w \in q} \log \frac{tf(w,d)}{|d|} \tag{2.4}$$

However, query likelihood model fails to provide the generative probability estimation for the terms not appearing in the documents (i.e., "zero probability" and "term sparsity" problems).

To overcome this deficiency, researchers have proposed a variety of smoothing strategies. Representative ones include Jelinek-Mercer (JM) smoothing and Dirichlet (Dir) smoothing [88]. These smoothing approaches benefit the generative probability estimation in that (1) they make the generative probability estimation more discriminative; and (2) they help achieve optimal performance for verbose queries through modeling query noise [136]. Jelinek-Mercer smoothing linearly interpolates background probability into modeling each individual document, defined as

$$p'(w|d) = (1-\lambda)p(w|d) + \lambda p(w|\mathcal{C}) \tag{2.5}$$

where $p(w|d)$ is the probability generated from document d using query likelihood model, and $p(w|\mathcal{C})$ is the probability generated from the whole corpus (background), and λ is a parameter in $[0,1]$, controlling the trade-off of these two portions. Dirichlet smoothing uses a dirichlet prior to smooth the multinomial probability generated from query likelihood model, defined as:

$$\begin{aligned} p'(w|d) &= \frac{tf(w,d)+\mu p(w|\mathcal{C})}{|d|+\mu} \qquad (2.6) \\ &= \frac{\frac{tf(w,d)+\mu p(w|\mathcal{C})}{|d|+\mu}}{\frac{\mu p(w|\mathcal{C})}{|d|+\mu}} \times \frac{\mu p(w|\mathcal{C})}{|d|+\mu} \\ &=_{rank} \log\,[1+\frac{tf(w,d)}{\mu}\frac{|\mathcal{C}|}{tf(w,\mathcal{C})}] - \log(|d|+\mu) + \log tf(w,C) \end{aligned}$$

where μ is the smoothing parameter. Compared with JM smoothing, Dirichlet smoothing takes document length into consideration, penalizing longer documents. Other probabilistic retrieval models include the divergence from randomness model [6] and the axiomatic approach to retrieval [55]. It is worthwhile pointing out that all these models are based on the bag-of-words model.

2.4 Field Retrieval Models

We reviewed several probabilistic retrieval models. One may notice that these models treat the terms in documents in the same way without considering the document fields in which terms appear. Here, document fields can include but not limited to title, heading, body, anchor text. Different document fields have different importance in estimating document relevancy. One example is that terms in the title field tend to better focus on the main topic of the document than the document body field. As a result, neglecting to differentiate document fields hurts the accuracy of document relevance estimation. To address this problem, researchers extended retrieval models to adapt multiple document fields. Representative models include BM25F [106] and field language models [88].

BM25F is the version of BM25 extended to apply to multiple document fields. The main idea is that the term frequency is computed by accumulating across all document fields, defined as:

$$BM25F(q,d) = \sum_{w \in q} \frac{\hat{tf}(w,d)}{k_1 + \hat{tf}(w,d)} \log \frac{N - df_w + 0.5}{N + 0.5} \tag{2.7}$$

where $\hat{tf}(w,d)$ is the normalized term frequency weighted over all fields, given by

$$\hat{tf}(w,d) = \sum_{fi=\{anc,doc,...\}} wt(fi) \frac{tf(w,fi,d)}{1 + b_{fi}(\frac{|d(fi)|}{avdl(fi)} - 1)} \tag{2.8}$$

where $wt(fi)$ is the trade-off among different document fields. To estimate document relevancy, the parameters $wt(fi)$, b_{fi} and k_1 can be learned, driven to optimize ranking metrics such as mean average precision.

Field language models are the version of language models extended to multiple document fields. Its main idea is that the probability of generating a query term is a mixture (linear combination) of the probabilities generated from each individual document field, defined as:

$$p'(w|d) = \sum_{fi=\{anc,doc,...\}} w(fi)p'(w|d,fi) \tag{2.9}$$

where $\sum_{fi=\{anc,doc,...\}} w(fi) = 1$. $p'(w|d,fi)$ can be estimated by using smoothing strategies, such as JM smoothing or Dirichlet smoothing as shown in Equations 2.5 and 2.6.

It is worthwhile pointing out that the focus of our work is to better weight the relative importance among multiple anchor texts for each page, so that field retrieval models perform better when using anchor text as one type of field in retrieval.

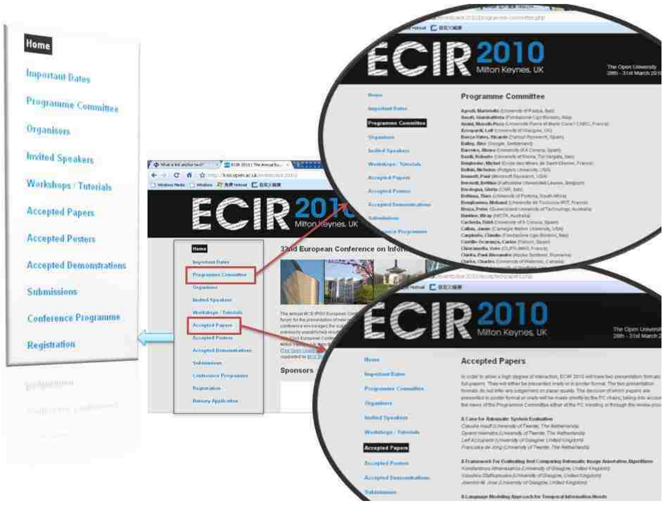


Figure 2.2: An example of anchor text on the web (The text within the red frame).

2.5 Enhanced Anchor Text Representation

We reviewed main statistical retrieval models and how they extend to multiple document fields. In this section, we focus on the anchor text field and review previous work on how it can benefits search relevance.

What is anchor text? When a web designer creates links pointing to other pages, she usually highlights a small portion of text on the current page, aiming to describe target page content or functionally link to target pages (e.g., “Click here”, “Last page”), and so facilitate visitors navigating to other information sources. Such highlighted text is referred to as *anchor text*. Figure 2.2 shows one example.

Why is anchor text important? Anchor text has been widely used in commercial search engines. Brin and Page [22] recognized the importance of anchor text to be associated with the page to which a link points. Representative research branches studied anchor text from two perspectives. One of them is to explore anchor distribution for better understanding of queries. Eiron and McCurley [53]'s work, which shows the properties of anchor text in a large intranet are similar to real user queries and web page titles, falls into this category.

More recent research focused on the other perspective, that is, using anchor text to enhance document representation for retrieval. Previous work has studied how to utilize anchor text for improving search relevance. [40] is among the earliest, in which the authors demonstrated the effectiveness of anchor text for answering the information need of finding specific web sites. The following work on using anchor text to improve search falls into three categories. One of them is to connect query intent with anchor text distribution on the web [82, 80, 60]. Their observation is that anchor text containing navigational query terms tends to have more skewed anchor-link distribution. It benefits web search in that we can use anchor text to customize ranking treatments for queries with different types of intent. The second category focuses on solving the anchor text sparsity problem [91, 132], i.e., relatively few web pages have considerable amount of anchor text associated with them. The reason is that the number of page in-coming links follows power law distribution [9]. The effort within this category is to incorporate appropriate complementary anchor text to enrich existing anchor text representation. The third category focuses on intelligent ways of anchor text importance estimation. Dou et al. [52]'s work that incorporated where source and target pages are from falls into this category.

However, we are not aware of any existing approach that smooths anchor text by its historical context to enhance document representation at the current time point. This distinguishes our proposed method from previous work. We will present how we incorporate historical anchor text context into anchor weighting for retrieval in Chapter 3.

2.6 Temporal Dynamic Ranking Signals

From Section 2.3 to Section 2.5, we reviewed some classical information retrieval techniques that estimate document relevancy with respect to queries. These techniques are only based on the statistics of term occurrence, and other aspects of queries and/or documents may be inevitably neglected. In this section, we review how previous work utilized the temporal characteristics of queries and documents to measure their temporal matchability. This further results in a series of temporal dynamic ranking signals, used as complementary indicator of how much documents are relevant with respect to queries. The research efforts on exploiting temporal signals that capture the dynamics of queries, web pages, hyperlinks, and user interaction to improve search quality fall into three categories.

The first category is to understand the temporal dynamics of information needs expressed through queries [81, 103]. The interpretation of queries may vary over time, and this directly influences the best answers to these queries. For many of the queries that correspond to events, the best answer may change over time (e.g., the latest SIGIR conference home page for the query “sigir conference”). In more extreme cases, the major intent behind the same query can temporally vary; for instance, the query “US open” is more likely to be targeting the tennis open in September, and the golf tournament in June. Kulkarni et al. [81] referred to this class of temporally ambiguous queries as shift topics. This observation inspires the ranking specialization that enables separate ranking treatment for different types of queries, which we will review in Section 2.11.

The second category is to characterize the temporal properties of web pages or terms. Motivated by the observation that the terms within each individual document demonstrates diverse stability, (i.e., the stability of term importance at different time points is diverse for documents) Elsas and Dumais [54] incorporated the dynamics of content changes into document language models and showed that their enhanced representations can improve retrieval effectiveness on navigational queries [24]. Their essential idea is that the terms with diverse variability contribute document relevancy in a different way. Compared with [54], Dong et al. [51] focused on the temporal properties of web pages. The authors used Twitter data to detect fresher documents for promoting their rankings. This family of work aims at directly generating ranking features.

The third category focuses on incorporating temporal factors into traditional retrieval models [121, 122, 76, 99, 13, 45, 74, 83, 90]. Typically this includes: (1) profiling query temporal characteristics, e.g., generating a temporal distribution over pseudo-feedback documents or based on query popularity over time [121, 122, 99, 45, 74, 83, 90]; and (2) emphasizing documents whose temporal characteristics are close to the query's temporal profile, e.g., enhancing document representation by adding temporal dimension and then incorporating temporal matching into the search process.

2.7 Web Authority Estimation

We reviewed previous work on retrieval models (Section 2.3 and Section 2.4), the ways of using anchor text to enrich document representation (Section 2.5), and the ways of incorporating web dynamics into dynamic ranking features (Section 2.6). These portions aim at producing dynamic ranking features used by search engines. In this section, we move to the generation of static ranking features that are independent of queries. Link analysis algorithms are one group of representative approaches for this purpose. Link analysis methods aim to compute web authority that measures the quality of web content, and so provide complementary information that differentiates web pages sharing similar content. In this way, the rankings generated by content-based matching can be further enhanced. Web authority estimation can be described as a stochastic process whose behavior depends on the link structure of the web. Representative approaches include PageRank [98] and HITS [78]. The underlying assumption is that pages give recommendations (distribute their authority) to the ones to which they point[2]. We start by introducing some notations in Table 2.2.

[2] For HITS, pages give recommendations to the ones pointing to them (the ones they point to) for being a hub (an authority). Authority and hub scores reinforce with each other via hyperlinks.

Table 2.2: Notations in web authority estimation (from Section 2.7 to Section 2.9).

Notation	Meaning
$O(p)$	out-degree of page p
$I(q)$	in-degree of page q
N	the total number of pages on the web
d	the probability of a random jump in the random surfer model
$A(p)$	the authority score of page p (HITS)
$H(p)$	the hub score of page p (HITS)
$p \rightarrow q$	there is a hyperlink pointing from page p to page q
$f(p)$	the "freshness" function of the page p (T-Rank)
$f(p,q)$	the "freshness" function of the hyperlink from page p to page q (T-Rank)
$a(p)$	the "activity" function of page p (T-Rank)
$a(p,q)$	the "activity" function of the hyperlink from page p to page q (T-Rank)

The PageRank algorithm operates on a random surfer model that simulates a Markov chain. Consider a surfer on the web. Suppose she is currently on page A, at the next step, she can choose to follow one of A's outgoing links to reach a page or randomly jump to any one page on the web. The PageRank score is computed by the probability of this surfer reaching a page. It is defined as follows:

$$PR(q) = d \sum_{p:p \rightarrow q} \frac{PR(p)}{O(p)} + (1-d)\frac{1}{N} \tag{2.10}$$

where O(p) is the out-degree of page p, and N is the total number of pages on the web. Such a model simulates a Markov chain, i.e., each web page is one state, and the transition between states is determined by the link structure (out-degrees) and damping factor d. The PageRank score is the stationary probability on each state. While the PageRank scores compromise the principle eigenvector of the transition matrix determined by link structure, faster PageRank computation is proceeded through iterative power methods [48, 10].

The HITS algorithm assumes each web page has two roles, i.e., as an authority and a hub. A good hub points to good information resources. A good authority contains good information, which is pointed to by good hubs. The HITS algorithm operates in a recursive way, in which hubs and authorities reinforce each other, defined as

$$
\begin{aligned}
A(q) &= \sum_{p:p\rightarrow q} \frac{H(p)}{O(p)} \qquad (2.11)\\
H(p) &= \sum_{q:p\rightarrow q} \frac{A(q)}{I(q)} \qquad (2.12)
\end{aligned}
$$

In each iteration, we normalize the authority (hub) scores over all pages, so that their sum equals 1. This process finally converges, and the pages are ranked by their authority or hub scores depending on search tasks.

More recent link analysis methods incorporate additional information to control the authority flow between web pages. The purpose is to improve the rationality of their original assumption. Two representative perspectives are incorporating the (1) topicality and (2) temporality of web pages and hyperlinks into web authority estimation. We now review previous work on topical link analysis and temporal link analysis in Section 2.8 and Section 2.9 respectively.

2.8 Topical Web Link Analysis

Page topicality is important to influence the authority distribution among web pages. The underlying assumption is that the recommendation from topically similar pages receive more credit. The Intelligent Surfer model (IS) is among the earliest work, in which the random surfer prefers more similar pages to jump to. However, the expense of computing page-page similarity prevents it from being applied to large-scale web graphs. Haveliwala's topic-sensitive PageRank [68] is a milestone in this direction. It is efficient in the ways that (1) each page is represented by its topical distribution; and (2) the topic-oriented random surfer models are computed over the web graph. More recent work [94, 95, 96, 93, 43] followed this direction and demonstrated that finer-grained topic-sensitive authority distribution further improves the effectiveness of web authority estimation on ranking performance. These works also demonstrated that topical link analysis can benefit web mining tasks, including web community discovery [93], web spam classification [101], question answering systems [69], and expert finding [130].

2.9 Temporal Web Link Analysis

One may notice that traditional link analysis approaches estimate web authority by using one snapshot of link structure. And so, they may suffer from unfairly favor old web pages since they have longer time to appeal in-coming links to point to. Previous work on mitigating this problem [133, 35, 8, 15, 14, 129, 2, 85, 16] follows two branches. We now review them both.

Using web temporal properties

One branch incorporates the time-related properties of web pages into authority estimation. Yu et al.'s work in [133] was among the earliest ones, in which the authors incorporated the paper age into quantifying paper authority to improve academic search.

In addition to utilizing paper citations, the authors modified PageRank by weighting each citation according to the citation date. The authors referred it as **TimedPageRank (TPR)**, defined as:

$$PR(q) = d \sum_{p:p \to q} \frac{w_p PR(p)}{O(p)} + (1-d)\frac{1}{N} \tag{2.13}$$

Compared with Equation 2.10, every page associates with a decay factor w_p, which is an exponential function of the age of paper p. In this way, the citation influence decays over time. However, this work only associated one type of activity, i.e., link (citation) creation, into link analysis in the scenario of academic search. Similar in spirit with Yu et al.'s work [133] but implemented differently, Amitay et al. [8] credits the links pointed from fresh pages. Their work attached a timestamp to each link, approximating the age of the page's content and gave bonus only to the links from fresh pages, rather than combining the freshness of the page itself when estimate web page authority.

Berberich et al.'s work [16] focused on temporal aspects of both web pages and links in web search via the web dynamics from page and link creation, modification and deletion. They assumed users are equally interested in recency of information, in addition to the quality. They proposed to use "freshness" and "activity" to convey whether a page is up to date with respect to user's temporal interests and the frequency of changes respectively. These two aspects mutually control the random surfer's behavior. The transition matrix is defined as:

$$t(p,q) \quad = \quad w_{t1} \cdot \frac{f(q)}{\sum_{q':p \to q'} f(q')} + w_{t2} \cdot \frac{f(p,q)}{\sum_{q':p \to q'} f(p,q')} \tag{2.14}$$

$$
\begin{aligned}
&+w_{t3} \cdot \frac{avg_{p':p'\to q} f(p',q)}{\sum_{q':p\to q'} avg_{p':p'\to q'} f(p',q')} \\
&+w_{t4} \cdot \frac{a(q)}{\sum_{q':p\to q'} a(q')} + w_{t5} \cdot \frac{a(p,q)}{\sum_{q':p\to q'} a(p,q')} \\
&+w_{t6} \cdot \frac{avg_{p':p'\to q} a(p',q)}{\sum_{q':p\to q'} avg_{p':p'\to q'} a(p',q')}
\end{aligned}
$$

where $f(*)$ and $a(*)$ are the "freshness" and "activity" of pages or hyperlinks functioning on web dynamics, and $\sum_{i=1}^{6} w_{ti} = 1$ are the parameters controlling the tradeoff among different portions. This approach is referred to as **T-Rank**. However, due to the definition of "freshness" and "activity" functions, the activities occurring at different time points are not distinguished as long as they were all in the period of users' temporal interests, which could span wide ranges.

Our work differs from prior work in two ways. First, we model the web freshness from two different perspectives by building temporal link profiles and temporal page profiles from multiple types of activities over time. Second, the influence of activities on web freshness decays over time. We will present our detailed methodology in Chapter 4, and compare with **TimedPageRank** and **T-Rank** in Chapter 5.5.

Using web temporal trends

The other branch which incorporates temporal factors directly utilizes or mines trends from multiple snapshots of the archival web [14, 15, 129, 85]. Motivated from Cho et al.'s observation that the number of page in-coming links increases exponentially over time [35], Berberich et al. [14] analyzed the potential of page authority by fitting an exponential model of page authority. Its hypothesis is that the success with which web pages attract in-links from others in a given period becomes an indicator of the page authority in the future. The approach requires an archival publication corpus which contains multiple snapshots of publication citation networks at different time points. Three critical steps are as follows. First, compute the PageRank scores of publications within each snapshot using Equation 2.10. Second, normalize PageRank scores by dividing them by the minimum authority score in the same web snapshot, so that the minimum normalized PageRank score of the page in any snapshot equals 1 [15]. The purpose of this step is to make PageRank scores within different snapshots comparable to each other. Third, fit the normalized PageRank score series of each individual publication into an exponential model. The parameter that controls the exponential model growth rate is used as an indiction of publication potential instead of PageRank score. The authors referred this approach as **BuzzRank**.

Yang et al. [129] proposed a new framework which utilizes a kinetic model to explain the evolution of page authority over time from a physical point of view (referred to as **TemporalRank**). Page authorities are viewed as objects subject to both "driving force" and "resistance", and so page authority at any time point can be a combination of the current authority score resulting from "driving force" and the decayed historical authority score from "resistance". This process finally results in a decayed accumulation of historical authority scores based on past web snapshots, defined as:

$$\begin{aligned} TR_t(i) &= e^{-\frac{\lambda}{m}k} \quad (t=0) \qquad (2.15) \\ TR_t(i) &= TR_{t-1}(i) + \frac{\eta}{m} PR_t(i) e^{-\frac{\lambda}{m}(k-t)} \qquad (t = 1, \ldots, k-1) \end{aligned}$$

$$TR_k(i) \quad = \quad TR_{k-1}(i) + \frac{\eta}{m} PR_k(i)$$

where λ, m, and η are model parameters controlling the temporal decay, $PR_t(i)$ is the PageRank score of page i at time point t, and k is our interested time point. Empirical experiments demonstrated that authority estimation can benefit from increasing use of archival web content. However, one may notice that this approach did not consider the accumulation of incomparable authority scores caused by an inconsistent number of pages in distinct snapshots.

Other than web search, the idea of propagation of authority flows among different snapshots has been found in some other domains, such as social network analysis. Li and Tang [85] modeled the decayed effects of old publications in expertise search by allowing authority exchange only between successive snapshots of the time-varying social networks. This approach is referred to as **T-Random**.

Our work differs from these approaches in two ways. First, in our method each page in any snapshot is directly influenced by the same page in all the snapshots in a one-step transition decayed by the difference in snapshot times. This process captures a comprehensive interaction between pages at different time points naturally. Second, we propose and evaluate a series of proximity-based kernel functions to control the authority propagation among multiple snapshots. Again, we will compare our approach with **BuzzRank**, **Temporal-Rank** and **T-Rank** in Chapter 5.5.

2.10 Learning to Rank for IR

We reviewed previous work on retrieval models and web authority estimation. These research directions aim to generate ranking signals for the effective ways of differentiating between web pages. In this section, we focus on learning to rank. It studies how to learn effective ranking models that can leverage the relative importance of different ranking signals by using machine learning techniques. Compared with traditional ranking approaches, learning to rank has some advantages: (1) automatically tuning parameters; (2) combining multiple sources of evidence; and (3) avoiding over-fitting.

The standard data set is composed of a large number of queries. Each query is associated with multiple documents and their relevance labels (relevance judgements). The main goal of learning to rank is to learn the ranking model which achieves the best performance on certain ranking metrics, which are computed based on the consistency between the ranks of documents and the query-document relevance judgements. Here, representative relevance judgements are binary judgements (relevant vs. irrelevant), multiple-scale ratings (e.g., perfect>excellent>good>fair>bad), and/or judgements on preferential query-doc pairs (e.g., For query q, the human editor prefers document A over B.) Representative ranking metrics are MAP (mean average precision), NDCG (normalized discounted cumulative gain), MRR (mean reciprocal rank), and etc [88]. These ranking metrics are usually the average performance over all queries, are sensitive to the positions of documents in the

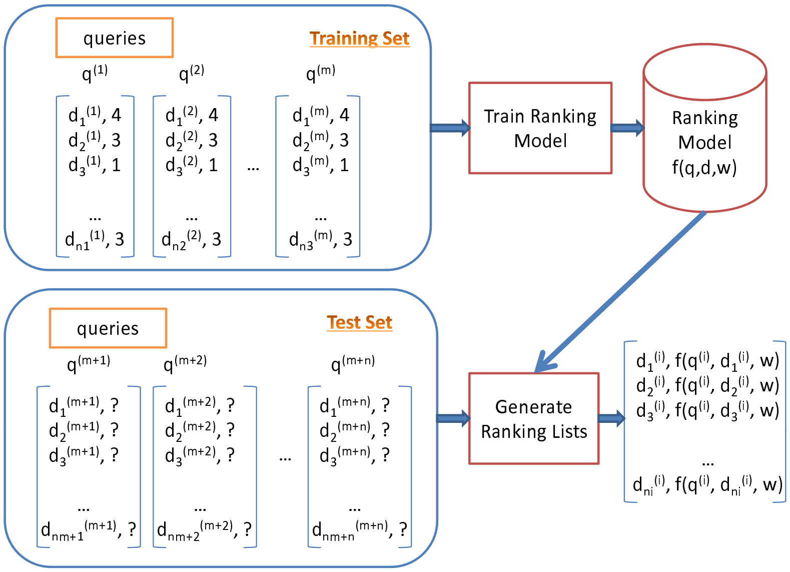


Figure 2.3: Framework of learning to rank for IR.

list, and are non-smoothed measures[3]. The framework of learning to rank for information retrieval is shown in Figure 2.3. Its process is: (1) the learning to rank system is trained by minimizing the loss from inconsistency between prediction and ground truth based on the training data set; and (2) the learned models are deployed into the ranking system to generate document rankings for unseen queries.

Three representative learning to rank approaches are: (1) pointwise approaches [92, 84]; (2) pairwise approaches [72, 58, 61, 26, 117, 137, 138, 38]; and (3) listwise approaches [25, 127, 134, 116, 131, 102, 29, 126]. Pointwise approaches reduce the document ranking problem to regression or classification on single documents. One representative

[3] We will introduce ranking evaluation metrics in Section 2.13.

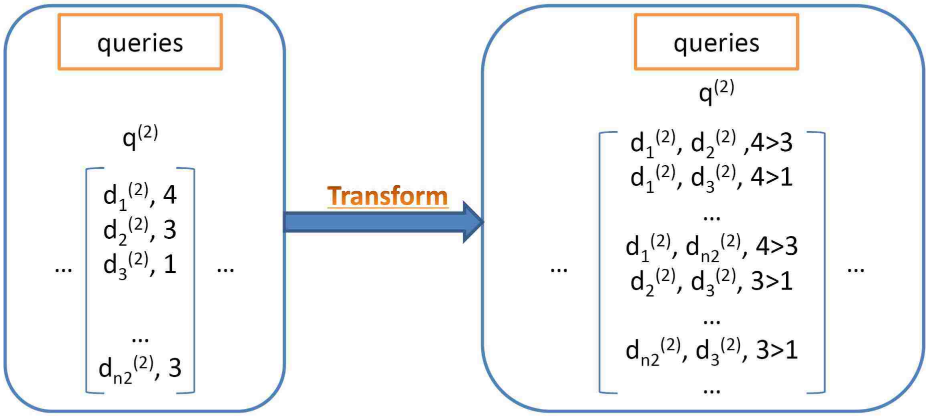


Figure 2.4: Transformation of pairwise learning problem.

example of conversion from classification to ranking is as follows. First, we train an individual classifier for query-document pairs with the same relevance judgements. Second, we convert the classifiers' outputs to probabilities by using logistic regression. Third, we convert the classification problem into ranking problem by: $S_i = \sum_{k=0}^{K-1} p_{i,k} \cdot k$. Empirical experiments demonstrate that converting the ranking problem to multiple ordinal classification problems outperforms converting it to the multiple-class classification problem, which further outperforms casting it to regression problem. In this way, pointwise approaches assume that relevance judgement is absolute, and query-independent in the sense that documents associated with different queries are put into the same category as long as they have same relevance scores, for training classifiers. As a result, ignoring the unique characteristics of queries may hurt ranker effectiveness. For example, the relevancy scores estimated by language models are much larger for popular queries than other queries on average.

Pairwise approaches mitigate the deficiencies of pointwise approaches. They cast learning to rank as a preferential relation learning problem. Given a query and a pair of associated documents, if one is more relevant than the other, then it is boosted in the training process to get a higher rank. Representative pairwise ranking approaches include RankSVM [72], RankBoost [58], RankNet [26], FRank [117], and etc. RankNet and FRank are similar in the sense that they are trained by minimizing the loss defined based on the consistency between the predicted probability of preferring one document over the other and the groundtruth preference. Their difference is the loss function, i.e., RankNet optimizes cross entropy defined as $C_{ij} = C(o_{ij}) = -\overline{P}_{ij} \log P_{ij} - (1 - \overline{P}_{ij}) \log(1 - P_{ij})$ while FRank optimizes fidelity defined as $F_{ij} = F(o_{ij}) = 1 - (\sqrt{\overline{P}_{ij} P_{ij}} + \sqrt{(1 - \overline{P}_{ij})(1 - P_{ij})})$, where P_{ij} is the groundtruth probability (1 if prefer doc i over j and 0 if prefer doc j over i), and $\overline{P}_{ij}$ is the estimated probability of preferring doc i over j, defined as $\overline{P}_{ij} = \frac{\exp(f(x_i) - f(x_j))}{1 + \exp(f(x_i) - f(x_j))}$. While RankNet has been widely deployed in real systems, FRank has shown its superiority on several scenarios. RankBoost and RankSVM utilizes AdaBoost [59] and SVM [39] respectively to perform pairwise classification, and so inherit their theoretical properties. While these pairwise approaches attempt to predict the relative preference between paired documents and no longer assume absolute relevance judgements, one may notice that the unique properties of ranking in information retrieval have not been fully modeled, i.e., there exists a gap between the predefined loss function and the ranking evaluation metrics.

The appearance of listwise approaches help mitigate such a gap. The reason is that the loss functions of listwise approaches are defined based on the consistency between the predicted document permutations and the ideal ones (based on relevance judgements), and ranking performance metrics, e.g., NDCG and MAP, are usually position-sensitive and defined based on a document list/sublist. Such reasoning suggests the two listwise directions: (1) directly optimizing IR evaluation metrics; and (2) defining listwise loss functions. Both of them face the challenges that IR evaluation metrics are non-smooth and are not differentiable, while most optimization techniques are designed for smooth and differentiable objective functions. To solve this problem, the first direction draws from (1) first convert the problem to another smooth and differentiable case, and then optimize it instead [127, 134, 116]; or (2) use optimization techniques designed for non-smooth and non-differentiable ranking scenarios [25, 131]. The second direction draws from the unique properties of ranking for information retrieval [102, 29, 126]. Representative properties include the relationship between the loss and ranking metrics, and the unbalanced popularity of URLs associated with the same training queries. Compared with pointwise and pairwise approaches, listwise approaches benefit ranking performance in that they directly optimize for ranking evaluation metrics.

So far we have reviewed three representative categories of learning to rank algorithms. It is worthwhile pointing out that while the state-of-the-art technologies show very close quality of the predictions from each other (suggesting the technique of learning to rank is relatively mature), new challenges are still not fully explored. Chapelle et al. [31] summarized some of these challenges, which include learning theory for ranking, online complexity versus accuracy, sample selection bias, and large-scale learning to rank. Among these challenges, one important challenge is recency ranking [50, 51], i.e., how to rank for temporal queries, balancing the trade-off between freshness and relevance of top search results. Its main problems include the adaptation of ranking models, given that the best trade-off between freshness and relevance may be sensitive to queries' temporal characteristics. To mitigate this, previous work has utilized the techniques of ranking specialization and multi-objective optimization. In the reminder of this chapter, we review these two techniques in Sections 2.11 and 2.12 respectively, drawn from which we propose our learning to rank for freshness and relevance work in Chapter 5.

2.11 Ranking Specialization

In traditional learning to rank approaches, information about the query type was ignored in ranking, which limits the effectiveness of ranking functions. For instance, navigational queries target specific websites, while informational queries have a broader range of relevant answers. Hence, their ranking models could be optimized in different ways depending on the query intent [75]. Query-dependent loss/ranking functions were introduced to address these issues [19, 20, 62]. The general idea is to adopt a query-dependent loss based on the query type (class). Geng et al. [62] proposed a k-Nearest Neighbor based method which trains a query-dependent ranking function for each query based on its nearest neighbors in the training set. Bian et al. [20] achieved better results by learning both multiple ranking functions (by minimizing query-dependent ranking risks) and query categorization (navigational, informational, transactional) simultaneously. Although the query-dependent loss function has been found superior to the query-dependent ranking method of Geng et al. [62], it still leaves a few issues unaddressed: (1) query categorization and taxonomies may not be available or could be too noisy; (2) external taxonomies may not necessarily provide the best way of splitting queries for training specialized rankers; and (3) such categories may not be fine-grained enough for training and ranking purposes. To overcome these problems, Bian et al. [20] proposed a divide-and-conquer framework (DAC) for ranking specialization and instantiated it with RankSVM [19].

Our approach differs from prior work given that it optimizes freshness and relevance simultaneously in an adaptive way. We enhance query representations by adding criteria-sensitive features that can capture different aspects (e.g., relevance, freshness) of query-document pairs. Each query is categorized according to both temporal and relevance features, and the final ranking is produced by merging the results generated from several different ranking models (See Chapter 5 for details.).

2.12 Multi-objective Optimization in Ranking

Training ranking models for multiple criteria beyond relevance, such as diversity, freshness, and efficiency, has been the subject of many recent papers [50, 51, 63, 118]. Dong et al.'s work on recency ranking [50, 51] is among the closest to our work; they consider freshness in instance labeling for training effective ranking models. They argued that freshness is especially important for breaking news queries and demoted the relevance labels of stale pages for training. Empirical experiments demonstrated that such demotion can result in significant improvements on both relevance and freshness. We similarly generate hybrid labels for documents based on their relevance and freshness grades, and show that the labels generated by our strategy are more effective than those demoted for training. Despite this resemblance, our optimization tasks are fundamentally different; Dong et al. [50, 51] studied learning single adaptive or over-weighting rankers that optimize for freshness and relevance primarily from the perspective of ranking adaptation.

Our work differs from theirs given that we investigate the multi-criteria ranking problem in a divide and conquer framework with balanced distribution of training data, and emphasize adaptive balance between different criteria.

2.13 Ranking Evaluation Metrics

Ranking evaluation metrics aim to measure the relevance of search results in an objective way. Representative ranking evaluation metrics include **Precision**, **NDCG** [71], **MAP**, etc. Each individual metric reflects one perspective of search relevance. We now review them one by one.

- Normalized Discounted Cumulative Gain (NDCG): It is especially designed for multiple-scale rating type relevance judgments, and is sensitive to document positions in the list. NDCG at truncation level k is defined as:

$$NDCG(\mathcal{Q},k)=\frac{1}{|\mathcal{Q}|}\sum_{j=1}^{|\mathcal{Q}|}Z_{kj}\sum_{m=1}^{k}\frac{2^{R(j,m)}-1}{\log_2(1+m)} \tag{2.16}$$

where $R(j, m)$ is the relevance score of the document at rank m for answering query j. Z_{kj} is the reciprocal of the ideal cumulative gain for query j at truncation level k, such that the discounted cumulative gain is normalized to 1 per query. Equation 2.16 demonstrates that NDCG penalizes more on the bad search results at top positions more than those at lower positions.

- Precision: It especially fits binary relevance judgments. It measures the number of relevant documents at truncation level k, defined as:

$$Precision(\mathcal{Q}, k) = \frac{1}{|\mathcal{Q}|} \sum_{j=1}^{|\mathcal{Q}|} \frac{\text{the number of relevant docs in top } k \text{ results}}{k} \tag{2.17}$$

- Mean Average Precision (MAP): It averages the precision at all the truncation levels on which relevant documents appear, defined as

$$MAP(\mathcal{Q}) = \frac{1}{|\mathcal{Q}|} \sum_{j=1}^{|\mathcal{Q}|} \frac{\sum_k Precision(\mathcal{Q}, k) \cdot \mathrm{I}(\text{doc@k is relevant})}{\text{the number of relevant docs}} \tag{2.18}$$

where I(doc@k is relevant) is an indicator function which equals 1 if the document at rank k is relevant, 0 if not.

While we mainly use the above three ranking evaluation metrics in this thesis, it is worthwhile pointing out that other ranking evaluation metrics exist to interpret search quality in different ways. For example, Winners Take All (WTA) and Mean Reciprocal Rank (MRR) emphasize the search quality at top positions, i.e., WTA quantifies the accuracy at top 1 position while MRR cares that the position on which the first relevant document appears. Therefore, the appropriateness of a given ranking evaluation metric depends on the characteristics of the application.

Chapter 3

Mining Anchor Text Trends for Retrieval

3.1 Introduction

The primary goal of this chapter is to incorporate the trends of the creation of page in-links associated with anchor text into measuring the anchor text importance for representing page content in the retrieval task. When used for retrieval, one anchor text might not be as useful as another, and so recent work [91, 52] has focused on how to determine the importance of anchor text for a given destination page. However, such work only considers one snapshot of the web graph (the current web), and so the influence from historical anchor text is effectively excluded.

More importantly, the creation of anchor text reflects how web content creators view the destination page. A historical trace of the variation in such viewpoints can help determine how to interpret the page. Consider a page which has 10 newly created in-links associated with a specific anchor text in the past 3 days. When compared with another page which only received ten in-links (with the same anchor text) within the past 10 years, the importance of the anchor text on the former page should be emphasized, even if the absolute weights based on the current snapshot cannot differentiate them.

Based on the above analysis, we operate on the assumption that better anchor text representation of pages can improve retrieval quality. We incorporate the historical trends on anchor text, i.e., the (dis)appearance of anchor text and its associated link structure, by propagating the anchor text weights among historical and predicted future snapshots over the time axis (See Section 3.3 for details.). Our work can be generalized onto other tasks, such as web page clustering and classification. It can also help to build time-sensitive document models.

Furthermore, we propose a variety of ways to incorporate the trends from historical snapshots to better estimate the importance of anchor text in the current snapshot. Finally, we verify our models via empirical experiments, and our experiments show significant improvement in retrieval quality on a real-world web crawl from the Stanford WebBase.

In the reminder of this chapter, we start by introducing the temporal anchor text data used in this work. We then describe our methods in Section 3.3, which utilizes temporal anchor text to better estimate the importance of anchor text for retrieval. The experiments in Sections 3.4 and 3.5 show the effectiveness of our approaches. We discuss and summarize our effots in Section 3.6.

3.2 Temporal Anchor Data

A destination page gets in-links from multiple source pages at different time points, each with distinct anchor text. We assign a timestamp to each pair of source and destination pages, which represents the creation time of the associated link. Naturally, we consider the item <source page, destination page, anchor text, creation time> to be unique[1]. If the anchor text on the link changes, we assume that the link associated with the old anchor text is removed and another link associated with the new anchor text is created.

Figure 3.1 demonstrates the variation of the self similarity of subsequent snapshots of a collection of anchor text terms from month to month over a five-year time period. We take the query "paris hilton" as one example. First, we achieve the top 2000 search results using BM25 [107] from the corpus at each of the past months. Second, we compute normalized TF-IDF scores of anchor terms associated with these 2000 search results. Third, we compute the L_1 distance of TF-IDF vectors on anchor terms in successive months. From Figure 3.1, earlier months show somewhat larger changes, while the changes are more moderate in later time periods. This seems sensible as many in-links were created during the time period from 2001 to 2002. However, we also found the change in 2004 has a larger deviation. We infer that in-links have sharp increase for some destination nodes, but not

[1] The position of anchor text within web pages is not considered in this chapter, while it also influences the estimation of anchor text importance.

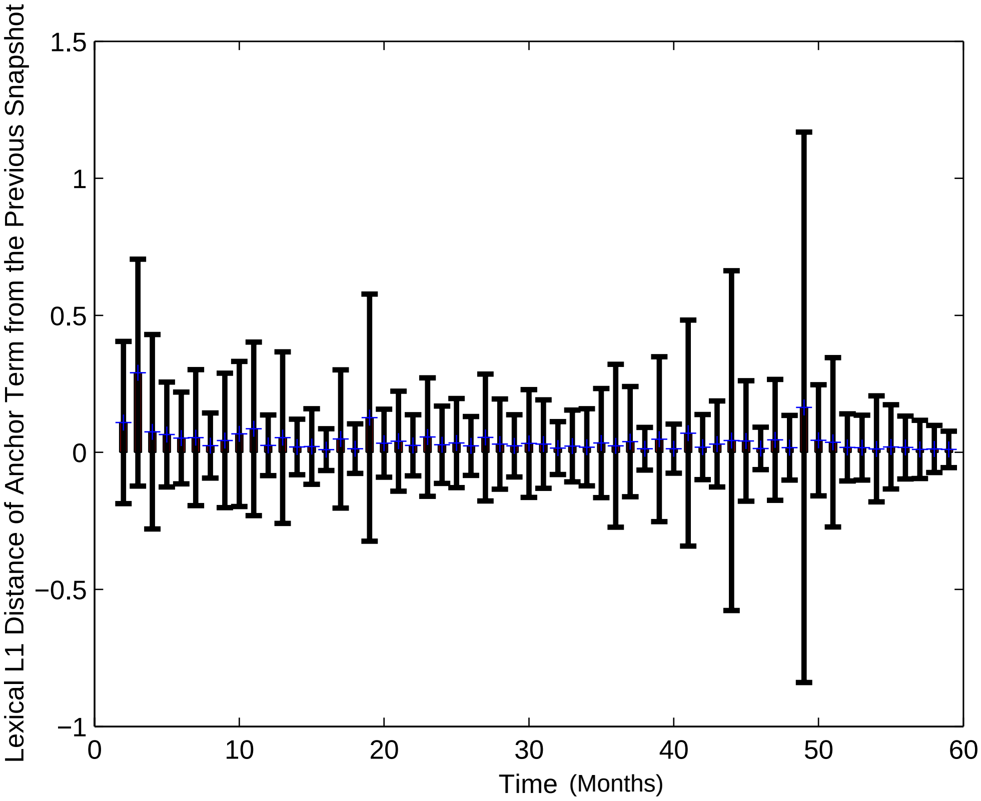


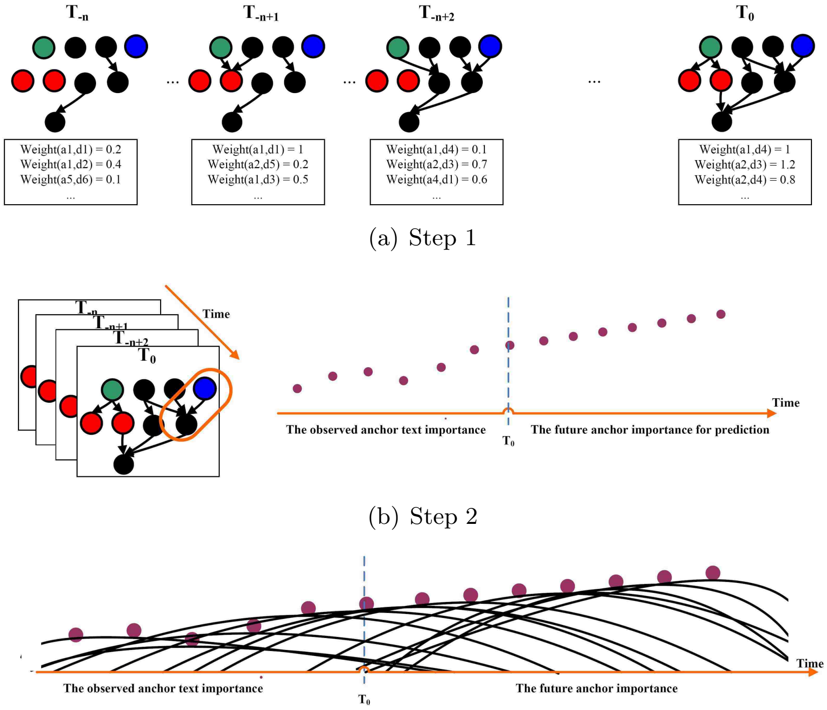
Figure 3.1: Average and standard deviation of the lexical L_1 distance of anchor text term distribution over time for each of the top 2000 search results of the query *"paris hilton"*. The X-axis is the time axis from early to late (from Jan. 2001 to Dec. 2005 with the time unit being 1 month). The Y-axis records the average and deviation of the lexical L_1 distance of destination nodes' anchor term distribution between two successive time points.

for others. To better understand the fine-grained variation of anchor text on links, we keep track of how the anchor text on each link change over time. The Jaccard coefficient of anchor terms on a specific link between two successive time points is 0.9954 ± 0.0514 on average. Based on these observations, we believe that the anchor text on links are relatively stable. Most anchor text does not change from the time point when the associated link was created to the time point when it was removed. The change in aggregated impact of anchor texts onto the relevance of a destination node can be potentially used to benefit web search. Motivated by these observations, we propose our temporal anchor text based retrieval method.

3.3 Temporal Anchor Text Based Retrieval

In this section, we describe our proposed methods which incorporate historical trends of page in-link creation rate and smooth the anchor text weights for destination pages in anchor text based retrieval. Our method requires a web graph and the time point t_0 on which it is crawled. Here, we define t_0 to be the current time point, and assume the retrieval evaluation is based on the situation at t_0. We follow the approach proposed by Metzler et al. [91] to determine weights on anchor text at each time point. Metzler et al. aggregated a set of unique *anchor text lines* for each given destination page, and calculated weights on them individually for improving search relevance. However, we propose using different weights on anchor text lines along different time points. Such weights on anchor text lines represent their importance on a given destination page at a specific time point. The output of our method is a collection of anchor terms and the final smoothed weights on them for a destination page at time point t_0. Specifically, our approach can be divided into the following three steps:

- aggregate anchor text lines and calculate weights on them for destination pages at each time point before t_0;



(a) Step 1

(b) Step 2

(c) Step 3

Figure 3.2: The overall procedure of our proposed approach.

- analyze the trend and use it to predict the possible weights on anchor text lines at the time points after t_0;
- propagate and diffuse the weights on anchor text lines along the time axis;

We illustrate the overall procedure of this approach in Figure 3.2.

3.3.1 Aggregate Historical Anchor Text

In order to better understand how to collect and weight the aggregated historical anchor text, we first describe how we weight the anchor text of the current snapshot. We use the methods in Metzler et al.'s method [91] to collect and weight anchor text for a specific web snapshot. While there are other ways to weight anchor text beyond Metzler et al.'s method [91], it is the only one to deal with *anchor text sparsity* problem. The reason we choose it as our basic anchor text weight estimator is that (1) it aims at enriching anchor text representation; and (2) the historical link information may sometimes be unavailable and deficient. We now briefly review the way of collecting and weighting anchor text in that work.

Given a URL u, all in-link pages P that are within the same site (domain) as u are collected as *internal pages*. Those in-link pages A that are in different domains from u are defined as *external pages*. The anchor text on the external pages are called *original anchor text*. For internal pages, we further collect the external pages of these internal pages. The anchor text on the newly collected external pages are known as *aggregated anchor text* of u. The original anchor text are weighted as follows:

$$wt(a,u) = \sum_{s \in S(u)} \frac{\delta(a,u,s)}{|anchors(u,s)|}$$

where $S(u)$ is the set of external sites that links to u, $\delta(a,u,s)$ is 1 iff anchor text line a links to u from site s. The aggregated anchor text are weighted in multiple ways; we choose two of them which are shown to have best performance in general in [91], defined as follows:

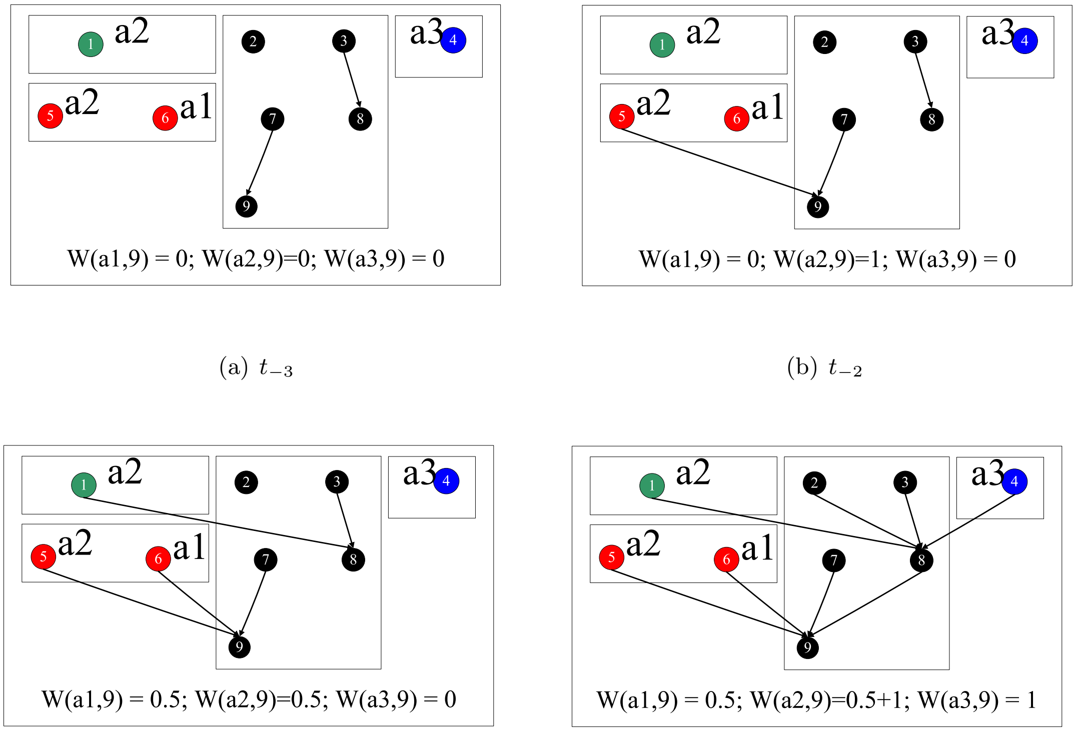
$$wt_{Min}(a,u) = \min_{u' \in N(u)} wt(a,u') \qquad wt_{Max}(a,u) = \max_{u' \in N(u)} wt(a,u')$$

where $N(u)$ is the set of internal in-linked pages and $wt(a,u')$ is the original weight of anchor text line a for URL u'.

Both original anchor text lines and external anchor text lines are used to enrich anchor text representation. We choose to use *combined* representation and *back off* representation to enrich destination pages' representation. Combined representation keeps the document structure and augments both original anchor text and aggregated anchor text, whereas back off representation exempts from the aggregated anchor text which have already appeared in original anchor text lines.

Once we have weights on anchor texts at the current time point t_0, we have actually known which links should contribute to anchor text weights. We keep track of these links by looking back to seek their creation time (see Section 3.3.4 for details). We define the difference of two successive time points as Δt, i.e., $\Delta t = t_i - t_{i-1}$. We map each link onto the time axis according to its creation time. If link l is created before t_i but after t_{i-1}, i.e., $t_{i-1} < t_{creation} < t_i$, then for any given past time point after time i, i.e., t_j for $t_i < t_j < t_0$, l is included in the snapshot at t_j. Given any time point t_i ($t_i < t_0$), we calculate the weight $w_i(a, u)$ of anchor text line a on the web page u based on all the links included at time point t_i.

Figure 3.3 shows an example of how the weights on anchor text change over time resulting from the creation of new links on the graph. To clarify this, we take Figure 3.3 (d) as one example to illustrate how we compute anchor text weights at each time point. The importance of anchor text $a2$ from page 5 to page 9 is 0.5 since two unique anchor text lines (i.e., $a1$ and $a2$) associated with page 9 are from the site colored by red. $a2$'s


a2
a2
a1
a3
1
2
3
4
5
6
7
8
9
W(a1,9) = 0; W(a2,9)=0; W(a3,9) = 0
W(a1,9) = 0; W(a2,9)=1; W(a3,9) = 0
W(a1,9) = 0.5; W(a2,9)=0.5; W(a3,9) = 0
W(a1,9) = 0.5; W(a2,9)=0.5+1; W(a3,9) = 1

(a) t_{-3}

(b) t_{-2}

(c) t_{-1}

(d) t_0

Figure 3.3: The variation of weights on anchor text caused by the creation of new links over time. The weights are calculated based on the combined representation of original and aggregated anchor text. Nodes (i.e., web pages numbered from 1 to 9) in different colors (also included in different rectangles) are from different domains.

importance to page 9 also passes through page 8 since $a2$'s importance on 8 is 1 and page 8 is one internal page of page 9. From Figure 3.3, with the creation of new links, the weights on anchor text for the target page keep increasing. Although such increase is the general case, we also notice that the weights on some anchor text lines may decrease when the number of other anchor text lines within the same site suddenly increases. The weights on anchor text actually depend on those on other anchor text with the same domain to some degree.

3.3.2 Quantify Trends to Predict Future

Quantifying trends of weights on anchor text can help to predict how the weights change at future time points. Given a destination page, if the importance of its particular anchor text increases more greatly than its other anchor text, we may have higher confidence to believe such anchor text should be emphasized in some way since the trend shows it may get a higher weight in the near future. Here, we assume that: (1) for a same target page, the anchor text created at closer time points tend to be more consistent; and (2) the weights on anchor text reflect the number of pages/sites pointing to the target page, using that anchor text.

ARIMA (Auto-Regressive Integrated Moving Average) [66] is a powerful way to predict time-series, but it is complex to use. Instead, we use linear regression on moving average of order m to predict the value at the next time point. The reasons are as follows: (1) we observe that weights on anchor text have stable and monotone trends through time once the anchor text begins associating with the destination page; (2) we tested the fitness of linear models on the weights (Max+combined) of individual anchor text lines over time. The average mean square error (MSE) is only 0.0656. Based on these observations, we believe the linear model can well fit the trends of historical anchor text weights.

Given a URL u and one associated anchor text line a, we have a series of historical weights $w_{-n}(a, u), w_{-n+1}(a, u), \ldots, w_0(a, u)$. We first use a sliding window with size $2k$

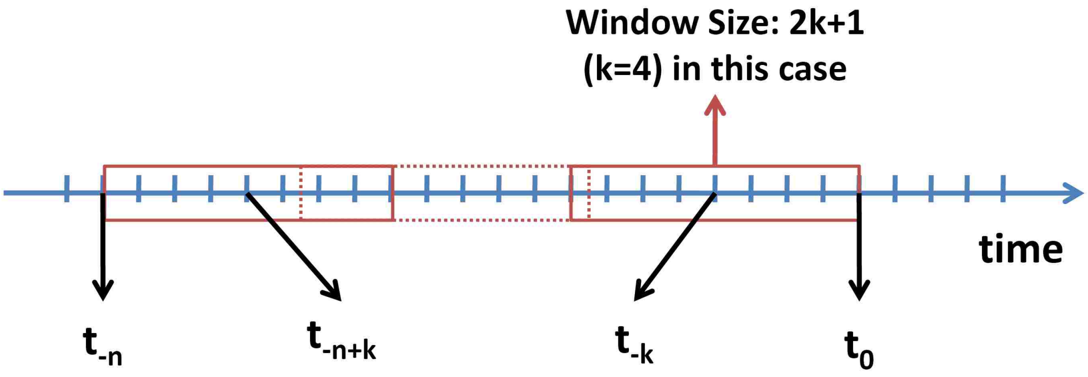


Figure 3.4: The computation of moving average of order k.

($k > 0$) to smooth the time series. We calculate a moving average of order $2k$ as the following sequence of arithmetic means:

$$\frac{\sum_{i=-n}^{-n+2k} w_i(a,u)}{2k+1}, \frac{\sum_{i=-n+1}^{-n+2k+1} w_i(a,u)}{2k+1}, \dots, \frac{\sum_{i=-2k}^{0} w_i(a,u)}{2k+1}$$

By using the sequence calculated above, we achieve the smoothed values from the time point t_{-n+k} to t_{-k}. The next step is to use linear regression to predict the possible average at time point t_{-k+1}. The model assumes the moving average of order $2k+1$ has a linear relationship with the time points given a pair of anchor text and destination page, which is given by:

$$\overline{w}_i(a,u) = b + c \times i, i \geq (-k+1) \tag{3.1}$$

We use existing evidence to estimate the parameters b and c. Once the weight $\widehat{\overline{w}}_{-k+1}(a,u)$ is achieved, $w_1(a,u)$ can be calculated by:

$$w_1(a,u) = \widehat{\overline{w}}_{-k+1}(a,u) \times (2k+1) - \sum_{i=-2k+1}^{0} w_i(a,u)$$

After we get the value of $w_1(a,u)$, we move the sliding window forward to calculate

		t_{-5}	t_{-4}	t_{-3}	t_{-2}	t_{-1}	t_0	t_1	t_2
w	Original weights	0.5	0.5	1	1	1.5	1.75	**1.78**	**2.30**
$\overline{w}$	Moving average		0.67	0.83	1.16	1.41	**1.68**	**1.94**	

Table 3.1: An example of predicting the future weights of anchor text on a destination node. The second line shows the moving average of order 3 (i.e., $k = 1$).

$w_i(a, u)(i > 1)$.

Table 3.1 shows an example of predicting the weights of at future time points t_1 and t_2. We first calculate the moving average of order 3 from t_{-4} to t_{-1}, and use them to predict the moving average at t_0 since the linear regression estimate the parameters b and c to be 1.6750 and 0.2650 respectively. Thus, the predicted weight on t_1 can be achieved from the moving average at t_0 and the weights at t_{-1} and t_0. In the same way, we can calculate the moving average of order 3 at t_1 and the weight at t_2.

3.3.3 Diffusing Temporal Anchor Text Weights

Analyzing the trends of anchor text weights on a destination page allows us to predict the anchor text weights in the future. However, in order to better measure the importance of anchor text lines at t_0, we need to combine both the predicted future weights and the historical weights. As discussed in the previous section, the predicted weights are extrapolated from historical trends, which help to differentiate two anchor text lines with identical weights at t_0. On the other hand, historical anchor text weights provide confirmation about what a destination page looks like. When we emphasize the predicted future weights, we give preference to newly created destination pages, since the new pages tend to have higher anchor text creation rate, and the predicted anchor text weights are usually overemphasized. Whereas, when we combine some historical weights, we likely emphasize old pages which have stable anchor text distribution. By combining both the historical weights and predicted future weights, we can harmonize the influence from these two sides.

Specifically, we imagine that the weights on an anchor text line at one time point can propagate through time to influence the weights of the same anchor text line at other time points for a given destination page. The intuition is that if an anchor text has a weight at a time point t_i, it can influence the weights on the same anchor text at other time points in a decayed way which is proportional to a temporal distance. Thus, weights on two close time points would have more influence on each other than those on two far time points. Furthermore, we assume that the change ratio of the destination page content will also influence the weight propagation since huge change is likely to cause such propagation to decay more quickly, that is, page snapshots with distinct content tend to associate with more diverse anchor text collections. Given a time window, we calculate weights at the middle time point by aggregating the discount weights from all time points within it.

We now describe our method to propagate the weights formally. Let γ be the size of time window T, i.e., the number of time points within the time window. Let a be an anchor text line. Let u be a destination node, and u_i be a destination node at time point t_i. $w_1(u, a)$, $w_2(u, a)$, $\ldots$, $w_\gamma(u, a)$ are the weights of a on u at time points within the time window T. The weights at time point $t_{\frac{\gamma}{2}}$ after combining the propagated weights of other time points within the time window is given by:

$$w'_{\frac{\gamma}{2}}(u,a) = \sum_{i=1}^{\gamma} f(u,\gamma,i) w_i(u,a) \tag{3.2}$$

where $f(u,\gamma,i)$ is the kernel function which determines the way of combining weight $w(u,a)$ at time point t_i.

Enlightened by previous work [47, 77, 100, 87] which used proximity-based methods, we use five modified kernel functions derived from Gaussian kernel (Equation 3.3), Triangle kernel (Equation 3.4), Cosine kernel (Equation 3.5), Circle kernel (Equation 3.6), and Rectangle kernel (Equation 3.7), which are defined by:

$$f_1(u,\gamma,i) = \exp[-\frac{1}{2}(\frac{i-\frac{\gamma}{2}}{\gamma(1+\overline{B}_u(i\leftrightarrow\frac{\gamma}{2}))})^2] \tag{3.3}$$

$$f_2(u,\gamma,i) = 1 - \frac{|i-\frac{\gamma}{2}|}{\gamma(1+\overline{B}_u(i\leftrightarrow\frac{\gamma}{2}))} \tag{3.4}$$

$$f_3(u,\gamma,i) = \frac{1}{2}[1+\cos(\frac{\pi(i-\frac{\gamma}{2})}{\gamma(1+\overline{B}_u(i\leftrightarrow\frac{\gamma}{2}))})] \tag{3.5}$$

$$f_4(u,\gamma,i) = \sqrt{1-(\frac{|i-\gamma/2|}{\gamma(1+\overline{B}_u(i\leftrightarrow\frac{\gamma}{2}))})^2} \tag{3.6}$$

$$f_5(u,\gamma,i) = 1 \tag{3.7}$$

where $\overline{B}_u(i\leftrightarrow\frac{\gamma}{2})$ is the average similarity between the destination page u's content at two successive time points within the range $[i, \gamma/2]$ if $i < \gamma/2$ or $[\gamma/2, i]$ if $i \geq \gamma/2$. Without loss of generality, we assume $i < \gamma/2$. $\overline{B}_u(i \leftrightarrow \frac{\gamma}{2})$ is defined by:

$$\overline{B}_u(i \leftrightarrow \frac{\gamma}{2}) = \frac{1}{\frac{\gamma}{2} - i} \sum_{i'=i}^{\frac{\gamma}{2}-1} B_u(i', i' + 1) \tag{3.8}$$

We compare the similarity of two snapshots of page u's content by comparing their associated language models via the Bhattacharyya correlation [18]:

$$B_u(i', i' + 1) = \sum_{v \in V} \sqrt{P(w|\theta_{u_{i'}})P(w|\theta_{u_{i'+1}})} \tag{3.9}$$

This metric renders a similarity score between 0 and 1. Although this similarity is only based on $P(w|\theta_u)$, we can consider combining other measures based on topic, timestamp, or out-link overlap so that all these measures can influence the probability of propagating the anchor text importance through the time axis.

3.3.4 Implementation

One key problem for utilizing temporal anchor text is that it is difficult to keep track of the information about when a link was created. And we are not aware of any previous work that mined the historical information by processing the data from the Internet Archive. In our experiments, given a link appearing in the current snapshot, we looked back to archival copies of the source page via the Wayback Machine portal of the Internet Archive [70]. We parsed these copies to get all out-links within the web pages, and checked whether the given link was still in the out-link collection and whether the anchor text associated with the given link had any change. If either the anchor text has changed or the link did not exist, we utilized the timestamp of the next latest copy to be the time when the given link was created.

3.4 Experiment Setup

3.4.1 Data set and Evaluation

Although many datasets, such as TREC .GOV collection [97], have been built for research purposes, they are usually small and biased, and cannot represent the characteristics of the real-world web graph. Hence, we choose to use a May 2005 crawl from the Stanford WebBase [34] as our dataset for ranking evaluation. This crawl has 58 million pages, and approximately 900 million links.

For ranking evaluation, 50 queries are selected from a set consisting of those frequently used by previous researchers, ODP category names, and popular queries from Lycos and Google. We list these queries in Table 3.2. For each query, we have relevance judgments of 35 URLs on average. When human editors (members of our research lab) judge each pair of <query, URL>, they are asked to give a score based on how relevant the URL is to the given query. The rating results in the selection among excellent, good, not sure, bad, and worse. We use a five-value scale which translates the ratings into the integers from 4 to 0. If the average score for this pair is more than 2.5, it is marked as relevant.

Based on the available relevance judgments, we evaluate the retrieval quality of our

harry potter	college football	diabetes
music lyrics	george bush	nfl
online dictionary	britney spear	pokemon
olsen twins	diamond bracelet	madonna
weight watchers	windshield wiper	brad pitt
playstation	jennifer lopez	maps
new york fireworks	moto racer	poker
halloween costumes	iraq war	tsunami
st patricks day cards	four leaf clover	games
the passion of christ	tattoos	jersey girl
automobile warranty	fox news	golf clubs
herpes treatments	paris hilton	pilates
skateboarding	taxes	seinfeld show
lord of the rings	hilary duff	american idol
angelina jolie	star wars	diets
final fantasy	janet jackson	poems
prom hairstyles	musculoskeletal disorders	

Table 3.2: Set of fifty queries used for relevance evaluation in WebBase.

ranking algorithms over the Normalized Discounted Cumulative Gain (NDCG) and Precision@10. We have introduced these metrics in Section 2.13.

3.4.2 Ranking Function

Combining different fields of web pages has been shown to be highly effective for retrieval on the web in previous work [135]. BM25F is such a ranking model, which combines term frequencies in different fields linearly for BM25 score calculation. In this work, we test our anchor text weighting strategies by combining body text and anchor text in the BM25F model for retrieval. While we introduced field retrieval models in Section 2.4, we now emphasize how to integrate the anchor text field into the retrieval model BM25F. Suppose $w_{body}(i, j)$ is the weight of term i for page j in the body field, i.e., the term frequency of term i in page j. Let $w_{anchor}(i,j)$ be the weight of term i in the anchor text lines associated with page j, which is calculated by:

$$w_{anchor}(i,j) = \sum_{a \in A(j)} wt(a,j) \times tf_{anchor}(i,a)$$

where $wt(a,j)$ is the weight on anchor text line a for the page j, and $tf_{anchor}(i,a)$ is the term frequency of i in the anchor text line a.

The aggregated term weights on i is a linear combination of weights i on anchor text and page body, which is given by:

$$w(i,j) = (1-\alpha) \times w_{anchor}(i,j) + \alpha \times w_{body}(i,j)$$

where α is a combination parameter, which controls the balance between term weights on anchor text and page body used in BM25F ranking function. The document length is calculated by the same method.

3.5 Experimental Results

Our goal is to demonstrate the superiority of our approach, which utilizes the historical anchor text information mined from the Internet Archive to improve search relevance. In this section, we report the results of our ranking evaluation. We start by showing how the proposed ranking algorithms significantly improve the retrieval quality. We then render some deeper analysis about the characteristics of these ranking algorithms with respect to the improvement of ranking quality.

3.5.1 Performance Comparison

As an overall comparison, we study the effectiveness of enlarging the window for propagating historical weights on anchor text lines over multiple aggregation functions and anchor text representation in this section. The selection of kernel functions and all parameters in BM25F are learned based on five-fold cross-validation. Our baseline is Metzler et al.'s method [91], operating on the latest snapshot. We show the comparison of ranking performance in Table 3.3. Given that about 97% inlinked pages do not have archival copies (we removed them), the improvement of using anchor text versus without using anchor text is not obvious. The performance of almost all combinations of window sizes, aggregation functions and document representation over all the metrics outperform the baseline significantly. Furthermore, the performance of all combinations of aggregation functions consistently increases with the window size, which indicates that the use of temporal inlinks, especially those with a long term historical context is a good resource to reflect the link evolution that can be utilized in improving the ranking quality in terms of document relevance. Furthermore, the combined aggregation functions outperform the Backoff approaches, which suggests that the benefits from the "confirmation" influence brought by duplicate anchor text lines outweigh the noise they introduce.

3.5.2 Deeper Analysis

Deeper analysis focuses on two research questions: (1) how our proposed approach benefits from different kernel functions for propagating anchor text weights; and (2) how our

Table 3.3: Performance comparison for different windows and different anchor text representations. The † and ‡ symbols demonstrate the performance has statistically significant improvement when compared with the baseline (Latest anchors) at the level of $p < 0.1$ and $p < 0.05$ by one-tailed student t test.

Baseline				
	P@10	NDCG@3	NDCG@5	NDCG@10
No anchors	1.6150	0.1860	0.1830	0.1749
Latest anchors	1.6170	0.1899	0.1846	0.1781
All historical anchors	1.6596	0.2023	0.1901	0.1856
Backoff+Max				
Window (Months)	P@10	NDCG@3	NDCG@5	NDCG@10
1	1.6383†	0.2019‡	0.1911†	0.1822†
2	1.6383†	0.2064‡	0.1945†	0.1858‡
4	1.6809‡	0.2064‡	0.1945†	0.1879‡
7	1.7234‡	0.2076‡	0.1984‡	0.1915‡
12	1.7234‡	0.2085‡	0.1990‡	0.1916‡
24	1.7660‡	0.2086‡	0.2002‡	0.1950‡
Backoff+Min				
Window (Months)	P@10	NDCG@3	NDCG@5	NDCG@10
1	1.6170	0.1956†	0.1901†	0.1813†
2	1.6170	0.2024‡	0.1913†	0.1829†
4	1.6596‡	0.2050‡	0.1921†	0.1853‡
7	1.7021‡	0.2063‡	0.1979‡	0.1892‡
12	1.7234‡	0.2072‡	0.1975‡	0.1909‡
24	1.7660‡	0.2073‡	0.1990‡	0.1943‡
Combined+Max				
Window (Months)	P@10	NDCG@3	NDCG@5	NDCG@10
1	1.6383†	0.2019‡	0.1889	0.1841‡
2	1.6809‡	0.2064‡	0.1935‡	0.1889‡
4	1.7234‡	0.2064‡	0.1951‡	0.1909‡
7	1.7660‡	0.2094‡	0.1972‡	0.1944‡
12	1.7660‡	0.2105‡	0.1980‡	0.1964‡
24	1.8298‡	0.2129‡	0.2025‡	0.2003‡
Combined+Min				
Window (Months)	P@10	NDCG@3	NDCG@5	NDCG@10
1	1.6170	0.1956†	0.1875	0.1830‡
2	1.6809‡	0.1994†	0.1902†	0.1875‡
4	1.7234‡	0.2033‡	0.1941‡	0.1899‡
7	1.7660‡	0.2081‡	0.1963‡	0.1937‡
12	1.7660‡	0.2092‡	0.1980‡	0.1958‡
24	1.8298‡	0.2115‡	0.2015‡	0.1996‡

method benefits from historical information, that is, the time span of web snapshots vs. ranking improvements.

To answer the first research question, we show the effectiveness of kernel functions used in propagating anchor text line weights in Table 3.4. The performance of the simple Rectangle kernel is arguably the best in general among all combinations of aggregation functions. Gaussian and Circle kernels show comparable performance, which outperform Triangle and Cosine kernels. This observation demonstrates that search results benefit from emphasizing both historical and predicted future anchor weights without deemphasizing the influence of time points far away from the current point. We infer that ranking quality will benefit from long-term temporal information rather than short-term since long-term information tends to express more stable trends.

To answer the second research question, we investigate the relationship between the average age of search results and the relative improvement of ranking quality in Table 3.5. We bucketize the queries according to the average age of their top 2000 search results. The queries in bucket 0 are those whose search results have the shortest average age, and the ones in bucket 3 have the longest average age on their search results. From Table 3.5, query results with longer ages benefit more by propagating anchor text weights from past time points, whereas the query results with shorter ages have better improvements by propagating predicted weights from future time points over all window sizes. By combining the weights on both past time points and future time points, the relative improvement is greater than only combining weights in one direction for most buckets in different window

Table 3.4: Performance comparison for different kernels for propagating temporal anchor line weights when the window size is 12. The kernels 1, 2, 3, 4, and 5 represent Gaussian kernel, Triangle kernel, Cosine kernel, Circle kernel, and Rectangle kernel respectively.

Baseline				
	P@10	NDCG@3	NDCG@5	NDCG@10
No anchors	1.6150	0.1860	0.1830	0.1749
Latest anchors	1.6170	0.1899	0.1846	0.1781
Backoff+Max				
Kernel	P@10	NDCG@3	NDCG@5	NDCG@10
1	1.7022	**0.2085**	0.1962	0.1897
2	1.7021	0.2044	0.1955	0.1900
3	1.7020	0.2044	0.1955	0.1900
4	**1.7234**	0.2063	0.1985	0.1899
5	1.7023	0.2050	**0.1990**	**0.1916**
Backoff+Min				
Kernel	P@10	NDCG@3	NDCG@5	NDCG@10
1	1.7021	**0.2072**	0.1953	0.1890
2	1.7019	0.2030	0.1950	0.1889
3	1.7019	0.2030	0.1950	0.1889
4	**1.7234**	0.2050	0.1955	0.1891
5	1.7021	0.2037	**0.1975**	**0.1909**
Combined+Max				
Kernel	P@10	NDCG@3	NDCG@5	NDCG@10
1	1.7457	**0.2105**	**0.1980**	0.1940
2	1.7447	0.2063	0.1955	0.1931
3	1.7447	0.2063	0.1955	0.1931
4	**1.7660**	0.2057	0.1966	0.1930
5	**1.7660**	0.2068	0.1977	**0.1964**
Combined+Min				
Kernel	P@10	NDCG@3	NDCG@5	NDCG@10
1	1.7447	0.2086	**0.1980**	0.1933
2	1.7438	0.2050	0.1946	0.1920
3	1.7438	0.2050	0.1946	0.1920
4	**1.7660**	**0.2092**	0.1957	0.1924
5	**1.7660**	0.2055	0.1967	**0.1958**

Table 3.5: Performance comparison for queries bucketized by the average age of search results. The weighting strategy is Combined+Max. P: Propagating weights on anchor text lines from past time points; F: Propagating predicted weights on anchor text lines from future points; T: Propagating weights on anchor text lines from both sides.

Window	Time	Bucket 0	Bucket 1	Bucket 2	Bucket 3
1	P	4.00%	2.72%	2.05%	2.92%
	F	4.00%	2.72%	2.05%	2.92%
	T	7.79%	2.67%	2.20%	3.54%
2	P	7.79%	2.75%	2.20%	3.54%
	F	7.79%	2.67%	2.20%	3.54%
	T	7.01%	6.44%	2.70%	4.27%
4	P	7.12%	6.32%	2.70%	4.27%
	F	8.82%	6.44%	2.63%	4.03%
	T	8.80%	7.12%	2.83%	4.27%
7	P	5.92%	7.11%	2.88%	4.58%
	F	7.24%	6.46%	3.04%	4.07%
	T	8.93%	7.18%	3.14%	4.27%
12	P	6.11%	4.78%	3.01%	4.64%
	F	8.08%	7.02%	3.03%	4.27%
	T	12.04%	6.18%	2.69%	4.27%
24	P	5.83%	2.88%	0.85%	5.05%
	F	12.76%	7.46%	2.75%	4.27%
	T	11.04%	5.30%	2.15%	3.92%

sizes.

3.6 Summary

The dynamic page in-links and associated anchor text reflect how other pages view destination page changes over time. However, the ever-changing weights on anchor text, as an indicator of the change of anchor text importance, is seldom used for web search, partly because such information is typically not available. In this chapter, we utilize the historical archival copies of web pages provided by the Internet Archive (a public resource) to investigate ways to benefit web search. We propose new methods to quantify anchor text importance, which are motivated by differentiating pages with different in-link creation rate over time and different historical in-link context. Experiments on a crawl from the Stanford WebBase show the ranking performance of our proposed methods has more than 10% improvement over the state-of-the-art method that does not consider historical information.

From this work, we recognize that the existing archival web pages only cover a small portion of the historical web, which causes a large amount of missing anchors (only 2.57% anchors have archival copies in our data set) and thus limits the application of the proposed method. Furthermore, the crawling policies used to collect these archival web page copies might not accurately record the trace of web activities. However, as an initial work, our results revealed that with enough historical information for pages on the web, we can give more accurate estimates about anchor text importance and page in-link importance to improve web search.

Chapter 4

Incorporating Web Freshness into Web Authority Estimation

4.1 Introduction

We presented our approach for incorporating the trends of the creation of page in-links into measuring anchor text importance in Chapter 3. In this chapter, we move to page authority estimation. Page authority is a measure that describes how important a web page is on the web. Because it is necessary to differentiate pages in such a large scale corpus, we consider page authority in addition to its relevance with respect to queries in web search. Much previous work [23, 78, 86] has been studied to estimate page authority based on different assumptions and successfully generalized onto multiple tasks [12, 21, 124]. However, most of these studies accumulated the authority contributions based only on the evidence of links between pages, without considering the temporal aspects concealed in pages and their connections.

Freshness is important to the quality of much in our daily lives, such as flowers and food. The same is also true for web page authority estimation. Pages being fresh tend to be welcome. However, traditional link analysis algorithms such as PageRank [23] estimate page authority by simply accumulating contributions from in-links on a static web link structure, without considering whether pages are still fresh when web users search for them. Freshness of web links is also important to link-based ranking algorithms. The web is widely recognized as one of the networks in which the rich get richer as the networks grow, leading to power law effects [30]. Old pages have more time to attract in-links, but may contain stale information. For example, as of this writing, `http://www.sigir2007.org/` has 902 in-links [128] while `http://www.sigir2010.org/` only has 208. Assuming the same contribution from each in-link, methods like PageRank would render a higher authority score for the earlier version of the SIGIR conference homepage.

Additionally, a branch of research [113, 36] unraveled the fact that the local link structures with sudden changes might indicate link spam. A single web snapshot is unable to detect such changes and further smooth or neutralize the influence automatically.

Motivated by these two points, in this work we propose to estimate web page authority by two separate steps. First, to avoid old pages dominating the authority scores, we keep track of web freshness over time from two perspectives: (1) how fresh the page content is, named *page freshness*; and (2) how much other pages care about the target page, named *in-link freshness*. To achieve this, we mine web authors' maintenance activities on page content, such as the creation and removal of out-links. Each activity is associated with the timestamp at which it occurs. We build temporal profiles for both pages and links. A random walk model is exploited to estimate the two predefined freshness measures. By modeling the web freshness from these two perspectives, we can bias the authority distribution to fresh pages, and so neutralize the unfair preference toward old pages by traditional link analysis ranking algorithms.

Given the web freshness measures we have quantified, the next steps are conducted in two different directions. One of them utilizes the correlation between page freshness and inlink freshness to estimate how influential the update of page content is. We then use such a "influential factor" to enhance our estimated page freshness. We refer to this approach as "correlation based temporal ranking model" (**C-Fresh**). The other direction is based on the random walk models, referred to as the "random walk based temporal ranking model" (**T-Fresh**). T-Fresh incorporates web freshness into time-dependent page authority estimation. It outputs an authority score for each page at every predefined time point. The authority is estimated in an approximated way, partly depending on the link structure and web freshness of nearby snapshots, with the ones at farther time points having smaller influence.

In the remainder of this chapter, we start by introducing how we quantify web freshness and how we incorporate it into a web surfer model to estimate time-dependent web page authorities in Section 4.2. We then present C-Fresh and T-Fresh in Sections 4.3.1 and

	Link activity	**Infl. on p's InF**	**Gain of p's InF**
1	creation of link $l : q \rightarrow p$	$\uparrow\uparrow\uparrow$	3
2	update on link $l : q \rightarrow p$ (changed anchor)	$\uparrow\uparrow$	2
3	update on link $l : q \rightarrow p$ (unchanged anchor)	$\uparrow$	1.5
4	removal of link $l : q \rightarrow p$	$\downarrow\downarrow$	-0.5
	Page activity	**Infl. on q's PF**	**Gain of q's PF**
1	creation of page q	$\uparrow\uparrow\uparrow$	3
2	update on page q	$\uparrow$	1.5
3	removal of page q	$\downarrow\downarrow$	-0.5

Table 4.1: Activities on pages and links and their influence on web freshness. (The link l points from page q to page p. $\uparrow$: positive influence on web freshness. $\downarrow$: negative influence on web freshness. The number of $\uparrow$ or $\downarrow$ indicates the magnitude. We assign the influence and gain of inlink and page freshness based on our intuition in this work, considering our emphasis is to demonstrate the effectiveness of incorporating web freshness on web page authority estimation.)

4.3.2 respectively. We present how we set up experiments in Section 4.4; and show the evaluation results of our proposed ranking algorithms in Section 4.5. We discuss and summarize this work in Section 4.6.

4.2 Representing Web Freshness Over Time

Web freshness reflects how fresh a web page is at a given time point t_i by in-link freshness (InF) and page freshness (PF) (See Figure 4.1 for details.). The reasons we separate these two web freshness measures are: (1) InF and PF depict web freshness from the perspectives of information recommenders and information providers respectively; and (2) it leverages two types of web freshness such that they mutually influence web authority estimation. Given a web page p, we assume that each update on p's parent page q is a direct validation

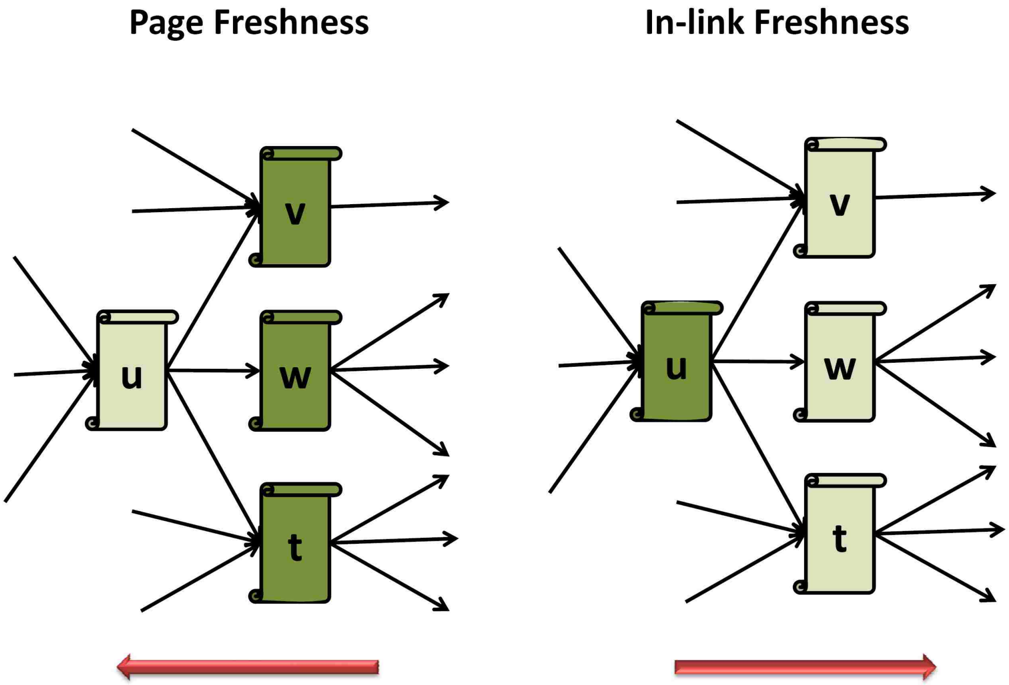


Figure 4.1: The computation of page and in-link freshness. For page freshness (left), the activities associated with page u, v, w and t all influence the page freshness of u. For in-link freshness (right), the activities associated with page u and t mutually influence the in-link freshness of page t. The influence of web maintenance activities on page and in-link freshness propagates through hyperlinks backward and forward respectively.

of the link from q to p, and so the updates on q implies that q pays attention to all of its out-linked pages, including p. Hence, we use InF to represent the attention from p's in-link pages, which is computed from the accumulation of activities on all of p's parent pages up to t_i. Unlike InF, PF represents how fresh p is up to t_i based on the activities on page p itself. We denote the inlink and page freshness of page p at time point t_i as $InF(p)_{t_i}$ and $PF(p)_{t_i}$ respectively.

4.2.1 Building Temporal Page and Link Profiles

In order to compute InF and PF, the first step is to generate temporal page profiles (TPP) and temporal link profiles (TLP). We proposed to use TPP and TLP to record the web authors' activities on the pages and links over time. Given a page p, each item on its TPP records the evidence of p proceeding a type of activity at a specific time point. It is written as a 3-tuple <`page ID, activity type, timestamp`>, where `activity type`∈{creation, update, removal}. Given a link l with its associated anchortext, TLP records the evidence of a type of activity on l at a specific time point. Each item on TLP can similarly be represented as the 3-tuple <`link ID, activity type, timestamp`>, where `activity type`∈{creation, update with unchanged anchor, update with changed anchor, removal}. In this way, each link and page is associated with a series of timestamped activities. Table 4.1 summarizes the influence of these activities on web freshness.

4.2.2 Quantifying Web Freshness

Based on TPP and TLP, we next quantify web freshness, i.e., InF and PF. In order to simplify analysis, we separate the continuous time axis into discrete time points, e.g. $(t_0, t_1, \ldots, t_n, \ldots)$, with a unit time interval Δt between successive time points, i.e., $\Delta t = t_i - t_{i-1}$. Web freshness at any time point t_i is dependent on (1) the web freshness at t_{i-1}, and (2) the activities recorded on TPP and TLP, which occur between t_{i-1} and t_i. When Δt is small enough, it is reasonable to assume that any activities in $[t_{i-1}, t_i]$ occur at t_i. In this way, we map all the web activities onto discrete time points. For web freshness at t_{i-1}, we assume it decays exponentially over time. Thus, $InF(p)_{t_i}$ and $PF(p)_{t_i}$ can be given by:

$$InF(p)_{t_i} = \beta_1 e^{-\beta_2 \Delta t} InF(p)_{t_{i-1}} + \Delta InF(p)|_{t_{i-1}}^{t_i} \tag{4.1}$$

$$PF(p)_{t_i} = \beta_3 e^{-\beta_4 \Delta t} PF(p)_{t_{i-1}} + \Delta PF(p)|_{t_{i-1}}^{t_i} \tag{4.2}$$

where $\Delta PF(p)|_{t_{i-1}}^{t_i}$ and $\Delta InF(p)|_{t_{i-1}}^{t_i}$ are the incremental freshness scores from the activities in $[t_{i-1}, t_i]$, and $\beta_1 e^{-\beta_2 \Delta t}$ is a coefficient that controls the decay of historical web freshness.

In the next step, we compute the incremental in-link freshness $\Delta InF(p)|_{t_{i-1}}^{t_i}$ for the given page p. Since in-link freshness depends on the activities on TLP, we compute $\Delta InF(p)|_{t_{i-1}}^{t_i}$ by accumulating all the activities on p's in-links in $[t_{i-1}, t_i]$. Let $C_j(l)$ be the number of the j^{th} type of link activity on link l in $[t_{i-1}, t_i]$. Let w_j be the unit contribution of the j^{th} type of link activity. The incremental in-link freshness is written as:

$$\Delta InF_0(p)|_{t_{i-1}}^{t_i} = \sum_{l:q \to p} \sum_{j \in LA} w_j C_j(l) \tag{4.3}$$

where LA is the set of link activity types. However, it is not enough to propagate such influence in one step; we additionally propagate in-link activities iteratively, leading to smoother in-link freshness scores. Let $\Delta InF_0(p)|_{t_{i-1}}^{t_i}$ in Equation 4.3 be an initial score. In each iteration, every page receives in-link freshness scores from its parent pages, and also holds its initial score. The process converges and produces a score for every page determined by both its parents' scores and its own in-link activities [110]. Thus, the incremental in-link freshness is given by:

$$\Delta InF(p)|_{t_{i-1}}^{t_i} = \lambda_{InF}\Delta InF_0(p)|_{t_{i-1}}^{t_i} + (1-\lambda_{InF})\sum_{l:q\to p} m_{qp}\Delta InF(q)|_{t_{i-1}}^{t_i} \tag{4.4}$$

where m_{qp} is the weight on the link from q to p. Equation 4.4 is actually the personalized PageRank (PPR) [67]. We use one-step transition probability from q to p based on link structure to represent m_{qp}, where $\sum m_{q*} = 1$ if q has at least one out-link.

We next compute the incremental page freshness $\Delta PF(p)|_{t_{i-1}}^{t_i}$. Similar to $\Delta InF(p)|_{t_{i-1}}^{t_i}$, we argue that how fresh one page is depends on both the page itself and its out-linked pages, since the out-linked pages are in some sense extensions of the current page. We thus propagate page freshness backward through links. In each iteration, every page receives page freshness scores from its out-linked pages, and also holds its initial score. This process converges finally and generates a page freshness score on every page. Let $C'_j(p)$ be the number of the j^{th} type of page activity on p in time period $[t_{i-1}, t_i]$. Let w'_j be the unit contribution of the j^{th} type of page activity. The initial incremental page freshness score $PF_0(p)|_{t_{i-1}}^{t_i}$ is defined as:

$$\Delta PF_0(p)|_{t_{i-1}}^{t_i} = \sum_{j\in PA} w'_j C'_j(p) \tag{4.5}$$

where PA is the set of page activity types. The incremental page freshness is given by:

$$\Delta PF(q)|_{t_{i-1}}^{t_i} = \lambda_{PF}\Delta PF_0(q)|_{t_{i-1}}^{t_i} + (1-\lambda_{PF})\sum_{l:q\to p} m'_{qp}\Delta PF(p)|_{t_{i-1}}^{t_i} \tag{4.6}$$

where m'_{qp} is the weight on the link from q to p. We use the inverted one-step transition probability to represent m'_{qp}, where $\sum m'_{*p} = 1$ if page p has at least one in-link.

Once $\Delta InF(p)|_{t_{i-1}}^{t_i}$ and $\Delta PF(p)|_{t_{i-1}}^{t_i}$ are computed, we compute $InF(p)_{t_i}$ and $PF(p)_{t_i}$ by Equation 4.1 and 4.2.

4.3 Temporal Ranking Models

Given the inlink freshness and page freshness of every page at individual time points we presented in Section 4.2, we next introduce these two types of freshness scores to enhance web authority estimation, using correlation and random walk temporal ranking models.

4.3.1 Correlation Based Temporal Ranking Model (C-Fresh)

C-Fresh quantifies the temporal freshness correlation between pages and their in-links. Its underlying assumption is that the consistency between the changes of page content and page in-coming links reflects the impacts of page change. To do this, we exploit the method by Chien and Immorlica [32], in which the authors measure query semantic similarity by using temporal correlation. Given a page p, its page and in-link freshness are denoted as $(PF_{t_c}(p), PF_{t_{c+1}}(p), \ldots, PF_{t_r}(p))$ and $(InF_{t_c}(p), InF_{t_{c+1}}(p), \ldots, InF_{t_r}(p))$ covering p's life span. The temporal freshness correlation (TFC) between page p and its in-links is given by:

$$TFC(p) = \frac{1}{n}\sum_{t=t_c}^{t_r}\Big(\frac{PF_t(p) - \overline{PF(p)}}{\sigma_{PF}(p)}\Big)\Big(\frac{InF_t(p) - \overline{InF(p)}}{\sigma_{InF}(p)}\Big)$$

where $\sigma_{PF}(p)$ and $\sigma_{InF}(p)$ are the standard deviations of $PF(p)$ and $InF(p)$, respectively.

Once we calculate the temporal freshness correlation for every page ($t_r - t_c \geq 2\Delta t$), we next combine it with page freshness by ranks, rather than scores. Given a time point of interest t_i, the combined page freshness rank of document d is written as:

$$Rank_{combined}(d) = (1-\beta)Rank_{PF_{t_i}}(d) + \beta Rank_{TFC}(d) \tag{4.7}$$

where $\beta = \frac{a-1}{n-1+a-1}$, and n is the total number of time points, and a is the number of time points on which p exists. As a increases, $TFC(d)$ becomes more stable, and therefore we emphasize its contribution in the combined page freshness estimation $Rank_{combined}(d)$ (Equation 4.7). We next use this combined page freshness score to represent web page authority.

4.3.2 Random Walk Based Temporal Ranking Model (T-Fresh)

T-Fresh follows proximity-based authority propagation rules. It outputs an authority score for each page at every predefined time point. The authority is estimated in an approximated way, partly depending on the link structure and web freshness of nearby snapshots, with the ones at farther time points having smaller influence.

We start by describing a "temporal random surfer model", which motivates our method T-Fresh. The "temporal random surfer model" is similar to the "random surfer model", which explains PageRank [23]. However, our surfer model differs from the traditional random surfer model in two aspects. First, the way that the web surfer chooses the pages on which snapshot to reach depends on the time point of her current snapshot. Second, the web surfer prefers fresh web resources. Figure 4.2 depicts one simple example of how the surfer behaves on an archival web of four snapshots.

Consider a web surfer wandering on an archival web corpus, which includes multiple web snapshots collected at different time points (t_0,t_1, ...,t_n). For every move, the surfer takes the following steps. First, she can choose either to follow one of the out-linked pages or to randomly jump to any page at the same time point. However, unlike PageRank in which a web surfer has equal probabilities to follow out-going links, the preference of our surfer choosing out-going links correlates to the page freshness of out-linked pages. Consider the example in Figure 4.2. Suppose the surfer is currently on page A at t_2. She follows the link to B at t_2 (solid link) with probability $(1-d)F_{t_2}(B, A)$, where $F_{t_2}(B, A)$ is a function which depends on the page freshness of all A's out-linked pages at t_2 and $\sum_{P:A\to P} F_{t_2}(P, A) = 1$. The probability that the surfer randomly jumps to any page at t_2, such as B, is d/N_{t_2}, where N_{t_2} is the total number of pages at t_2, and d is a constant 0.15.

After the surfer reaches the page chosen in the first step, she next selects the specific snapshot of that page to jump based on her locality, which correlates to the time difference between the current snapshot and the snapshot that the surfer will reach next time. This process actually propagates authority among snapshots and uses the link structure at one time point to influence the authority computation at other time points. The propagation decays with time difference between snapshots. In the example shown in Figure 4.2, t_1 to t_4 represent four successive snapshots, and the same pages on different snapshots represent their states at different time points. Based on such a archival web, suppose the surfer reaches B at t_2 after the first step, she can jump to B at any time point as long as it exists (following dash bi-directed links), i.e., t_2, t_1, and t_0. Specifically, the probability that she

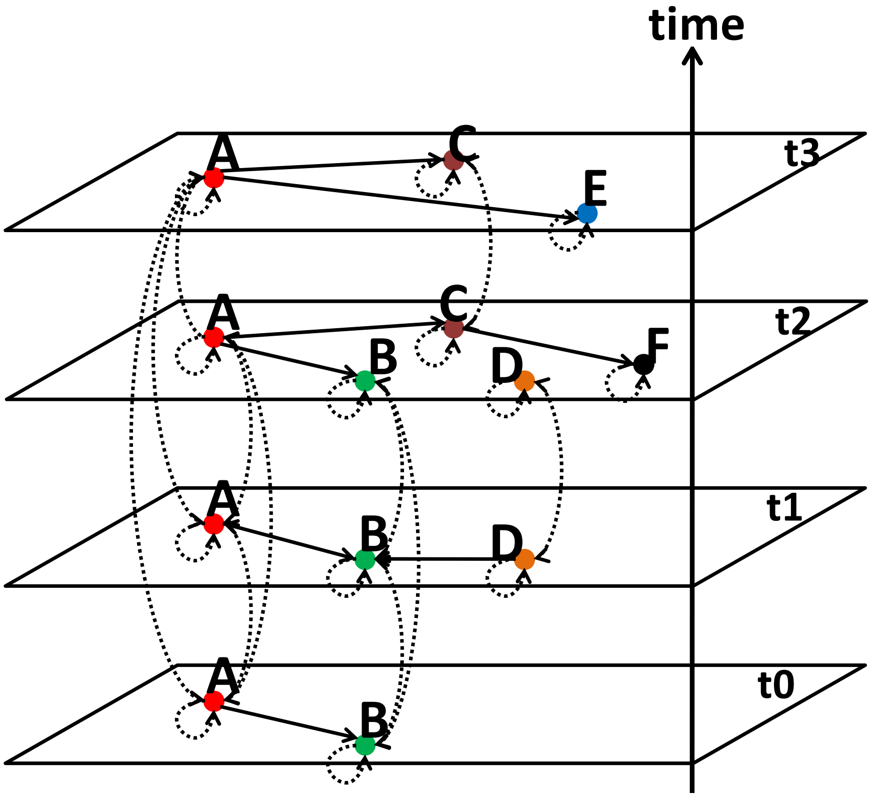


Figure 4.2: The process of T-Fresh. Each node represents one web page.

jumps to B at t_1 is written as $P_{t_1|t_2}(B)$, which depends on the time difference between t_1 and t_2.

Once the surfer reaches the page at the chosen time point, e.g., page B at t_1, she browses it with the mean stay time $\mu_{t_1}(B)$, which correlates B's in-link freshness at t_1 before the next move.

In this way, the surfer's behavior on the archival web can be separated as (1) moving from one page to another (this can proceed either within the same snapshot or between two different snapshots); and (2) staying on a page. It leads to a semi-Markov process [110] for page authority estimation.

Definition 1 *A semi-Markov process is defined as a process that can be in any one of N states $1, 2, \ldots, N$, and each time it enters a state i it remains there for a random amount of time having mean μ_i, and then makes a transition into state j with probability P_{ij}.*

Suppose the time that the process spends on each state is a fixed constant, the semi-Markov process leads to a Markov chain. Assuming all states in such a Markov chain communicate with each other, the process can generate a stationary probability π_i for any state i. The long-run proportion of time that the original semi-Markov process is in state i is given by:

$$A(i) = \frac{\pi_i \mu_i}{\sum_{j=1}^{N} \pi_j \mu_j}, i = 1, 2, \ldots, N \tag{4.8}$$

This solution divides the time-dependent page authority estimation into (1) computing the stationary probability that a surfer reaches every page in the archival corpus; and (2) computing the mean of a surfer staying on every page.

Estimating Stationary Probability

We now introduce the computation of probability π_{p,t_i} that a web surfer enters a page p at the snapshot t_i. In the first step of each move, the surfer reaches page p at any time point t_j by: (1) following p's in-link at t_j to reach p; (2) jumping from any page at t_j to p at t_j.

$$P_{t_j}(Follow|q) = (1 - d), \qquad P_{t_j}(p|q, Follow) = F_{t_j}(p, q) \tag{4.9}$$

$$P_{t_j}(Jump|q) = d, \qquad\qquad P_{t_j}(p|q, Jump) = 1/N_{t_j} \tag{4.10}$$

where d is 0.15 by default. $F_{t_j}(p, q)$ is the web surfer's preference on following out-linked pages. Intuitively, a fresh web resource is more likely to attract a surfer's attention. We define $F_{t_j}(p, q)$ as:

$$F_{t_j}(p, q) = \frac{PF_{t_j}(p)}{\sum_{p':q \rightarrow p'|_{t_j}} PF_{t_j}(p')} \tag{4.11}$$

In the second step of each move, the surfer reaches page p at t_i from page p at t_j is given by:

$$P_{t_i|t_j}(p) = \frac{w(t_i, t_j)}{\sum_{q \in V_i, q \in V_j} w(t_i, t_j)} \tag{4.12}$$

where V_i and V_j are the sets of pages at time point t_i and t_j respectively, and $w(t_i, t_j)$ is the weight that represents the influence between the snapshots at t_i and t_j. Motivated by previous work [47, 77, 87, 100] which used proximity-based methods, we utilize 6 kernel functions to model the authority propagation between snapshots: gaussian kernel (equation 4.13), triangle kernel (equation 4.14), cosine kernel (equation 4.15), circle kernel (equation 4.16), passage kernel (equation 4.17) and PageRank kernel (equation 4.18). We formally define them as follows.

$$w_1(t_i, t_j) = \exp\left[-\frac{(t_i - t_j)^2}{2|T|^2}\right] \tag{4.13}$$

$$w_2(t_i, t_j) = 1 - \frac{|t_i - t_j|}{|T|} \tag{4.14}$$

$$w_3(t_i, t_j) = \frac{1}{2}\left[1 + \cos\left(\frac{|t_i - t_j|\pi}{|T|}\right)\right] \tag{4.15}$$

$$w_4(t_i, t_j) = \sqrt{1 - \left(\frac{|t_i - t_j|}{|T|}\right)^2} \tag{4.16}$$

$$w_5(t_i, t_j) = 1 \tag{4.17}$$

$$w_6(t_i, t_j) = \begin{cases} 0.85 & t_i = t_j \\ \frac{0.15}{|T-1|} & t_i \neq t_j \end{cases} \tag{4.18}$$

where $|T|$ is the window size of one step authority propagation between snapshots. Except for Equation 4.13, all the other kernels require $|t_i - t_j| < |T|$, that is, the one step authority propagation proceeds only within the window with a specified size. Larger $|T|$ results in more choices for the web surfer at each move between snapshots, while smaller $|T|$ leads to direct influence mainly from nearby time points. In this work we set $|T|$ to the total number of snapshots involved in authority propagation by default.

Combining the analysis above, the probability that a web surfer reaches page p at snapshot t_i can be written as:

$$\begin{aligned} \pi_{p,i} &= \sum_{t_j \in T_i} P_{t_i|t_j}(p) \sum_{q:q \rightarrow p|_{t_j}} P_{t_j}(Follow|q) P_{t_j}(p|q, Follow) \\ &\quad + \sum_{t_j \in T_i} P_{t_i|t_j}(p) \sum_{q|_{t_j}} P_{t_j}(Jump|q) P_{t_j}(p|q, Jump) \\ &= \sum_{t_j \in T_i} P_{t_i|t_j}(p) \times \left[(1-d) \sum_{q:q \rightarrow p|_{t_j}} F'_{t_j}(p,q) \pi_{q,j} + d \sum_{q|_{t_j}} \frac{\pi_{q,j}}{N_{t_j}} \right] \end{aligned} \tag{4.19}$$

where T_i is the set of snapshots which can directly distribute authority to t_i within one step. Based on the surfer's behavior, this Markov process guarantees all the states to communicate with each other, leading to a transition matrix that is irreducible and aperiodic [110]. As a result, it converges and generates a stationary probability on every page existing in any snapshot.

Estimating Staying Time

Pages with more in-link activity are likely to attract a surfer to spend time in browsing it. We assume the web surfer prefers fresh web resources, and so the mean time ($\mu_{p,i}$) of the surfer staying on page p at t_i can be proportional to p's web freshness at t_i. As discussed in Section 4.3.2, the web surfer prefers pages with high page freshness when choosing among out-going links; we use in-link freshness to model the time of a surfer staying on a web page. In this way, the pages with both high in-link freshness and page freshness are more likely to be given high authority scores. Specifically, we utilize a sliding window and compute p's weighted in-link freshness centroid within it as the estimation of $\mu_{p,i}$, which is formally given by

$$\mu_{p,i} = k \sum_{t_j \in T'_{t_i}} w'(t_i, t_j) InF(p)_{t_j} \tag{4.20}$$

where T'_{t_i} is the set of snapshots included in the sliding window centered on t_i, and $\sum_{t_j \in T'_{t_i}} w'(t_i, t_j) = 1$. In this work we evaluate one special case, in which $w'(t_i, t_j) = \frac{1}{|T'_{t_i}|}$ for any $t_j \in T'_{t_i}$. In this way, the authority score $A(i)$ in Equation 4.8 is determined by both $\pi_{p,i}$ in Equation 4.19 and $\mu_{p,i}$ in Equation 4.20.

4.4 Experimental Setup

4.4.1 Data set and relevance judgment

Most standard data sets such as those used at TREC [97] usually only contain one snapshot of a web corpus, and so are not suitable to show the effectiveness of ranking models utilizing temporal information. To evaluate our proposed method, we use a corpus of archival web pages in the `.ie` domain collected by Internet Archive [70] from January 2000 to December 2007. This corpus contains 158 million unique web pages, and approximately 12 billion temporal links. To avoid the influence of transient web pages, we extract one web graph for each month from the sub-collection of pages for which we have at least 5 crawled copies. These graphs comprises a collection of 3.8M unique pages and 435M temporal links in total.

For ranking evaluation, we choose `April 2007` as our time period of interest since Internet Archive changed crawling policies right after `April 2007`. Ninety queries are selected from a set of sources, including those frequently used by previous researchers, and popular queries from Google Trends [65] (See Table 4.2 for details.). For each query, we have an average of 84.6 URLs judged by at least one worker of Amazon's Mechanical Turk [7]. When human editors judge each <query,URL> pair, they are required to give a score based on (1) how relevant the page is to the query; and (2) how fresh the page would be as a result for the requested time period. The relevance score is selected from among highly relevant, relevant, borderline, not relevant and not related, which is translated to an integer gain from 4 to 0. A page with a score higher than 2.5 is marked as relevant. Similar to the relevance judgement, the freshness score is selected from *very fresh, fresh, borderline, stale,* and *very stale*, which we translate into an integer scaled from 4 to 0. A page with a score higher than 2.5 is marked as fresh. All human editors were asked to give the confidence of their provided judgments, in the selection of high, medium and low. Judgements with low confidence are not included in the ranking evaluation[1]. A random sample with 76 <query, URL> pairs judged by 3 editors show that the average standard deviations of relevance and freshness judgements are 0.88 and 1.02 respectively.

4.4.2 Ranking Evaluation

We evaluate the ranking quality of our approach on both relevance and freshness over the Normalized Discounted Cumulative Gain (NDCG) [71] metric. It penalizes highly relevant or fresh documents appearing at lower positions. Precision@k is also utilized to measure ranking quality, which calculates the number of relevant or fresh documents within the top k results across all queries.

To show the effectiveness of C-Fresh and T-Fresh, their outputs are combined with Okapi BM2500 [107] linearly by ranks for ranking evaluation, defined as:

$$(1-\gamma)rank_{authority}(p) + \gamma rank_{BM}(p)$$

The parameters used in Okapi BM2500 are the same as Cai et al. [28].

[1] We will report the detailed judgment guidance in Chapter 5.4.

2007 cricket world cup	amazon	american idol
angelina jolie	arsenal	barbie
baseball	bbc sports	best buy
bill clinton	bird flu	black friday
bmw ireland	britney spears	car zone
casino	college football	continental airlines
craigslist	da vinci code	democratic national convention
desperate housewives	disneyland	dublin bus
earthquake	halloween costumes	expedia
facebook	firefox	fox news
george w bush	groundhog day	hannah montana
harry potter	hello kitty	hip hop
housing bubble	hurricane	iphone
iraq war	irish independent	jennifer lopez
kill bill	liverpool fc	lord of the rings
lunar eclipse	map of ireland	medicine
meteor	michael jackson	mobile games
monet	mtv	myspace
national weather service	nba	netflix
new york times	nfl	obama
olympics schedule	oscar nominations	perl programming
pink floyd	playstation	poker
porsche	presidential polls	prince charles
prison break	real madrid	reuters
richard hammond	rte tv	skype
spring break	staples	starbucks
summer olympics	super bowl	terrorism
thanksgiving	tom cruise	tsunami
verizon	wedding dresses	whitney houston
wikipedia	world cup	youtube

Table 4.2: Set of ninety temporal queries used for relevance evaluation in IA data set.

4.4.3 Web Activity Detection

While accurate web maintenance activities are recorded on Web servers' logs, we infer such activities from the comparison between successive web snapshots in this work since we are not able to access the logs of most servers on the web. Specifically, we assume that each page was created at the time at which it was first crawled, and each link was created when it was first found. Although some pages can automatically change a portion of its content in every crawl, we suppose one page has an update when its content has any difference from the previous version, or its meta-data can show the last-modified time is after the crawling time of the previous one. To identify the link update, we assume that once a page has an update, all its out-links are considered to be updated. We admit that the perfect quantification on link update activity may depend on a variety of factors, including the distance to page blocks being changed, the burstiness of page editing frequency over time, and so on. We leave the sensitivity of web activity detection accuracy on ranking performance to future work. We also assume that a page disappears when its returned HTTP response code is 4xx or 5xx. While the gain associated with each type of link and page activity can influence the ranking performance, as a preliminary study, we define these gains in Table 4.1, and leave the sensitivity of ranking performance with respect to gains on web activity to future work.

4.5 Experimental Results

In this section, we report the results of our ranking evaluation and compare C-Fresh and T-Fresh with representative link-based algorithms. Our purpose is to demonstrate the superiority of C-Fresh and T-Fresh. To do this, we ask the following research questions.

- Is there any relationship between InF and PF? Will they help predict future activity, inferring web freshness at future time points? Will the propagation of activity influence helps better measure InF and PF respectively? To understand such questions may help improve future temporal ranking models that also expect to incorporate web freshness.

- **C-Fresh**: Does InF help improve search quality? Does the correlation between InF and PF further boost search quality? If so, is such influence proportional to the time span based on which the correlation is computed.

- **T-Fresh**: How does T-Fresh outperform the representative link-based ranking algorithms that incorporate the temporal information? To what extent does T-Fresh outperform those algorithms? Which components within T-Fresh result in its superiority? To explore such questions helps better understand the ways in which T-Fresh works.

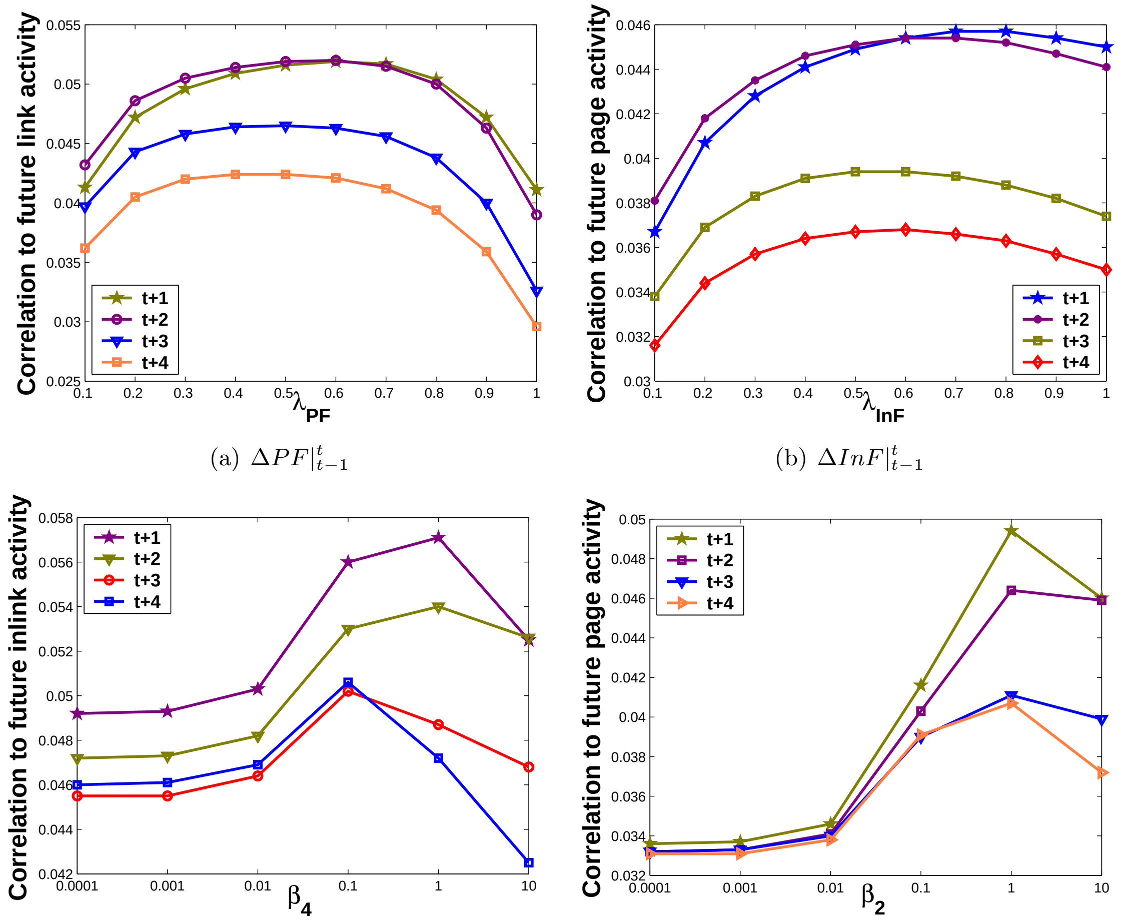
We will see that comparable experimental results—by incorporating web freshness, both C-Fresh and T-Fresh can achieve more relevant and fresh search results.

4.5.1 Correlation of InF and PF

We focus on the first research question in this section. As introduced in Section 4.2.2, each page in the temporal graph associates with InF and PF. A reasonable criteria for the good estimation of InF and PF would be their potential capability of predicting future web activities even though the correlation between them would be rather small. To better understand it, we compute the average correlation between web freshness scores at t and web activities at future time points, i.e., $t+1$, $t+2$, etc., given by Equation 4.3 and 4.5.

From Figure 4.3 (a), $\Delta PF|_{t-1}^{t}$ and future in-link activities show positive correlation, with the strength inversely proportional to the time difference between the incremental page freshness and future in-link activities. In most cases, the correlation is the greatest when λ_{PF} and λ_{InF} are 0.6. It indicates that pages derive freshness scores from both the activities on themselves and their neighbor pages via propagation. The correlations between $\Delta InF|_{t-1}^{t}$ and future page activities show similar trends (Table 4.3 (b)). One may notice that the average correlation between $\Delta PF|_{t-1}^{t}$ and in-link activities at $t+1$ is 0.0519, which is higher than that between $\Delta InF|_{t-1}^{t}$ and page activities at $t+1$ by over 13.5%. One interpretation is that a page with very fresh content tends to attract new in-links or existing in-links to validate in next time periods. From Table 4.3 (c) and (d), the cumulative web freshness scores can show stronger correlation to future web activities, varying with the decay parameter β_2 and β_4 given $\beta_1 = \beta_3 = 1$ constantly. For both PF_t and InF_t, the correlations are the highest when β_2 and β_4 are 1 in most cases.

In summary, our observations are as follows.



(a) $\Delta PF|_{t-1}^{t}$

(b) $\Delta InF|_{t-1}^{t}$

(c) PF_t

(d) InF_t

Figure 4.3: Correlation between web freshness and future web activities.

- The correlation between page and future in-link activities is stronger than the one between in-link and future page activities.

- When incorporating the activities associated with neighbor pages, the correlations between page (in-link) and future in-link (page) activities are stronger than the correlations without considering the influence from neighbor pages.

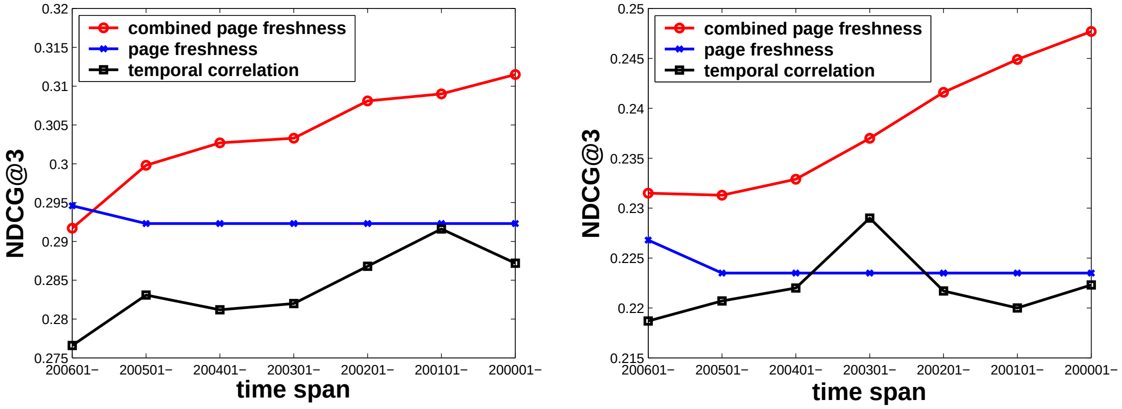
Relevance				
Method	P@10	NDCG@3	NDCG@5	NDCG@10
Okapi BM2500	0.4695	0.2478	0.2740	0.3344
PageRank	0.4894	0.2589	0.2840	0.3457
200601-200704	**0.5021**†	0.2917††	0.3152††	**0.3675**††
200401-200704	0.4893	0.3027††	0.3201††	0.3657††
200201-200704	0.5002†	0.3081††	0.3157††	0.3642††
200001-200704	0.4986†	**0.3115**††	**0.3211**††	0.3647††
Freshness				
Method	P@10	NDCG@3	NDCG@5	NDCG@10
Okapi BM2500	0.3138	0.2137	0.2379	0.2805
PageRank	0.3325	0.1946	0.2345	0.2838
200601-200704	0.3288†	0.2315††	0.2490†	0.2979†
200401-200704	0.3342†	0.2329††	0.2552††	0.2988†
200201-200704	0.3361†	0.2416††	0.2565††	0.3027††
200001-200704	**0.3374**†	**0.2477**††	**0.2617**††	**0.3028**††

Table 4.3: Ranking performance comparison. A † means the performance improvement is statistically significant (p-value<0.1) over Okapi BM2500. Performance improvement with p-value<0.05 is marked as ††.

- When aggregating past activities, the correlations between page (in-link) and future in-link (page) activities are stronger than the correlations without considering the aggregation from past activities.

4.5.2 C-Fresh: Ranking Evaluation

We now focus on the ranking evaluation of C-Fresh. We set $\lambda_{PF} = \lambda_{InF} = 1$ and $\beta_1 = \beta_2 = \beta_3 = \beta_4 = 1$ for computing InF and PF respectively without the loss of generality. Table 4.3 lists the ranking performance comparison varying the time span involved in the combined page freshness computation. We use Okapi BM2500 and PageRank as



(a) relevance

(b) freshness

Figure 4.4: Ranking performance on metric NDCG@3 while varying the time span involved in page freshness calculation.

our baselines. For relevance, except for NDCG@3, the correlation between ranking performance and the time span is not consistent. Unlike relevance, freshness performance consistently improves with the increase of time span used in the combined page freshness computation. This suggests temporal freshness correlation calculated from long-term web freshness measures can benefit more from accurate page freshness estimation. Figure 4.4 shows the performance on NDCG@3 with the variance of the time span for both relevance and freshness. We observe that (1) the ranking performance of page freshness first decreases, and then keeps nearly constant with the increase in time span, indicating the page activities within the past 1-2 years influence page freshness estimation the most; (2) the ranking performance of temporal freshness correlation shows unstable trends with variance of time span; and (3) the combined page freshness shows promising performance, and demonstrates its superiority over either page freshness or TFC.

Notation of T-Fresh variants: T-Fresh(kernel, window, snapshot)	
kernel	The kernel controlling authority propagation among different web snapshots, where $kernel \in \{1, 2, 3, 4, 5, 6\}$
window	The window size used in calculating average in-link freshness for estimating staying time, where $window \in N$
snapshot	The number of months spanned over the temporal graph where $1 \leq snapshot \leq 88$ (from Jan. 2000 to Apr. 2007)

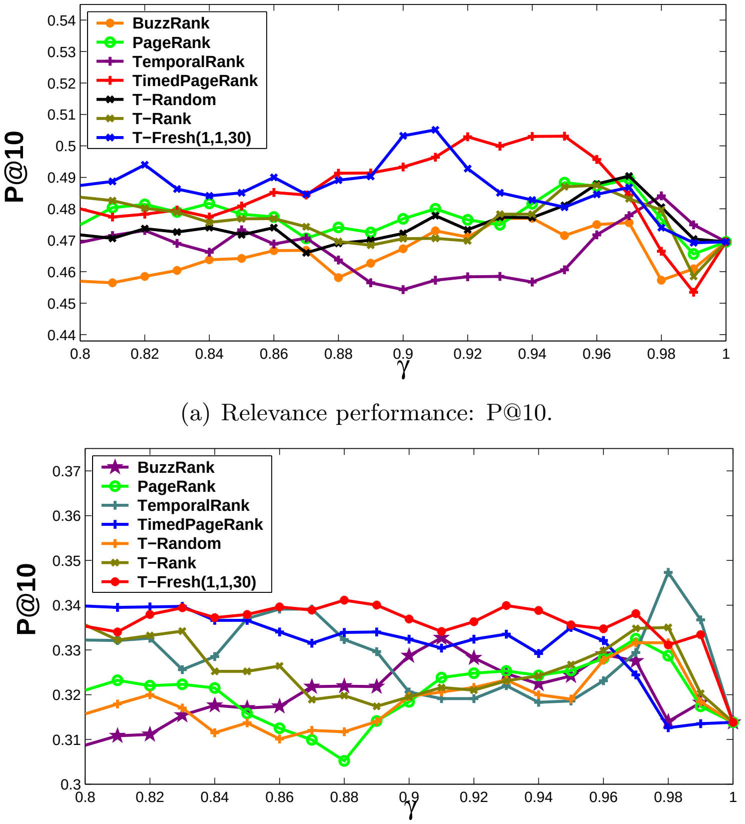
Table 4.4: Notation of T-Fresh variants.

4.5.3 T-Fresh: Ranking Evaluation

We focus on the ranking evaluation of T-Fresh, aiming to answer the third research question. We set $\lambda_{PF} = \lambda_{InF} = 0.6$ and $\beta_1 = \beta_2 = \beta_3 = \beta_4 = 1$ in the ranking evaluation of this section. We compare with PageRank [23] (the baseline) and several representative link-based ranking algorithms which incorporate temporal information, including TimedPageRank [133], T-Rank [17], BuzzRank [14], TemporalRank [129], and T-Random [85]. The variants of T-Fresh are summarized in Table 4.4.

Ranking Performance

Figure 4.5 demonstrates the ranking performance in terms of relevance and freshness on metric P@10 over all the compared algorithms, under the variance of combination parameter γ from 0.8 to 1. The variant of T-Fresh we choose to compare is T-Fresh(1,1,30). For relevance evaluation, PageRank achieves its highest P@10 at 0.4894 when γ is 0.97. T-Fresh performs the best among all the algorithms, achieving its highest P@10 at 0.5051 when γ is 0.91, which is over PageRank by 3.2%. The TimedPageRank places the second on



(a) Relevance performance: P@10.

(b) Freshness performance: P@10.

Figure 4.5: Sensitivity of P@10 with respect to combination parameter γ.

metric P@10, which reaches 0.5031 when γ is 0.92. For each method, we set the combination parameter γ such that it achieves the best performance on P@10 for comparison. The ranking performance over all metrics is reported in Table 4.5. T-Fresh performs the best among all the algorithms over all the metrics. Specifically, it outperforms PageRank over 24.7%, 17.8% and 7.8%, in terms of NDCG@3, NDCG@5 and NDCG@10. Single-tailed student t-tests at a confidence level of 95% demonstrate the improvements are statistically significant over PageRank on NDCG@3, NDCG@5 and NDCG@10, with p-values 0.0001, 0.0001, 0.0016 respectively.

For freshness evaluation, Figure 4.5 (b) shows ranking performance on metric P@10, varying with the combination parameter γ. T-Fresh demonstrates a stable trend on P@10, which exceeds PageRank on all the experimental data points. Unlike relevance evaluation in which improvements of other temporal link-based algorithms are not obvious, more methods can produce fresher search results than PageRank. One reason is that these temporal link-based algorithms incorporate diverse temporal factors, which favor fresh web pages. T-Fresh reaches its best P@10 at 0.3412 when γ is 0.88, which is only inferior to TemporalRank with its highest P@10 at 0.3473 when γ is 0.98. PageRank has its best P@10 at 0.3325 when γ is 0.97. With individual best combination parameter γ on P@10, we compare all the ranking algorithms over other metrics in Table 4.5. T-Fresh outperforms PageRank in terms of NDCG@3, NDCG@5 and NDCG@10 over 23.8%, 13.5% and 8.3%, with p-values 0.0090, 0.0260 and 0.0263 respectively. One observation is the performance of PageRank on metric NDCG@3 is extremely low while its performance on NDCG@5 and NDCG@10 are not so bad. We infer that stale web pages can achieve high authority scores by PageRank, and so dominate top positions in search results.

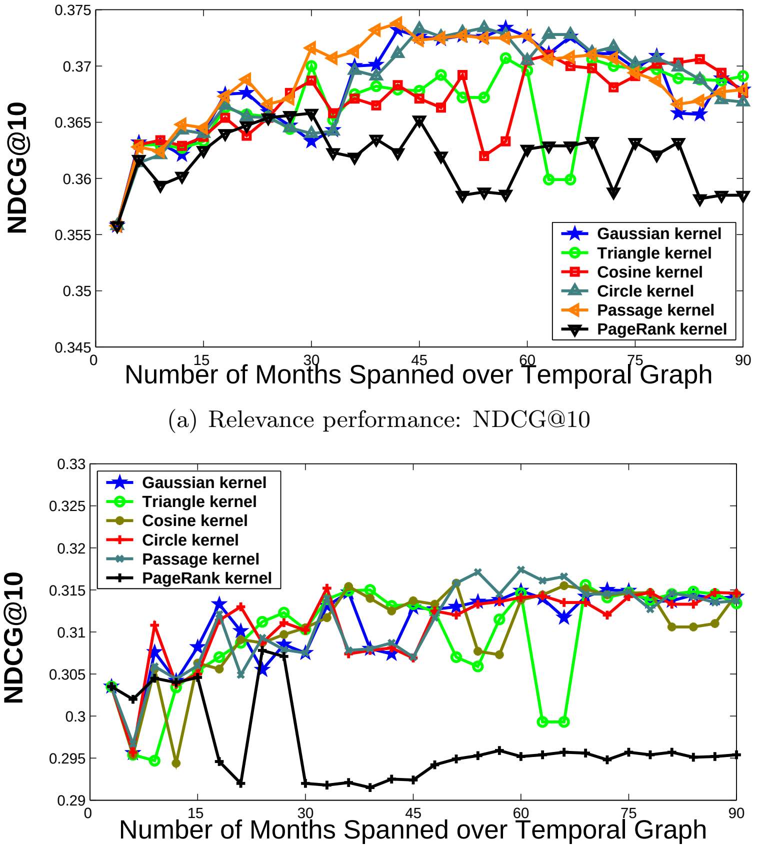
Deeper Analysis

We study the effects of propagation kernels and window sizes used in the staying time estimation on ranking performance in this section.

Figure 4.6 (a) and (b) show the best ranking performance of T-Fresh(*,1,*) on metric

Relevance				
Method	P@10	NDCG@3	NDCG@5	NDCG@10
BM25	0.4695	0.2478	0.2740	0.3344
PageRank	0.4894	0.2589	0.2840	0.3457
BuzzRank	0.4770	0.2770	0.2980	0.3460
TemporalRank	0.4841	0.2706	0.2875	0.3524
TimedPageRank	0.5031	0.2830	0.3063	0.3587
T-Random	0.4904	0.2690	0.2877	0.3495
T-rank	0.4875	0.2669	0.2870	0.3496
T-Fresh(1,1,30)	**0.5051**	**0.3229**	**0.3347**	**0.3729**
Freshness				
Method	P@10	NDCG@3	NDCG@5	NDCG@10
BM25	0.3138	0.2137	0.2379	0.2805
PageRank	0.3325	0.1946	0.2345	0.2838
BuzzRank	0.3327	0.2043	0.2234	0.2797
TemporalRank	**0.3473**	0.2312	0.2510	0.2992
TimedPageRank	0.3398	**0.2443**	0.2514	0.2972
T-Random	0.3316	0.2054	0.2403	0.2879
T-rank	0.3356	0.2269	0.2498	0.2950
T-Fresh(1,1,30)	0.3412	0.2411	**0.2662**	**0.3076**

Table 4.5: Performance Comparison.

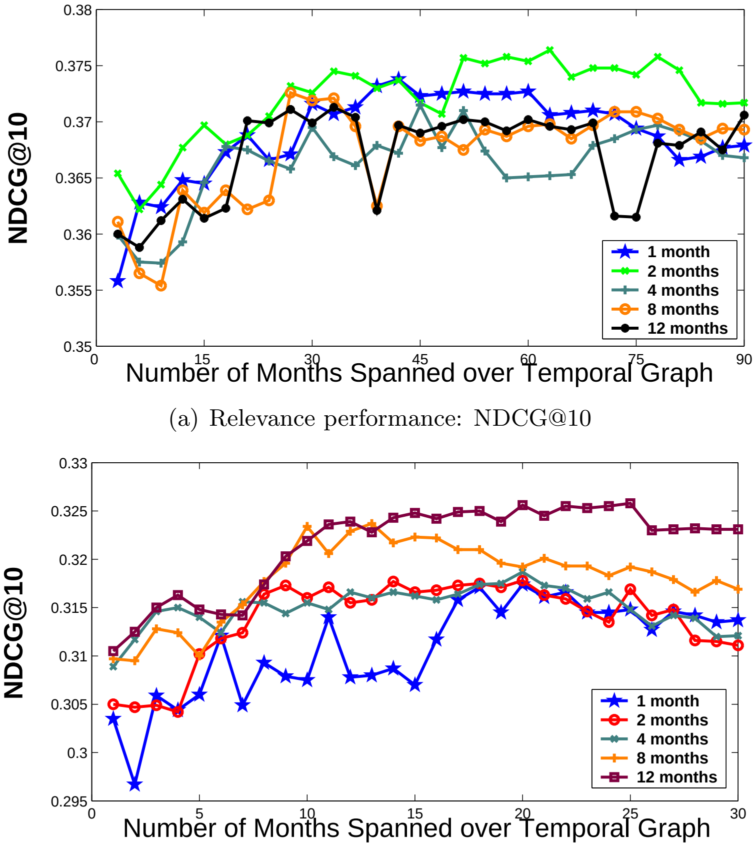


(a) Relevance performance: NDCG@10

(b) Freshness performance: NDCG@10

Figure 4.6: T-Fresh(*,1,*): Sensitivity of NDCG@10 with respect to kernel for authority propagation.

NDCG@10 for relevance and freshness. For most kernels, the relevance performance improves with the time span of the temporal graph, and reaches the highest in $[30, 60]$, i.e., from 2.5 to 5 years. The improvements upon using single snapshot are 4.9%, 4.1%, 4.2%, 4.9%, 5.0% and 2.8% for gaussian, triangle, cosine, circle, passage and PageRank kernels respectively. The passage kernel achieves a stable and best overall performance, followed



(a) Relevance performance: NDCG@10

(b) Freshness performance: NDCG@10

Figure 4.7: T-Fresh(5,*,*): Sensitivity of NDCG@10 with respect to window size used in the stay time estimation.

by gaussian and circle kernels. Results from triangle and cosine kernels show larger fluctuations over time span of the temporal graph. Combining with the kernel expressions defined in Equations 4.13-4.18, we conclude that the ranking improvements on relevance benefit from appropriate emphasis on authority propagation between far away snapshots.

The ranking performance on freshness shows similar trends to relevance, though the variance is typically larger. Except PageRank kernel, all the other ones can achieve their highest performance in the time interval $[30, 60]$. Passage kernel gets the best performance 0.3171 on metric NDCG@10 by outperforming the baseline (using a single snapshot) by 4.5%. One observation is that the performance of PageRank kernel suddenly falls down to around 0.295 when the graph time span is beyond 30 months. One possible reason is that the authority propagation among any distinct web snapshots become very weak in PageRank kernel when the graph time span is large enough, and so historical link structures only have tiny influence on page authority estimation at the current time point. In addition, the freshness performance tends to stablize when the graph time span is over 70 months, which indicates temporal web graphs with long time span render more stable ranking performance on freshness, and it reflects the long-term freshness of web resources.

Figure 4.7 (a) and (b) show the best ranking performance of T-Fresh(5,*,*) on metric NDCG@10 in terms of relevance and freshness. For relevance evaluation, our results demonstrate: (1) To use the average in-link freshness on several adjacent time points is better than to use it at a single time point when estimating staying time. We infer that average in-link freshness can render a good estimation about how active the page in-links are during a time period; (2) It does harm to ranking performance on relevance when the window size is too large; (3) Large window sizes result in large variance of ranking performance when varying the number of snapshots in the temporal web graph; (4) The ranking performance improves with the increase of graph time span in general for all the window sizes. For freshness evaluation, a clear trend in Figure 4.7 (b) shows that a larger window size used in staying time estimation helps generate fresher search results with smaller deviation.

4.6 Summary

Dynamic web resources reflect how active web pages are over time. From the perspectives of in-links and the page itself, we quantify web freshness from web creators' activities. We argue that web freshness is an important attribute of web resources, which can benefit a series of time-sensitive applications, including archival search, news ranking, twitter message recommendation, tag recommendation and so on.

In this work we propose two temporal ranking models, i.e., C-Fresh and T-Fresh, both of which draw from the web freshness inferred from web page and link maintenance activities. C-Fresh incorporates a temporal freshness correlation (TFC) component in quantifying page freshness. Experiments show that by using TFC, we can achieve a good estimate of how up-to-date the page tends to be, which is helpful to improve search quality in terms of both result freshness and relevance. Such benefits are proportional to the time span based on which the page freshness and the correlation between inlink and page freshness are computed.

T-Fresh is a temporal web link-based ranking algorithm to estimate time-dependent web page authority. It incorporates web freshness at multiple time points to bias the web surfer's behavior on a temporal graph composed of multiple web snapshots. Experiments on a real-world archival corpus demonstrate its superiority over PageRank on both relevance and freshness by 17.8% and 13.5% in terms of NDCG@5. Results show ranking performance can benefit more from long-term historical web freshness and link structure. The best period covers the past 2.5 to 5 years.

Chapter 5

Learning to Rank for Freshness and Relevance

5.1 Introduction

The query stream seen by a web search engine and the interpretation of those queries change over time. Previous analysis has shown that web logs clearly reflect daily events in user queries [32]. For example, during seasonal events such as Halloween, there are always spikes in the frequency of related queries such as "halloween", "halloween costumes" and "pumpkins" (Figure 5.1). For many of the queries that correspond to events, the best answer may change over time (e.g., the latest SIGIR conference homepage for the query "sigir conference"). In more extreme cases, the major intent behind the same query can temporally vary; for instance, the query "US open" is more likely to be targeting the tennis open in September, and the golf tournament in June (Figure 5.2). Kulkarni et al. [81] refers to this class of temporally ambiguous queries as *shift* topics.

News events, depending on their significance, can cause enormous growth in frequency of related queries.[1] It is also not uncommon for news events to change the general meaning of a query. For example, the query "ipad" which could be treated as a misspelling for "ipod" in 2009, suddenly turned into a valid query with several related websites in 2010.[2] Therefore, making search engine results appear current and fresh is important to satisfy users' ever-changing information needs.

In this chapter, we focus on improving the ranking of results for queries based on their temporal profiles. Of course, the importance of the temporal profiles of queries extends beyond web result ranking; advertisement rankers have to address similar problems; related search and auto-complete suggestions must provide users with fresh and relevant alternatives to their queries; vertical search [49] ranking and triggering can be affected by temporal changes; and in general, the entire search experience can be influenced according to the temporal aspect of a query.

Learning ranking functions that can respond effectively to diverse temporal dynamics of queries is challenging. One of the difficulties is that traditional machine learning ranking algorithms fail to consider the interaction between freshness and relevance. While

[1] For example, the traffic caused by queries related to Michael Jackson's death in 2009, was so huge that Google mistook it as an attack (Source: Google Blog, 26 Jun 2009 [64]).

[2] It is probably still the case that some people mistype ipod as ipad. However, this group no longer represents the majority.

relevance clearly quantifies the topical matchability between query and web pages, freshness can be interpreted in different ways. For certain *temporal* queries such as breaking news, freshness is more meaningful when the actual page content reflects new information. Whereas, for *non-temporal* (time-insensitive) queries, it makes more sense to interpret freshness as the recency of page maintenance with respect to the time point of generating ranking lists (suppose web pages contain such information). Therefore, these two interpretations for freshness may be correlated to some extent but are not the same, considering that pages updated recently tend to record fresh information. It is worthwhile pointing out that both explanations can be part of the overall quality of search results that influences user search experience. In this work, the definition of freshness is sensitive to query temporal characteristics, varying on whether human editors (judges) can identify temporal intents concealed within queries. (See Section 5.4.3 for details.)

For certain *temporal* queries such as breaking news, relevance and freshness are highly correlated. Therefore, a ranker optimized for returning fresh documents may produce satisfactory results. However, for queries that are not usually time-sensitive (e.g., “facebook”, “machine learning”), paying too much attention to freshness may significantly hurt ranking effectiveness in terms of relevance. Among common ranking features, clicks, anchor-text and historical data might be the most powerful for answering time-insensitive queries. For temporal queries however, other features such as the rate of content change in documents may provide better signals [81]. Therefore, a ranker optimizing either freshness or relevance only may not be flexible enough to deal with the temporal dynamics of queries effectively.

To address this issue, previous work [20, 50] suggested training separate rankers for different classes of queries. The query is first classified according to its temporal profile, and then is sent to the appropriate ranker that has been optimized for either relevance or freshness. The main disadvantage of classification-based techniques is that selecting a wrong ranker due to misclassification can significantly degrade performance.

We propose a machine learning model that optimizes freshness and relevance simultaneously. Our flexible framework allows training multiple rankers with different optimization functions, and runs each query against all rankers with weights varying according to the query's temporal profile. This is in contrast with existing solutions that suggest selecting one ranker per query, and consequently has a lower risk of poor performance when queries are misclassified. In addition, instead of splitting the labeled data to train separate rankers, our technique leverages the entire data set in training all rankers. This approach is the first attempt to incorporate the trade-off between freshness and relevance into a single ranking framework.

Our work can be regarded as an extension to the family of *divide and conquer* (DAC) techniques for ranking [19]. In DAC, queries are clustered based on their feature representations, and separate rankers are trained with each for one cluster simultaneously. At test time, the query is compared against the generated cluster centroids and is ranked under all rankers with the weights depending on query-cluster similarity values. We follow a similar path since DAC enables specialized ranker training by considering query features, but we

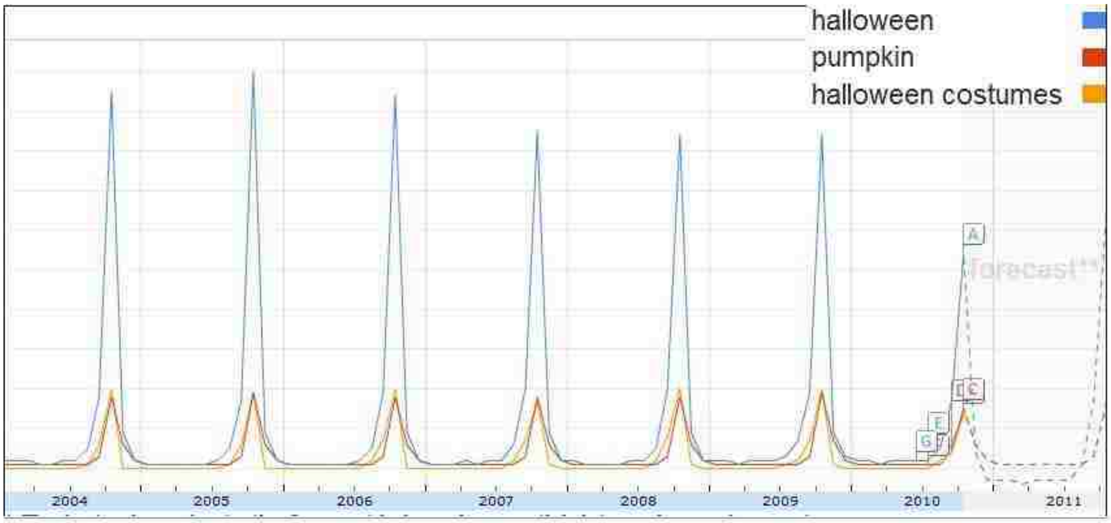


Figure 5.1: The query histograms for “Halloween”, “pumpkin” and “Halloween costumes” since 2004 as reported by Google insight for search. It can be seen that the queries follow similar temporal patterns.

incorporate multiple criteria (freshness and relevance) into ranking optimization. We also modify the DAC loss function by introducing a new query-document importance factor that emphasizes certain documents during training, and leads to further improvements in the results. Our experiments on a large web archive demonstrate that the rankers trained by our techniques can achieve better relevance and freshness compared to state-of-the-art alternatives.

5.2 Criteria-Sensitive Ranking

In this section, we introduce our *criteria-sensitive* divide-and-conquer ranking framework (denoted as CS-DAC) that incorporates the balance between relevance and freshness into training customized rankers that optimize both freshness and relevance.

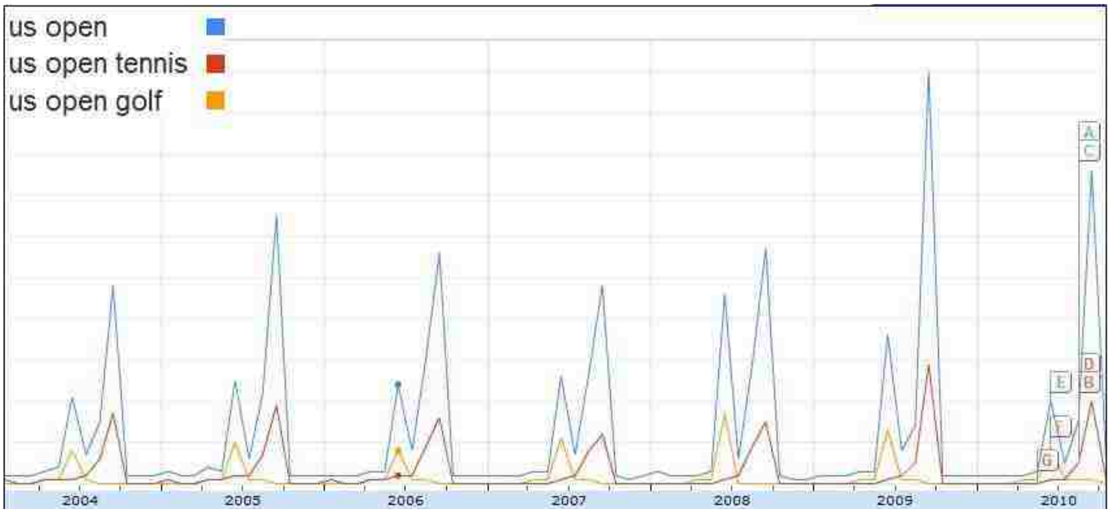


Figure 5.2: The query histogram for "us open", "us open tennis" and "us open golf" since 2004 as reported by Google insight for search. This example shows how the majority intent for a query can change over time.

5.2.1 CS-DAC framework

A typical ranking function f with ω parameters takes a query-document feature vector $\mathbf{X}$ as input and produces ranking scores of documents.

$$\hat{\mathbf{y}} = f(\mathbf{X}, \omega) \tag{5.1}$$

The common goal of learning to rank systems is to find a ranking model f^* that takes query-document feature vectors as input, and produces a document ranking—as close as possible to the *oracle* ranking of documents according to their relevance labels $\mathbf{y}$—by minimizing the ranking risk aggregated from the loss $\mathcal{L}$ of all training queries.

$$f^* = \arg\min_f \sum_q \mathcal{L}(f(\mathbf{X}_q, \omega), \mathbf{y}_q) = \arg\min_f \sum_q \mathcal{L}(\hat{\mathbf{y}}_q, \mathbf{y}_q)$$

By considering query differences in the DAC framework, we essentially *cluster*[3] training queries based on their ranking characteristics, and train one ranker per cluster. Each query contributes to learning all rankers with different importance based on its topical affinity to query clusters. Each ranker f_i^* is learned via:

$$f_i^* = \arg\min_{f_i} \sum_{q \in \mathcal{Q}} \mathcal{I}(q,i)\mathcal{L}_i(\hat{\mathbf{y}}_q, \mathbf{y}_q) \tag{5.2}$$

where $\mathcal{Q}$ is the training query set, and $\mathcal{I}(q,i)$ is the importance of query q with respect to the i^{th} ranking model.

To account for relevance and freshness simultaneously, we propose to use hybrid labels that are generated based on freshness and relevance judgments.[4] For this purpose, we exploit a weighted harmonic mean function which maps relevance and freshness grades (i.e., $y^R_{q,d}$ and $y^F_{q,d}$ on the query-document pair $<q,d>$) to a single equivalent numerical score $\widetilde{y}_{q,d}$ for training f_i^*. We believe harmonic mean is appropriate here since (1) it heavily biases towards the minimum score; (2) it is more sensitive when $y^R_{q,d}$ and $y^F_{q,d}$ are close; and (3) it has been shown as a good optimization metric for tasks such as learning to rank for efficiency [118] and classification. Formally, $\widetilde{y}_{q,d,i}$ is defined as:

$$\widetilde{y}_{q,d,i} = \frac{(1+\beta_i^2) \cdot y^R_{q,d} \cdot y^F_{q,d}}{y^R_{q,d} + \beta_i^2 \cdot y^F_{q,d}} \tag{5.3}$$

where parameter β_i sets the trade-off between relevance and freshness for each ranker, and is learned during training. Allowing different values of β for rankers enables a flexible

[3] We use query cluster, topic and category interchangeably.

[4] Generating hybrid labels (single aggregate objective functions), is a simple form of multi-criteria optimization [115].

framework where each ranker can assign different weights to freshness and relevance. It also means that each query-document pair may affect the pairwise learning of each ranker differently.[5] Therefore, we factorize query-document pair importance as follows:

$$f_i^* = \arg\min_{f_i} \sum_{q\in\mathcal{Q}} \mathcal{I}(q,i)\times$$

$$\sum_{<d_1,d_2>\in\mathcal{D}_q} \mathcal{U}'(q,i,d_1,d_2)\mathcal{L}_i\Big(\begin{bmatrix}\hat{y}_{q,d_1,i}\\ \hat{y}_{q,d_2,i}\end{bmatrix},\begin{bmatrix}\widetilde{y}_{q,d_1,i}\\ \widetilde{y}_{q,d_2,i}\end{bmatrix}\Big) \tag{5.4}$$

where, $\mathcal{D}_q$ is the set of preferential query-document pairs with respect to query q, and $\mathcal{U}'(q,i,d_1,d_2)$ is the importance of $<d_1,d_2>$ in training for query q with respect to the i^{th} ranking model. For simplicity, we assume $<q,d_1>$ and $<q,d_2>$ are independent, and so factorize the importance of the preferential pair $\mathcal{U}'(q,i,d_1,d_2)$ as follows.

$$\mathcal{U}'(q,i,d_1,d_2) = \mathcal{U}(q,i,d_1)\cdot\mathcal{U}(q,i,d_2) \tag{5.5}$$

where $\mathcal{U}(q,i,d_1)$ is the importance of query-document pair $<q,d_1>$ in training for query q with respect to the i^{th} ranking model.[6]

5.2.2 Ensemble ranking

Given an unseen query q', we first profile its query characteristics, and then calculate its distances to the centroids of existing query clusters $\mathbf{c}_1$, $\mathbf{c}_2$, ..., $\mathbf{c}_n$. The trained ranking functions are then scored according to the normalized distance between the query and

[5] Similar ideas can be applied to list-wise and point-wise ranking learning algorithms.

[6] The independence assumption is unrealistic, but we believe it is not unreasonable because if two query-documents pairs are important, then so is their preferential pair.

their corresponding clusters (a.k.a. query importance $\mathcal{I}$), given by:

$$W_i = \frac{\mathcal{I}(q', i)}{\sum_{i'=1}^{n} \mathcal{I}(q', i')} \tag{5.6}$$

The query q' is run against all n rankers (one for each cluster), and the final results $\theta_{q'}$ are produced according to the ensemble ranking of their outputs. That is,

$$\theta_{q'} = \sum_{i=1}^{n} W_i f_i^*(\mathbf{X}_{q'}, \omega_i) \tag{5.7}$$

where f_i^* is the i^{th} ranking model, $\mathbf{X}_{q'}$ is the query-document feature vectors for query q', and ω_i is the feature weights.

The CS-DAC framework summarized in Equation 5.4 consists of three main factors: query importance ($\mathcal{I}$), ranker-specific query-document importance ($\mathcal{U}$), and the loss function ($\mathcal{L}$). We continue by describing each of these items.

5.2.3 Query importance ($\mathcal{I}$)

In the *divide* step of the DAC framework, the query space is split into a few clusters based on *criteria-sensitive* features. These are the features that are extracted from the top-ranked documents of a basic reference ranker (BM25 [104] in our work) for the query. We will provide more details about these features in Section 5.4.5.

The $\mathcal{I}(q, i)$ values provide a Binomial distribution over each of the criteria-sensitive query clusters, and specify the importance of different ranking functions. We use a Gaussian Mixture model as a soft k-means clustering to group queries into clusters. The importance of query q with respect to the i^{th} cluster is thus given by:

$$\mathcal{I}(q,i) = 1 - \frac{\|\mathbf{p}_q - \mathbf{c}_i\|^2}{\max_{q' \in \mathcal{Q}} \|\mathbf{p}_{q'} - \mathbf{c}_i\|^2} \tag{5.8}$$

where $\mathbf{p}_q$ and $\mathbf{c}_i$ respectively denote the feature vectors of query q and the centroid of the i^{th} cluster, and $\mathcal{Q}$ represents the set of training queries. Therefore, $\mathcal{I}(q,i)$ is scaled between $[0,1]$, and is inversely proportional to the distance between query feature vector $\mathbf{p}_q$ and cluster centroid $\mathbf{c}_i$.

5.2.4 Document importance ($\mathcal{U}$)

In pairwise learning to rank methods, the importance of a document with label $\mathbf{y}$ during training depends on the number of times it is compared to other documents with different labels. Due to the ranker-specific value of β which is set during training, a query-document pair with the same relevance and freshness grades can get unequal hybrid labels under different rankers, and hence may contribute unequally in training various rankers. Besides, centralizing hybrid label distribution within each query cluster stabilizes the correlation between freshness and relevance, which further emphasizes the effect of β_i in Equation 5.3. To factorize these impacts, we introduced the $\mathcal{U}$ component in Equation 5.4. We estimate the importance of a query-document pair with label $\mathbf{y}_{q,d}$ by the likelihood of visiting that label in the training dataset, under the assumption that the importance of a hybrid label is proportional to the ratio of query-document pairs with that label in the training dataset.

We define the document importance $\mathcal{U}$ as below.

$$\mathcal{U}(q,i,d) = \frac{\sum_{q'\in\mathcal{Q}} N(q',i,\mathbf{y}_{q,d})\cdot N(q',i,\neg\mathbf{y}_{q,d})}{\sum_{\mathbf{y'}\in\mathcal{Y}_i}\sum_{q'\in\mathcal{Q}} N(q',i,\mathbf{y'})\cdot N(q,i,\neg\mathbf{y'})} \tag{5.9}$$

where $\mathcal{Y}_i$ is the space of labels for ranker i, and $\mathcal{Q}$ denotes the training query set. The number of documents with and without label $\mathbf{y}$ are represented by $N(q,i,\mathbf{y})$ and $N(q,i,\neg\mathbf{y})$. Equation 5.9 can be regarded as a function of the unique hybrid label $\mathbf{y}_{q,d}$, and is denoted as $w(\mathbf{y}_{q,d})$ for short.

There are two potential problems with this type of normalization: (1) additional inter-label dependencies may arise from comparing common labels (e.g., $\mathbf{y}_a$ and $\mathbf{y}_b$, versus $\mathbf{y}_b$ and $\mathbf{y}_c$), and, (2) overemphasizing certain documents inevitably introduces bias in ranking. To overcome these issues, we exploit a random walk approach to determine $\mathcal{U}$ (instead of Equation 5.9) that has the effect of smoothing document importance values.

To perform a random walk, we first construct a fully connected bipartite graph $G(V,E)$ (one graph per ranker) in which each node (state) v stands for a unique *hybrid* label $\mathbf{y}$ (associated with the weight $w(\mathbf{y})$), and each edge e is associated with a weight computed according to the number of times the labels of the connected nodes compare with each other during training. At each step, the random walk surfer jumps to a random node with probability d (selection among random nodes is proportional to $w(\mathbf{y})$ values) or follows some connected edge with probability $1-d$ (the selection among connected edges is proportional to the weights on edges). The value of d can be pre-defined or set during the training and validation. When d equals 1, the probability that the random surfer reaches every node (state) is proportional to the direct comparison between preferential query-document pairs with different hybrid labels. Whereas, $d = 0$ suggests document importance entirely propagates through indirect comparison between preferential query-document pairs. Parameter d actually controls the extent that such propagation (from indirect comparison) influences the computation of document importance. We analyze the importance of $\mathcal{U}$, with and without smoothed probabilities in Section 5.5.4.

5.2.5 Loss function ($\mathcal{L}$)

The core of each ranker in our CS-DAC framework is a loss function that is trained for hybrid labels (Equation 5.3). We follow Bian et al. [20] and use RankSVM [72] as our basic learning algorithm although it is important to note that the framework is flexible and not restricted to any particular learning technique.

RankSVM [72] is designed to maximize the margin between positively and negatively labeled documents in the training data by minimizing the number of discordant pairs. The RankSVM optimization problem is defined as:

$$\arg\min_{\omega,\xi_{q,i,j}} \frac{1}{2}\|\omega\|^2 + C\sum_{q,i,j}\xi_{q,i,j} \quad \textit{subject to} \tag{5.10}$$

$$\forall y_i^q \succeq y_j^q : \quad \omega^T X_i^q \geq \omega^T X_j^q + 1 - \xi_{q,i,j},$$

$$\forall_q \forall_i \forall_j : \quad \xi_{q,i,j} \geq 0$$

where the non-negative slack variable $\xi_{q,i,j}$ is used to approximate the NP-hard optimization solution by minimizing the upper bound $\sum \xi_{q,i,j}$. Parameter C sets the trade-off between the training error and the margin size. The query-document feature vectors for documents i and j are respectively represented by X_i^q and X_j^q. The notation $y_i^q \succeq y_j^q$ implies that the document i is ranked higher than document j with respect to query q in the training dataset (i has the same or higher relevance than j).

CS-DAC modified the RankSVM loss function by incorporating query importance ($\mathcal{I}$) and document importance ($\mathcal{U}$). Formally, the i^{th} ranking model of CS-DAC is optimized via:

$$\begin{aligned} \arg \min_{\omega_i, \xi_{q,j,k}} \frac{1}{2}\|\omega_i\|^2 + C \sum_{q,j,k} \xi_{q,j,k} \\ \textit{subject to,} \quad \forall \widetilde{y}_{q,j,i} \succeq \widetilde{y}_{q,k,i} : \mathcal{I}(q,i)\mathcal{U}(q,i,j)\omega_i^T X_j^q \\ \geq \mathcal{I}(q,i)\mathcal{U}(q,i,k)\omega_i^T X_k^q + 1 - \xi_{q,j,k}, \\ \forall_q \forall_i \forall_j : \quad \xi_{q,i,j} \geq 0 \end{aligned} \tag{5.11}$$

where $\xi_{q,j,k}$ is the slack variable and parameter C sets the trade-off between training error and the margin size.

In CS-DAC, several rankers are trained simultaneously, and each ranking function f_k^* (see Equation 5.4) is optimized using the CS-DAC loss function and hybrid labels. The β values are tuned via hill climbing based on the *hybrid NDCG* values of the final ranking lists merged from different rankers. That is, each ranker is trained on different values of β and the best combination of rankers is chosen by hill climbing on the training and validation data. Here, *hybrid NDCG* extends the commonly used evaluation metric NDCG [71] to take hybrid labels for evaluation, since this new freshness-sensitive metric can take into account both freshness and relevance into a single measurement, aiming to quantify the overall search quality. Formally, we define *hybrid NDCG* as below:

$$\textit{hybrid NDCG}(n) = Z_n \sum_{j=1}^{n} \frac{2^{(\gamma \mathbf{y}_R + (1-\gamma)\mathbf{y}_F)} - 1}{\log_2(j+1)} \tag{5.12}$$

where Z_n is the oracle *discounted cumulative gain* at ranking cutoff n, that bounds the NDCG values between 0 and 1. The $\mathbf{y}_R$, and $\mathbf{y}_F$ values—also known as *gains*—are assigned according to the relevance and freshness labels of documents. Parameter γ specifies the trade-off between relevance and freshness and is set to 0.5 in our experiments. Note that $\gamma = 1$ turns hybrid NDCG into typical relevance-based NDCG, while setting γ to zero, makes it the same as the NDCF metric [51]. Dai and Davison [41] also adopted NDCG with freshness labels, although they did not refer to it as NDCF. While other combination forms may better fit the search utility that quantifies comprehensive users' satisfaction, we leave the best definition of *hybrid NDCG* for future work.

5.3 Multi-objective Optimization in Ranking

One may notice that the way that criteria-sensitive ranking leverages freshness and relevance is through the hybrid label defined in Equation 5.3. While the parameter β controls freshness-relevance trade-off within a harmonic function, it is unknown whether harmonic mean is the most appropriate way of combining multiple ranking criteria more generally. In this section we focus on this problem. It is not a trivial problem since these ranking criteria may interact with each other in a query-dependent manner. Relevance and freshness with respect to breaking news queries is one such example [50]. Similar scenarios exist in the information filtering and recommendation domains, where users' ratings on several aspects may correlate with each other depending on user profiles, and consequently affect the prediction models of user preferences on items [89].

Prior work that considered users' multi-criteria objectives in search or collaborative filtering have been mostly inspired by multi-criteria decision making (MCDM) theory from the operations research community [115]. The preference between different criteria is quantified by utility measures that affect optimization through preference model representation. The commonly used preference models for search or recommendation tasks include *value-focused* models [118, 123] and *outranking relations* models [56]. While these approaches exploit the search quality on each aspect (criterion-specific ratings) to enhance overall quality (measurement ratings), they ignore the inter-relationship between different objectives.

In this section, we explore the influence of interactions and correlations between multiple criteria for ranking optimization in the context of web search. As a preliminary step, we analyze the influence of bi-criteria inter-relationship on pairwise ranking models though the analysis can be generalized to other multi-criteria scenarios. While the definition of *measurement* utility is an open issue, we use the minimum relative ranking improvement on both criteria (denoted as *RelImp*) to measure the influence of bi-criteria optimization, emphasizing its relative benefits compared to optimizing for each single objective. We define $RelImp$ as follows:

$$RelImp = \frac{\min_{c,obj}[\text{perf}(c, bi\text{-}obj) - \text{perf}(c, obj)]}{\text{perf}(c, obj)} \tag{5.13}$$

where perf$(c, *)$ is the performance on criteria c when optimizing for objective "$*$". Our research explores the effect of correlation between two ranking criteria on the benefit of bi-objective ranking optimization, focusing on three main issues: (1) what is the correlation scale that can benefit *RelImp*? (2) how much benefit can it bring? and (3) what does a useful preference model look like under different correlation scales? We exploit a value-focused preference model implicitly for ranking optimization through minimizing bi-criteria ranking risk based on *hybrid* labels that combine the quality of documents on both aspects. We will demonstrate that the correlation between multiple objectives (freshness and relevance in our case) may influence the outcome of multi-criteria on ranking optimization in Chapter 5.5.

5.3.1 Methodology

Given a query q and its associated documents $d_1,\ldots,d_n$, each query-document pair $< q, d_k >$ is rated based on its quality on each facet, i.e., $y^{(1)}_{q,d_k}$ and $y^{(2)}_{q,d_k}$. By exploiting *hybrid* labels to combine the overall quality, we average the score achieved on each aspect as the hybrid label for $< q, d_k >$, defined as:

$$\widetilde{y}_{q,d_k} = \Big(\frac{1}{n} \cdot \sum_{i=1}^{n} (y^{(i)}_{q,d_k})^m\Big)^{\frac{1}{m}} \tag{5.14}$$

where $n = 2$ is the number of facets (e.g., freshness and relevance), and m determines the type of *hybrid* label function; quadratic mean (QM), arithmetic mean (AM), geometric mean (GM) and harmonic mean (HM) respectively for $m = 2$, $m = 1$, $m \to 0$ and $m = -1$. These variants reflect how sensitive the *hybrid* label is with respect to the lower (higher) rating scores on both aspects, assuming that the rating scores on two aspects fall into the same scale. We believe this perspective is reasonable since the criteria for judging query-document pair quality may vary from one person to another. We also include two extreme cases, i.e., MIN and MAX, representing the minimum and maximum rating scores on two aspects.

Pairwise ranking learning algorithms train a set of parameters ω by minimizing the ranking risk aggregated from loss of misclassified preferential query-document pairs based on relevance. By exploiting *hybrid* labels, we optimize model parameters by:

$$f^* = \arg\min_f \sum_{q\in\mathcal{Q}} \sum_{<d_i,d_j>\in\mathcal{D}_q} \mathcal{L}(\widehat{y}_{q,d_i}, \widehat{y}_{q,d_j}, \widetilde{y}_{q,d_i}, \widetilde{y}_{q,d_j}) \tag{5.15}$$

where $\mathcal{D}_q$ is the set of preferential query-document pairs for query q, and $\mathcal{L}$ is the loss function that penalizes $< d_i, d_j >$ if its predicted preferential relationship (based on $\widehat{y}_{q,d_i}$ and $\widehat{y}_{q,d_j}$) is discordant with groundtruth (based on $\widetilde{y}_{q,d_i}$ and $\widetilde{y}_{q,d_j}$). We use RankSVM [72] as our basic ranker. We note that the loss function defined on preferential query-document pairs $< d_i, d_j >$ is a linear combination of the loss on each criterion (i.e., quantifying how much the prediction of the relative preference between d_i and d_j is inconsistent with the users' judgements on each individual criterion), with the coefficient depending on the actual rating scores for d_i and d_j on both criteria (groundtruth).

5.3.2 Generality

The preference between URL i and j can be represented by $I(y_i^R > y_j^R)$ for pair-wise ranking learning algorithms, where $I(y_i^R > y_j^R)$ is an indication function, achieving 1 if

Name	Mapping Function	$f^c(y_i^1, y_i^2, \ldots, y_i^n; y_j^1, y_j^2, \ldots, y_j^n)$
AM	$\frac{1}{n}\sum_k y^k$	$\frac{1}{n}\lvert\Delta y^c\rvert$
GM	$\sqrt[n]{\prod_k y^k}$	$(\prod_{k=1}^{c-1} y_i^k)(\prod_{k=c+1}^{n} y_j^k)\lvert\Delta y^c\rvert$
HM	$(\frac{1}{n}\sum_k \frac{1}{y^k})^{-1}$	$\frac{n\prod_{k'\neq c} y_i^{k'} y_j^{k'}\lvert\Delta y^c\rvert}{(\sum_{k=1}^{n}\prod_{k'\neq k} y_i^{k'} y_i^{k'})(\sum_{k=1}^{n}\prod_{k'\neq k} y_j^{k'} y_j^{k'})}$
QM	$\sqrt{\sum_k (y^k)^2}$	$(y_i^c + y_j^c)\lvert\Delta y^c\rvert$

Table 5.1: f^c in linear combination among multiple criteria.

$y_i^R > y_j^R$ is true, else -1. When extending to multiple criteria, we have the following claim.

Claim 5.3.1 *Given URL i and j, the preference based on $\mathbf{y'}$ linearly correlates with the preference based on $\mathbf{y}^c$, where c is one type of criteria, such as relevance or freshness. Formally,*

$$I(\Delta y' > 0) = I(\sum_c f^c(y_i^1, \ldots, y_i^n; y_j^1, \ldots, y_j^n) I(\Delta y^c > 0) > 0) \tag{5.16}$$

where y's are the comprehensive scores for URLs, and y^cs are the actual rating scores on criterion c for URLs. $\Delta y' = y_i' - y_j'$, and $\Delta y^c = y_i^c - y_j^c$. They measure the difference of the comprehensive scores and rating scores on criterion c between preferential pair $< d_i, d_j >$. f^c is a coefficient of Δy^c functioning on the scores of all criteria associated with i and j, and n is the number of criteria.

We summarize f^c for all proposed mapping functions in Table 5.1. It clearly shows how the multi-criteria-based scores mutually determine the instance-dependent importance on each criteria for learning ranking models.

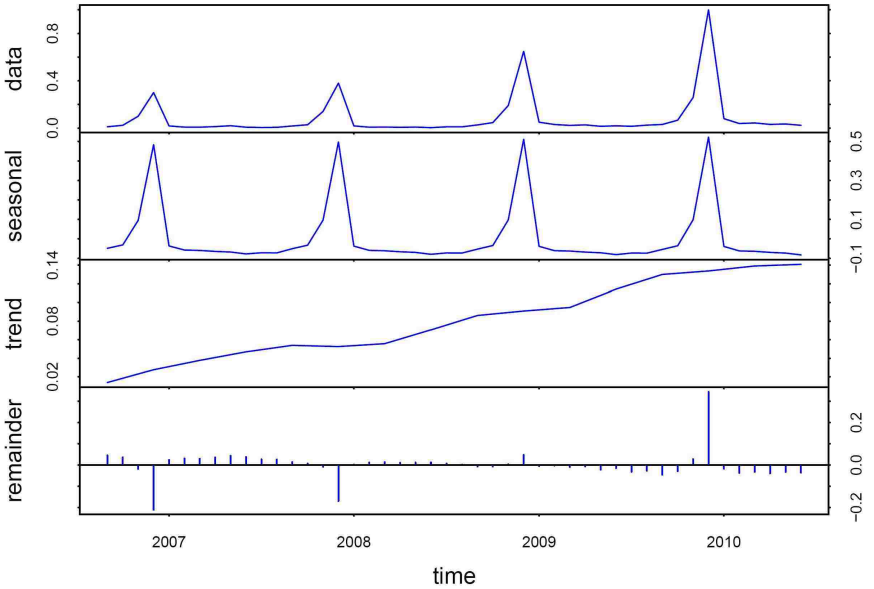


Figure 5.3: The STL decomposition (a seasonal-trend decomposition procedure based on Loess) [37] of a time series into seasonal, trend and remainder components. The data is generated from the click histogram of the query *jingle bells* in a commercial search engine.

5.4 Evaluation Platform

5.4.1 Introduction

We presented the way of incorporating the temporal characteristics of queries into learning to rank systems in Chapter 5. To show the effectiveness of our proposed approach, this chapter focuses on how we built the evaluation platform. The purpose of this evaluation platform is to provide a relatively objective environment for comparing multiple ranking systems on relevance and freshness of their search results. Such a platform includes

- a web corpus and queries (Section 5.4.2);
- relevance/freshness judgements (Section 5.4.3);

- ranking features (Section 5.4.4);
- query cluster features (Section 5.4.5);

We introduce them one by one. We will present the evaluation results in Chapter 5.5.

5.4.2 Testbed data

Standard learning to rank datasets only contain relevance judgments for query-document pairs without any information regarding their freshness. The query-URL pairs are typically characterized by dynamic features (e.g., BM25, tfidf) and static features (e.g., PageRank score). Document temporal features are not included to better respond to diverse query temporal characteristics. Based on these concerns, common publicly available datasets are not suitable for our experiments.

We built a new testbed based on a large archival web corpus that is the same as the one used in Section 4.4. Our dataset contains 158 million unique URLs and 12 billion links from the `.ie` domain, covering the time span from January 2000 to December 2007 (one snapshot per month and 88 in total). We removed pages with less than five snapshots, and only kept the remaining 3.8 million unique pages with 435 million links in total.

We choose *April 2007* as our time point of interest for ranking evaluation. We constructed two *temporal* and *non-temporal* query sets, each containing 90 queries. While the query size is small, the queries in the temporal set are manually selected from Google Trends suggestions for Ireland, which were popular during *April 2007* [65]. These ninety queries are the same as those in Table 4.2. For the non-temporal set, we first randomly sampled queries from a 2006 MSN query log (i.e., generating a representative query sample from a real-world search log), and then automatically filtered out about 10% of them that were detected as *potentially temporal* by a commercial classifier. The classifier has high precision (almost all Google Trend queries are detected as temporal), and uses several years of the query-frequency history extracted from the query logs of a major commercial search engine. We report the ninety non-temporal queries in Table 5.2.

5.4.3 Judgments and metrics

To evaluate the quality of search results based on our testbed, we choose to use Amazon Mechanical Turk[7] to collect the groundtruth, i.e., the freshness and relevance labels of query-document pairs. Amazon Mechanical Turk is an open and convenient marketplace for working on human intelligence tasks (HITs). Representative HITs include but are not limited to basic open-ended questions, categorization, and surveys. Requesters bid their HITs and appeal to the workers who are interested to work on these HITs. The reasons that we select Amazon Mechanical Turk (AMT) to collect our groundtruth are: (1) it provides easy, cheap and fast labeling; (2) it maintains a ready-to use infrastructure; and (3) it allows early, iterative, and frequent experiments [5]. Given such advantages, AMT has been a proven crowdsourcing platform for major IR shared task evaluations.

Given a query-document (URL) pair, the judges were instructed to assess the quality of the URL with respect to both relevance and freshness. For relevance, the selection was

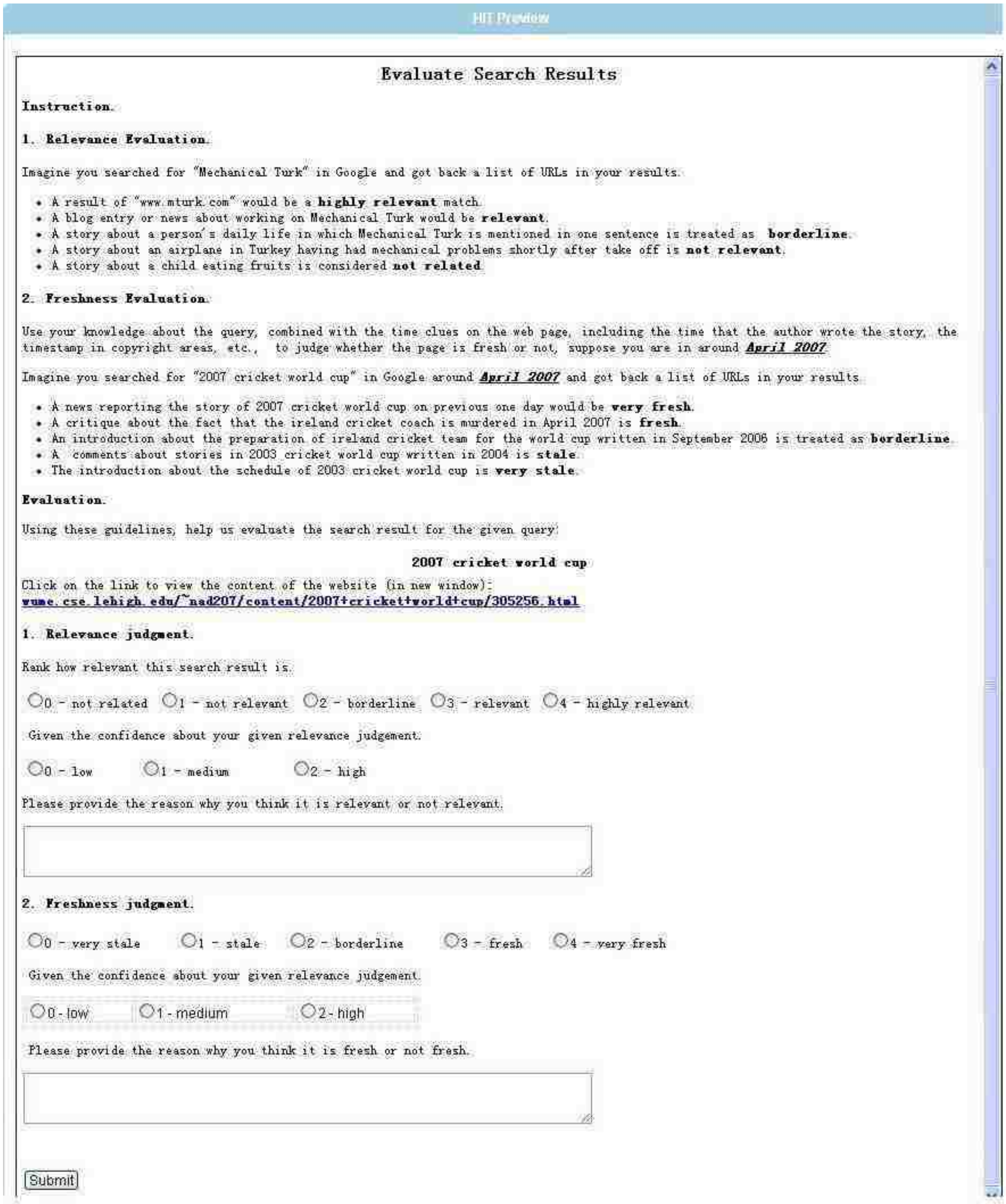
[7] `http://www.mturk.com`

hawaii child custody	autocad lt
core youth leader training	hardcore movie download
gateway community college	wvde job bank
middle bay country club	quest diagnostics
folding closet doors	wood floor finishers in the chicago area
kelly blue book	how to sell a service based company
babe ruth home runs	motels naples fl
vacations for seniors	kgo am radio
layered cake and pudding recipe	sex with ouji
levaquin and class	home inspection hamilton ohio
ups track	faith reformed church
dcx co car programs lease boat building	stargate atlantis rachel luttrell download
	critical mass
beautiful cheerleaders of the chivas	image line ezgenerator
fintess center floor plans	adobe mountain school
parrandero means	monterey hotels reviews
real men of genius	zip code crestview florida
ebay motors	funny quotes
mail	combined metals inc. nevada
walmart cakes	hill high school
gm parts	rockdale citizen
wse verify trust	steel riders mc
cheap ipod	ganley motors
american airlines	house value
las vegas	bake dbean with hamburger recipes
concan texas	photo of naked women
models femen	norwegian cruise
compact tractor discount	highest credit score
nj lottery results	massage harmony austin
deer feeding recipes	wells fargo
best non-composite slow pitch softball bats	osha government
spencer studio dirt race car	kazaa 2.6 patch connect
music	darkblondie spain
maps	five year old complains of legs hurting
symptoms of heat stroke	alkyphenols structure
kids dada supreme	agape christian fellowship church in virginia
the lavon affair	hotmail
mission to nigeria extrajudicial	cna boat moak
world of outlaws	elm and red river
njaac national plant city	vets id stolen
santo security	assessing student learning
jean vest	dillion beach northern california
cronin attorney	liberalism and colonial america
portuguese steak sandwish recipe	faa far part 91
window treatments lincoln nebraska	ford focus
coach david elson	australian flowrs

Table 5.2: Set of ninety non-temporal queries used for ranking evaluation in IA data set.

among *highly relevant*, *relevant*, *borderline*, *not relevant* and *not related*, which was further translated to integer gains ranging from 4 to 0. For freshness, editors were instructed to judge the URL freshness for the given query according to our chosen point in time (April 2007).[8] Judges could select between *very fresh*, *fresh*, *borderline*, *stale*, and *very stale*, which we transferred into $\{4, 3, 2, 1, 0\}$. Judges were also required to provide the confidence of their judgements by choosing between *high*, *medium* and *low*. Judgments with low confidence were resubmitted for labeling. Table 5.3 shows the guideline of query-URL pair judgments used by Mturk workers. Figure 5.4 gives one example of HITs that we designed for evaluating the freshness and relevance of query document pairs. We set the reward per assignment to 5 cents. We only select the participants whose past HITs approval rate was greater than or equal to 95%. The standard deviations of relevance and freshness judgements on a random sample of 76 query URL pairs among three judgers are 0.88 and 1.02 respectively. We have an average of 71 URLs per query judged by one or more participants from AMT.

Freshness and relevance are evaluated by *hybrid NDCG*, and so when $\gamma = 0$ or $\gamma = 1$, this corresponds to NDCF [51] and NDCG, respectively.



Evaluate Search Results

Instruction.

1. Relevance Evaluation.

Imagine you searched for "Mechanical Turk" in Google and got back a list of URLs in your results.

- A result of "www.mturk.com" would be a **highly relevant** match.
- A blog entry or news about working on Mechanical Turk would be **relevant**.
- A story about a person's daily life in which Mechanical Turk is mentioned in one sentence is treated as **borderline**.
- A story about an airplane in Turkey having had mechanical problems shortly after take off is **not relevant**.
- A story about a child eating fruits is considered **not related**.

2. Freshness Evaluation.

Use your knowledge about the query, combined with the time clues on the web page, including the time that the author wrote the story, the timestamp in copyright areas, etc., to judge whether the page is fresh or not, suppose you are in around ***April 2007***

Imagine you searched for "2007 cricket world cup" in Google around ***April 2007*** and got back a list of URLs in your results.

- A news reporting the story of 2007 cricket world cup on previous one day would be **very fresh**.
- A critique about the fact that the ireland cricket coach is murdered in April 2007 is **fresh**.
- An introduction about the preparation of ireland cricket team for the world cup written in September 2006 is treated as **borderline**.
- A comments about stories in 2003 cricket world cup written in 2004 is **stale**.
- The introduction about the schedule of 2003 cricket world cup is **very stale**.

Evaluation.

Using these guidelines, help us evaluate the search result for the given query:

2007 cricket world cup

Click on the link to view the content of the website (in new window):
wume.cse.lehigh.edu/~nad207/content/2007+cricket+world+cup/305256.html

1. Relevance judgment.

Rank how relevant this search result is.

○0 - not related ○1 - not relevant ○2 - borderline ○3 - relevant ○4 - highly relevant

Given the confidence about your given relevance judgement.

○0 - low ○1 - medium ○2 - high

Please provide the reason why you think it is relevant or not relevant.

2. Freshness judgment.

○0 - very stale ○1 - stale ○2 - borderline ○3 - fresh ○4 - very fresh

Given the confidence about your given relevance judgement.

○0 - low ○1 - medium ○2 - high

Please provide the reason why you think it is fresh or not fresh.

Figure 5.4: An example of HITs used for evaluating the freshness and relevance of query document pairs.

Table 5.3: Relevance and freshness judging guidelines for mechanical turk editors.

1. Relevance Evaluation.

Imagine you searched for "Mechanical Turk" in Google and got back a list of URLs in your results.

- A result of "www.mturk.com" would be a **highly relevant** match.
- A blog entry or news about working on Mechanical Turk would be **relevant**.
- A story about a person's daily life in which Mechanical Turk is mentioned in one sentence is treated as **borderline**.
- A story about an airplane in Turkey having had mechanical problems shortly after take off is **not relevant**.
- A story about a child eating fruits is considered **not related**.

2. Freshness Evaluation.

Use your knowledge about the query, combined with the time clues on the web page, including the time that the author wrote the story, the timestamp in copyright areas, etc., to judge whether the page is fresh or not, suppose you are in around April 2007. Now imagine you searched for "2007 cricket world cup" in Google around April 2007 and got back a list of URLs in your results.

- A news reporting the story of 2007 cricket world cup on previous one day would be **very fresh**.
- A critique about the fact that the ireland cricket coach is murdered in April 2007 is **fresh**.
- An introduction about the preparation of ireland cricket team for the world cup written in September 2006 is treated as **borderline**.
- A comment about stories in 2003 cricket world cup written in 2004 is **stale**.
- The introduction about the schedule of 2003 cricket world cup is **very stale**.

Table 5.4: Temporal ranking features used by RankSVM in the CS-DAC framework and baseline methods. The features (except for TPR) are produced from the STL decomposition [37] of time series generated from the content changes in title, body, heading, anchor, and page/link activities [41].

Feature name	**Feature description**
Slp(τ)	Slope of trend component T_τ.
Amp(τ)	Amplitude of seasonal component S_τ.
Rp(τ)	Relative position in S_τ.
Cs(τ)	Confidence of seasonality.
Cr(τ)	Confidence of regularity.
TPR	Timed PageRank [133].

Table 5.5: The way of fold splitting.

Run	**Training Set**	**Validation Set**	**Test Set**
1	folds 1,2,3	fold 4	fold 5
2	folds 2,3,4	fold 5	fold 1
3	folds 3,4,5	fold 1	fold 2
4	folds 4,5,1	fold 2	fold 3
5	folds 5,1,2	fold 3	fold 4

5.4.4 Ranking features

Each individual query-document pair is characterized by a feature vector when training rankers. These features can be grouped into non-temporal and temporal ones. The non-temporal features include several commonly used text-similarity scores such as BM25 [107], and language modeling [136], computed over different fields of documents (heading, title, body). They also include a few well-known link-based static features such as the number of inlinks and PageRank [23]. We have 97 non-temporal features in total.[9] To avoid the over-fitting problem, we proceed a simple feature selection. First, we train and evaluate a rankSVM ranker based on five-fold cross-validation. We split 90 temporal and non-temporal queries respectively into five folds in a sequential way. We present how we conduct five-fold cross-validation in Table 5.5. Second, we select the top n discriminative features ($n = 27$), i.e., the features corresponding to the largest coefficients in our rankers, as the final ranking feature set, listed in Table 5.6.

[8] Admittedly, judging for freshness according to an arbitrary time in the past could be a difficult task. However, the choice was dictated to us by the time span of our dataset.

[9] See `http://wume.cse.lehigh.edu/~nad207/temporalquery/featurelist.pdf` for details (from Feature 1 to 91).

The temporal ranking features are generated by measuring the changes in the contents of documents with respect to their previous snapshots. For this purpose, we build a time series of each document's content changes, by going through the entire time span and comparing the TFIDF similarity of the document at each point with the previous and next versions. We generate separate time series for different document fields (heading, title, body), and use *STL seasonal-trend decomposition* [37] to decompose each time series τ into trend ($\mathcal{T}$), seasonal ($\mathcal{S}$) and remainder ($\mathcal{R}$) components.

$$STL(\tau) = \mathcal{T}_\tau + \mathcal{S}_\tau + \mathcal{R}_\tau \tag{5.17}$$

The same steps are repeated to decompose the time series generated based on link and page activities (create, remove, update) [41]. Figure 5.3 depicts an example of STL decomposition on a time series. In this instance, the time series (data) is generated from the frequency distribution of the query *jingle bells* in the logs of a commercial search engine. The same decomposition can be applied to a sequence of TFIDF scores, PageRank values or any other type of time series data. We use the output of STL decomposition for different time series to generate our temporal ranking features as summarized in Table 5.4. The details of each individual feature are as follows.

- **Slp**: Slope of T_i. It captures the speed of field content change in the long term. Previous work on mining anchor text trends for retrieval [42] has demonstrated the change of historical anchor importance weights on ranking effectiveness. We infer that this feature can boost ranking performance in a similar way.

- **Amp**: Amplitude of S_i. It captures the scale of field content change speed in each year. We conjecture that obvious change on field content maintenance speed within each year may be a good sign for answering seasonal queries, and so we hope this feature can be especially useful for seasonal queries.

- **Rp**: Relative position of the current time point (i.e., our interested time point for ranking evaluation) with respect to the nearest peak and valley values of S_i component, defined as $\frac{|\text{highest\_slp}|-|\text{lowest\_slp}|}{|\text{highest\_slp}|+|\text{lowest\_slp}|}$, where *highest_slp* and *lowest_slp* are the slopes of lines connecting from the value at current time point to its nearest peak and valley values on S_i respectively. We conjecture that ranking performance is sensitive to query temporal position within its period, and so use this feature to suggest the insight to how relevant web pages are for a given query to some extent.

- **Cs**: Confidence of seasonality. It represents how well the time series can be explained by S_i versus T_i, defined as $\sum_i |S_i| / \sum_i |T_i|$.

- **Cr**: Confidence of regularity. It indicates how well the time series can be interpreted by S_i or T_i, defined as $(\sum_i TS_i^w / \sum_i TS_i)/(1 - w/100)$, where TS_i^w belongs to the top w% points which are closest to $TS_i/R_i = S_i \times T_i$, and w is 80 by default in this work.

Note that **Cs** and **Cr** directly incorporate the confidence of **Slp**, **Amp**, and **Rp** into document feature representation. In addition, we also employ the *Timed PageRank* of Yu et al. [133] as our temporally-sensitive static-rank feature.

5.4.5 Query clustering features

The query importance $\mathcal{I}$ features (in Section 5.2.3) are used to cluster queries and assign the weights in each corresponding ranking function. We follow the approach taken by Bian et al. [19] and used the η top-ranked documents returned by a reference ranker (BM25 [107]) to generate our clustering features. We set the value of η to 15 in all our experiments. Once the pseudo-feedback documents are gathered, we compute the average value of each *ranking feature* over them and use the final mean value as a clustering feature. The feature importance is computed by training a reference RankSVM model for hybrid NDCG ($\gamma = 0.5$) on the training data sets.

5.5 Experimental Results

5.5.1 Introduction

So far we presented a learning to rank system framework, in which we leverage the freshness and relevance of search results, adaptive to queries' temporal characteristics in Chapter 5. We presented the evaluation platform which enables we conduct comparable experiments in Chapter 5.4. In this chapter, we focus on the evaluation of our proposed system framework. To do this, we ask the following research questions:

- How sensitive are the learned rankers to queries' temporal characteristics? Is it a good solution to train separate rankers to queries with different temporal characteristics? If not, which drawbacks are we able to mitigate?

Table 5.6: Non-temporal ranking features used by RankSVM in the CS-DAC framework and baseline methods. Body, title, heading and anchor-text fields are respectively represented by B, T, H and A.

Feature name	**Feature description**
Okapi(B)	Okapi BM25 score [107] for body-text.
RQT(B)	Ratio of covered terms in body-text.
RQT(H)	Ratio of covered terms in heading-text.
LM.JM(B)	body-text language modeling (Jelinek-Mercer) score [136].
LM.Dir(B)	Body-text language modeling (Dirichlet) score [136].
RQT(T)	Ratio of covered terms in title-text.
InNum	Number of inlinks.
TF(B)	Term frequency in body-text.
AvgNTF(B)	Average normalized TF in body-text.
LM.JM(T)	title-text language modeling (Jelinek-Mercer) score.
STFIDF(H)	Sum of term TFIDF in heading-text.
NumQT(A)	Number of covered terms in anchor-text.
MaxNTF(B)	Maximum normalized TF in body-text.
PR	PageRank score [23].
AvgNTF(T)	Avgerage normalized TF for title-text.
LM.Dir(T)	title-text language modeling (Dirichlet) score.
MxTFIDF(T)	Maximum term TFIDF in title-text.
MaxNTF(T)	Maximum normalized TF in title-text.
LM.Dir(H)	heading-text language modeling (Dirichlet) score
MaxTF(T)	Maximum query term frequency in title-text.
ATFIDF(T)	Average term TFIDF in title-text.
AvgTF(T)	Average query term frequency in title-text.
SumTF(T)	Sum of term frequency in title-text.
LM.JM(H)	heading-text language modeling (Jelinek-Mercer) score.
L(B)	Body-text length.
AvgTF(H)	Average query term frequency in heading-text.
SumTF(H)	Sum of term frequency in heading-text.

- CS-DAC: How superior is our system framework CS-DAC to baselines? How much benefit can we gain from each individual system component? Under what circumstances can we optimize relevance and freshness together? How much can we benefit from this for temporal and non-temporal queries respectively?

- As an extension of the second question, what are the best ways of optimizing multiple ranking criteria under the circumstances that these criteria may correlate in different ways?

In the remaining parts of this chapter, we explore these questions one by one.

5.5.2 Baseline Comparison

We start by comparing a set of baseline approaches, focusing on the first research question. Comparison among these baselines aim to explore (1) the sensitivity of ranking models with respect to queries with different temporal characteristics; and (2) whether optimizing for freshness and relevance can improve search quality on both of them. We then compare these baseline approaches with our CS-DAC in Section 5.5.4.

5.5.3 Baseline Approaches

These baseline approaches are as follows.

- Single ranker (SinR).

- Separate ranker training and selection (SepR).

- Over-weighting model [50].
- TopicalSVM [19].

In SinR, we train a single RankSVM ranker with all features. This could be regarded as a *weak* baseline that has no form of query categorization, and has been shown to perform more poorly than the other baselines in previous work [19, 62]. Nevertheless, we report its results because it represents one of the most common learning to rank architectures.

The SepR baseline is representative for the family of query-dependent loss function methods [19, 20, 62], in which the loss function is determined according to the temporal aspect of the query. Separate RankSVM rankers are trained for temporal and non-temporal queries, and each query is tested on the *correct* ranker for its type. Note that using the correct query type information—which is generally unavailable without manual effort—means that the performance numbers for this baseline are unaffected by potential query type misclassification, and therefore are overstated.

Dong et al. [50] investigated several techniques for ranking optimization with imbalanced amount of training data for freshness and relevance. Among their methods the *over-weighting* approach was most effective. The over-weighting model combines relevance and freshness labeled data to train a single ranker. This is similar to SepR except that the training pairs of the criterion with fewer labels are over-weighted. Dong et al. [50] used GBrank [138] as their ranking model. However, we modify the over-weighting loss function to RankSVM for consistency with the other methods in our experiments as follows:

$$\arg\min_{\omega,\xi_{q,i,j}} \frac{1}{2}\|\omega\|^2 + C\sum_{q,i,j}\xi_{q,i,j} \quad \textit{subject to} \tag{5.18}$$

$$\begin{array}{ll}
\forall y_i^q \succeq y_j^q : & \left\{ \begin{array}{ll} \frac{\alpha}{N_T}\omega^T X_i^q \geq \frac{\alpha}{N_T}\omega^T X_j^q + 1 - \xi_{q,i,j} & q \in \mathcal{Q}_T \\ \frac{1-\alpha}{N_N}\omega^T X_i^q \geq \frac{1-\alpha}{N_N}\omega^T X_j^q + 1 - \xi_{q,i,j} & q \in \mathcal{Q}_N \end{array} \right. \\
\forall_q \forall_i \forall_j : & \qquad \qquad \xi_{q,i,j} \geq 0
\end{array}$$

where $\mathcal{Q}_T$ and $\mathcal{Q}_N$ denote the sets of queries from Google Trends and MSN query log. N_T and N_N are respectively the number of preferential pairs of query-documents in each of those sets. α is a parameter that controls the balance of Google Trends queries vs. MSN queries, ranging over [0,1]. ω represents the feature weights within the ranking model.

Our last experimental baseline is TopicalSVM [19] which is the state-of-the-art in the family of divide and conquer techniques. TopicalSVM trains all rankers using a global loss function, and does not factorize the query-document importance $\mathcal{U}$ in contrast to CS-DAC.

We investigate the performance of these baseline approaches when trained for one of four optimization goals:

1. Relevance (Rel): The baselines are trained using relevance labels only.

2. Freshness (Fre): The baselines are trained using freshness labels only.

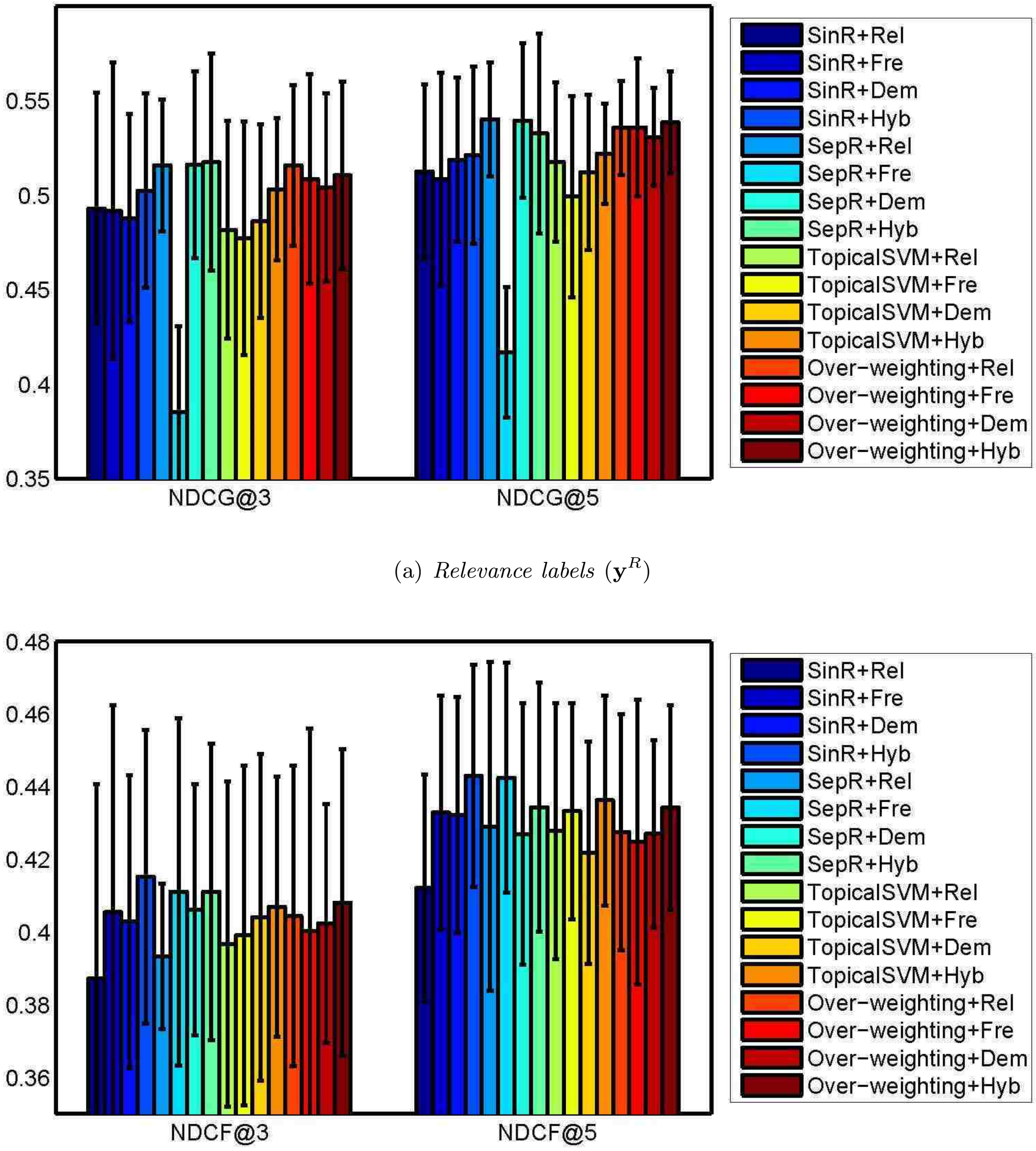
3. Hybrid labels (Hyb): The baselines are trained using hybrid labels (Equation 5.3).

4. Demoted labels (Dem): Dong et al. [50, 51] suggested *demoting* the the relevance grades of outdated documents. They suggested that if a document is somewhat outdated, then its relevance label should be demoted by one grade. For totally outdated documents the relevance labels are demoted by two grades. We followed the same strategy to compute our demoted labels. In essence, this is a special case of hybrid labeling.

The final results of each optimized ranker are evaluated separately for freshness and relevance using NDCG with corresponding labels. In all our experiments we run 5-fold cross-validation in which the first three folds are used for training, and the remaining two folds are used for validation and testing. The number of ranking functions (clusters) in CS-DAC and TopicalSVM to are set to three ($k = 3$), since preliminary results demonstrate CS-DAC and TopicalSVM perform the best when $k = 3$ and $k = 4$ (slightly outperforms the case when $k = 3$) respectively.

Performance Comparison

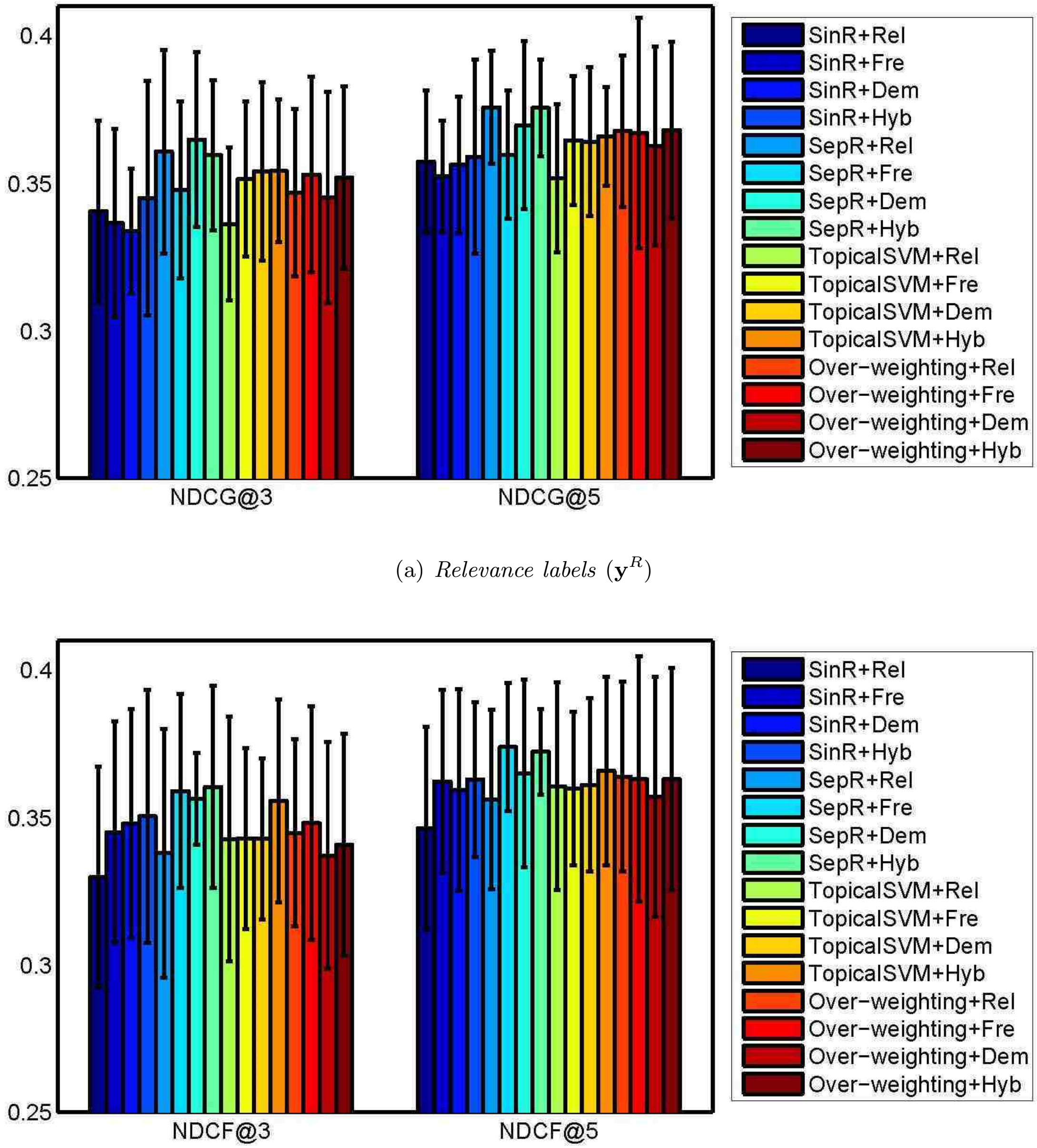
Figure 5.5 shows the performance of baseline techniques on the *non-temporal* query set (sampled from the MSN logs). As expected, when evaluating using the relevance labels ($\mathbf{y}^R$), it is more effective to optimize for relevance (Rel) rather than freshness (Fre). Similarly, optimizing for freshness produces results that have better NDCF values. The methods optimized for demoted (Dem) and hybrid (Hyb) labels consistently outperform those that are optimized for either freshness or relevance. The results also suggest that



(a) *Relevance labels* ($\mathbf{y}^R$)

(b) *Freshness labels* ($\mathbf{y}^F$)

Figure 5.5: Ranking performance of baseline systems on relevance (top) and freshness (bottom) for the *non-temporal* query set. Error bars are the standard deviations of performance across five cross-validation folds.



(a) *Relevance labels* ($\mathbf{y}^R$)

(b) *Freshness labels* ($\mathbf{y}^F$)

Figure 5.6: Ranking performance of baseline systems on relevance (top) and freshness (bottom) for the *temporal* query set. Error bars are the standard deviations of performance across five cross-validation folds.

our hybrid labels are better for improving both relevance and freshness compared to the demoted labels of Dong et al. [50, 51]. Among the baselines, SinR has overall the poorest performance which is consistent with previous observations [20]. TopicalSVM, over-weighting and SepR show similar effectiveness while the latter might be considered marginally better—not surprising given that we use correct query type information in SepR.

We repeat the analysis on the *temporal* query set and the results are illustrated in Figure 5.6; as in the previous experiment, SinR has the lowest performance on both sets of labels while the other methods show similar effectiveness. Compared to the experiments on the non-temporal query set, there is less variation in performance when optimized for different types of labels. Our investigations revealed that this is due to high correlation between relevance and freshness labels on the temporal set. The Pearson's correlation between relevance and freshness labels on the temporal query set is 0.912 ± 0.004, statistically significantly higher than 0.429 ± 0.021 for the non-temporal set.

Based on the summarized results, we choose hybrid labels for training rankers for investigating the following research questions. We also drop SinR as it consistently showed inferior effectiveness compared to all other methods.

5.5.4 CS-DAC: Performance Comparison

We now focus on the second research question, investigating the effectiveness of CS-DAC. We start by comparing it with baselines approaches, in terms of freshness and relevance of

Table 5.7: Freshness comparison on the *temporal* (top) and *non-temporal* (bottom) query sets. All methods are trained using the hybrid labels and the evaluation is based on the freshness ratings ($\mathbf{y}^F$). Symbols †, §, and ‡ respectively denote statistically significant differences according to a single-tailed student t-test ($p - value < 0.05$) over the SepR, TopicalSVM and Over-weighting baselines.

	Temporal Queries (Google Trends)			
	NDCF1	NDCF3	NDCF5	NDCF10
SepR	0.378	0.360	0.372	0.408
TopicalSVM	0.365	0.355	0.365	0.402
Over-weighting	0.340	0.348	0.363	0.404
CS-DAC	0.398‡	0.364	0.376	0.411
CS-DAC($\mathcal{U}$)	0.416†§‡	0.379‡	0.388	0.400
	Non-Temporal Queries (MSN logs)			
	NDCF1	NDCF3	NDCF5	NDCF10
SepR	0.348	0.411	0.434	0.475
TopicalSVM	0.355	0.408	0.430	0.485
Over-weighting	0.335	0.408	0.434	0.480
CS-DAC	0.427†§‡	0.454†§‡	0.473†§‡	0.510§‡
CS-DAC($\mathcal{U}$)	0.452†§‡	0.466†§‡	0.488†§‡	0.527†§‡

search results. We next explore the effectiveness of each individual component of CS-DAC.

Comparative Performance on Freshness

We use NDCG with freshness $\mathbf{y}^F$ labels (NDCF [51]) to compare the performance of CS-DAC with the baselines on both temporal (Google Trends) and non-temporal (MSN logs) query sets. We report the results for CS-DAC in the presence and absence of the query-document importance factor ($\mathcal{U}$) described in Equations 5.4 and 5.9. We respectively refer to these two versions as CS-DAC($\mathcal{U}$) and CS-DAC.

Table 5.7 includes the NDCF results on both query sets. The over-weighting baseline performs worst. This is not surprising given that over-weighting is originally designed for scenarios with imbalanced training data [50], and the fact that it does not leverage any type of query classification or clustering. Consistent with the observations in the previous section, SepR and TopicalSVM produce similar results on the temporal queries, while they are both outperformed by CS-DAC. Introducing the $\mathcal{U}$ factor leads to further improvements in performance particularly at higher cutoffs. On non-temporal queries, TopicalSVM and SepR and over-weighting show similar effectiveness while CS-DAC consistently outperforms all baselines significantly. It is interesting to observe that CS-DAC improvements over the baselines are larger on the non-temporal query set. This can be explained by two reasons: (1) the documents returned for temporal queries tend to be fresher on average than those returned for the non-temporal ones, and (2) the high correlation between relevance and freshness labels in this set leads to more effective learning by reducing impact of potential noise in clustering and hybrid labels.

Comparative Performance on Relevance

We run a similar analysis, and compare the NDCG values of different techniques as measured by the relevance labels ($\mathbf{y}^R$) in Table 5.8. For non-temporal queries, the CS-DAC results are marginally better than the baselines, although none of the differences are statistically significant. On the temporal query set, SepR has the edge over the other baselines while CS-DAC outperforms the three of them at all cutoff values. Adding the $\mathcal{U}$ factor significantly improves the results for NDCG@1 and NDCG@10. As in the NDCF numbers on this query set, the NDCG values could be also affected by the high correlation between freshness and relevance.

Table 5.8: Relevance comparison on the *temporal* (top) and *non-temporal* (bottom) query sets. All methods are trained using the hybrid labels and the evaluation is based on the freshness ratings ($\mathbf{y}^R$). Symbols †, §, and ‡ respectively denote statistically significant differences according to a single-tailed student t-test ($p - value < 0.05$) over the SepR, TopicalSVM and Over-weighting baselines.

	Temporal Queries (Google Trends)			
	NDCG1	NDCG3	NDCG5	NDCG10
SepR	0.373	0.359	0.375	0.411
TopicalSVM	0.342	0.354	0.365	0.408
Over-weighting	0.355	0.351	0.368	0.411
CS-DAC	0.385	0.365	0.377	0.417
CS-DAC($\mathcal{U}$)	0.401†‡	0.375	0.389	0.426†
	Non-Temporal Queries (MSN logs)			
	NDCG1	NDCG3	NDCG5	NDCG10
SepR	0.481	0.517	0.532	0.562
TopicalSVM	0.490	0.508	0.521	0.566
Over-weighting	0.476	0.510	0.538	0.570
CS-DAC	0.493	0.520	0.541	0.574
CS-DAC($\mathcal{U}$)	0.509	0.522	0.541	0.574

Deeper Analysis

We showed that our CS-DAC method could significantly improve both freshness and relevance of the results compared to state-of-the-art baselines in Section 5.5.4 and 5.5.4. In this section, we investigate the impact of random walk smoothing in improving the query-document factor $\mathcal{U}$ for training. We also compare CS-DAC and the baselines in terms of hybrid NDCG by assigning various weights to relevance and freshness. Finally, we report the most effective features according to our experiments for ranking temporal and non-temporal queries.

Smoothing query-document importance We described earlier how original query-document importance values can be smoothed by random walk, where the probability d of random jumping can be tuned during training and validation. Figure 5.7 shows how choosing different fixed values for d may affect the results. On the non-temporal query set, different degrees of smoothing have little advantage over no smoothing ($d = 0$). On the temporal query set however, random-walk helps to smooth inter-label dependencies, and hence improves the results on both freshness and relevance.

Hybrid labels for evaluation In Section 5.5.2 we showed that training for hybrid NDCG ($\gamma = 0.5$) was effective for improving both freshness and relevance. Here, we provide the evaluation results on hybrid NDCG, the metric we used for optimizing the ensemble ranking. Although we used $\gamma = 0.5$ for training, we report the evaluation results for different values of γ in Figure 5.8 to account for scenarios where freshness and relevance are weighted differently. The results are consistent with our previous experiments; CS-DAC outperforms the baselines, and the weighting between freshness and relevance is less important for temporal queries. Increasing the γ value grows the overall hybrid NDCG almost monotonically because the relevance-based NDCG values are generally greater than those computed based on the freshness labels. It is worthwhile pointing out that this observation does not suggest that ranking performance benefits the most when only optimizing for relevance.

Feature Group	Rank	Feature Importance
Slp(*)	32.00 ± 11.94	22.77 ± 8.20
Amp(*)	38.83 ± 18.68	21.67 ± 20.06
Rp(*)	26.33 ± 9.60	26.86 ± 7.01
Cs(*)	47.83 ± 10.66	11.17 ± 7.00
Cr(*)	50.08 ± 11.96	9.53 ± 8.49

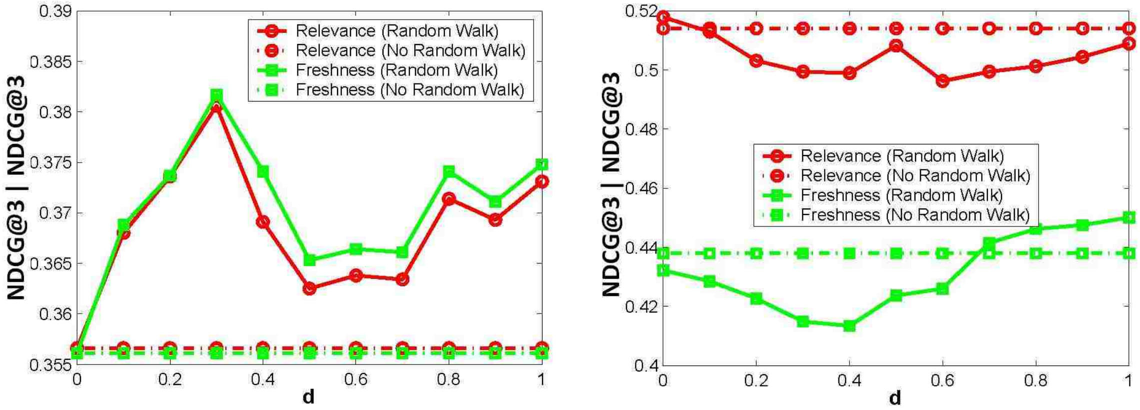
Table 5.9: Feature study for *TempQueries* (Rank $\in [1, 64]$, Feature Importance $\in [1, 100]$).

Feature Group	Rank	Feature Importance
Slp(*)	55.16 ± 2.91	4.14 ± 1.04
Amp(*)	28.66 ± 8.25	14.05 ± 2.58
Rp(*)	37.33 ± 15.61	10.85 ± 5.73
Cs(*)	52.5 ± 6.31	5.16 ± 2.63
Cr(*)	44.08 ± 15.52	8.07 ± 5.9

Table 5.10: Feature study for *NonTempQueries* (Rank $\in [1, 64]$, Feature Importance $\in [1, 100]$).

Feature Analysis CS-DAC relies on several temporal and non-temporal features for query clustering and document ranking. We examined all cross-validation folds to find the features that are assigned with highest weights during training. Among the temporal features, the confidence values for the seasonality CS(τ), and regularity Cr(τ) of STL decompositions were generally the most effective. Furthermore, the features generated from the time-series decomposition of changes in anchor-text and inlinks were more successful than those similarly produced based on other fields (e.g., title, body, heading).

Among the non-temporal features, BM25 and language modeling scores had the highest weights and were most effective when computed over the body and title text. We report the ranks and the importance (the normalized version with the sum equal to 100) of each group of time series based features within the whole feature set in Tables 5.9 (temporal queries) and 5.10 (non-temporal queries) respectively.



(a) *Temporal queries*

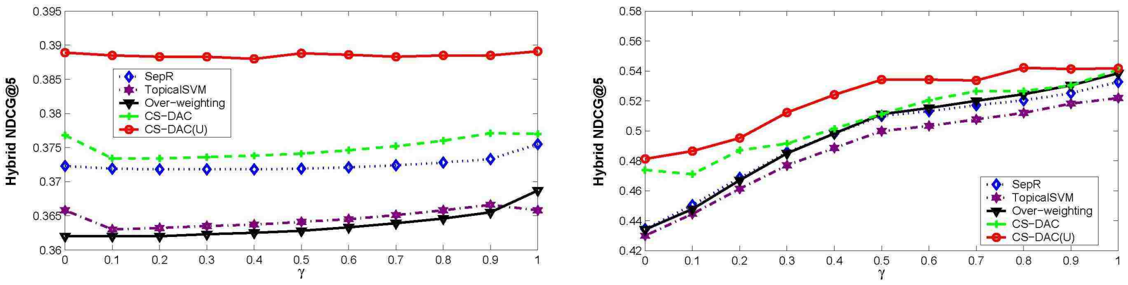
(b) *Non-temporal queries*

Figure 5.7: The impact of changing the random jump probability d during smoothing of the query-document importance values $\mathcal{U}$. The results are evaluated on temporal (left) and non-temporal (right) queries using both relevance and freshness labels.

5.5.5 Multiple Ranking Optimization

To answer the third research research question, we now focus on investigating how the gain of bi-criteria ranking optimization varies with the bi-criteria correlation and the optimization capability. To avoid data bias, we conduct experiments on two data sets.

We use syntactic data to simulate the process in which pairwise ranking models generate search results. Our dataset consists of 21 subsets, with each composed of 1000 simulated bi-criteria rating scores that have a fixed score correlation from -0.9 to 0.9 with a step size 0.1. Three pseudo-classifiers (i.e., simulated rankers) are used to generate preferential score pair relationships based on each aspect of bi-criteria (i.e., producing baseline results) and the *hybrid* labels in Section 5.3.1 respectively. To incorporate optimization capability variance, we exploit a probability threshold (ranging from [0.1,0.9] with step



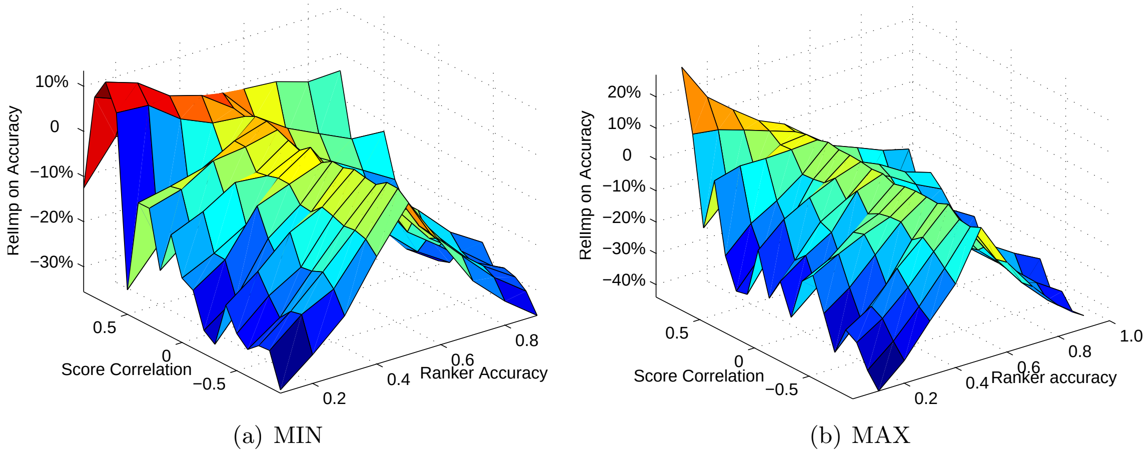
(a) *Temporal queries*

(b) *Non-temporal queries*

Figure 5.8: Hybrid NDCG5 values for different values of γ (in Equation 5.12) on the *temporal* (top) and *non-temporal* (bottom) query set. Similar trends were found for NDCG at different cutoff values.

size 0.1) to control the chance that pseudo-classifiers generate correct pair relationships, denoted as *ranker accuracy*. For instance, if the *ranker accuracy* is 90%, the generated pair relationship has 90% chance to be consistent with the ground truth. The gain of bi-criteria ranking optimization is measured by *RelImp* based on the percentage of correctly classified preferential score pairs (i.e., *RelImp on accuracy*).

Figure 5.9 shows the minimum relative improvement on preferential pair classification accuracy for MIN and MAX as the bi-criteria correlation and *ranker accuracy* vary. Preliminary results demonstrate that the trends of others typically fall in between. The bi-criteria optimization brings benefits when the bi-criteria correlation is highly positive and *ranker accuracy* is low. When the ranker accuracy is high, bi-criteria optimization has negative impact on performance. It is not surprising given that it actually incorporates more inaccurate optimization objectives, and this can be mitigated with the increase of bi-criteria correlation.

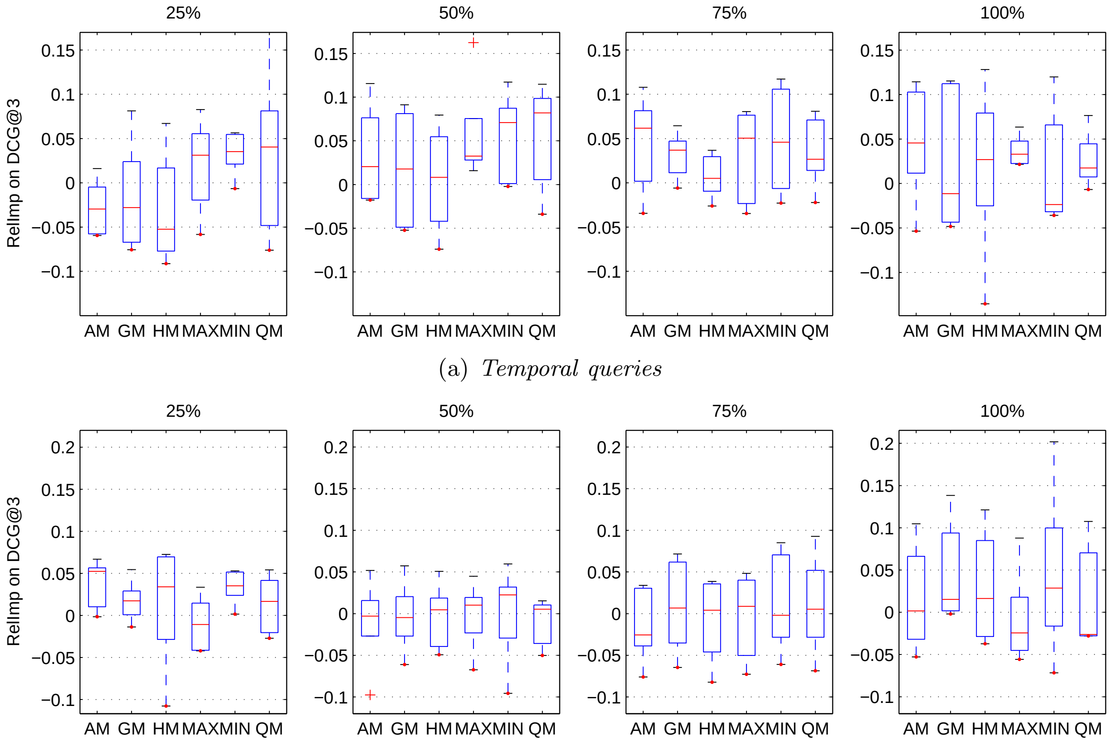


(a) MIN (b) MAX

Figure 5.9: The minimum relative ranking improvement on accuracy based on MIN and MAX *hybrid* labels under the variance of bi-criteria correlation and ranker accuracy.

To further investigate the effect of bi-criteria ranking optimization on real search scenarios, we conduct the comparable experiments on our evaluation platform. Given the freshness and relevance have stronger positive correlation for temporal queries, rather than non-temporal queries, the relative ranking improvements are compared based on these two types of queries.

Figure 5.10 shows the average and standard deviation of RelImp on DCG@3 [71] across five fold cross-validation for *temporal* and *non-temporal* query sets respectively. By using the different top $k\%$ effective ranking features that are selected by a reference model (a RankSVM model in this work) based on training data, we incorporate the influence of ranker effectiveness into the sensitivity study on the gain of bi-criteria ranking optimization. The results confirm our previous observations on simulated data and demonstrate



(a) *Temporal queries*

(b) *Non-temporal queries*

Figure 5.10: The average and standard deviation of *RelImp* on DCG@3 across five folds for the *temporal* (top) and *non-temporal* (bottom) query sets by using the top 25%, 50%, 75% and 100% (all) effective ranking features. (AM: arithmetic mean; GM: geometric mean; HM: harmonic mean; MAX: maximum; MIN: minimum; QM: quadratic mean.)

that (1) RelImp is more sensitive to hybrid labels and ranker effectiveness when the correlation between relevance and freshness is highly positive (i.e., the *temporal* query set); and (2) bi-criteria ranking optimization can bring more benefits under highly positive bi-criteria correlation.

5.6 Summary

From Chapter 5 to Chapter 5.5, we proposed a learning to rank approach (CS-DAC) for optimizing for relevance and freshness simultaneously, built an evaluation platform, and showed the effectiveness of CS-DAC on it. We extended the state-of-the-art in divide and conquer ranking [20] by adding two new key elements; first, instead of optimizing for relevance labels, we generated and used hybrid labels based on relevance and freshness grades. Second, we introduced a new query-document importance factor ($\mathcal{U}$) that allows each ranker to set different importance to relevance and freshness. Compared with traditional metasearch engines, divide-and-conquer ranking frameworks generate merged ranking lists on the model level instead of the result level. It enables automatic identification of effective ranking features for individual type of queries. Our experiments on a large web archive demonstrated that CS-DAC can improve both relevance and freshness compared to existing baselines.

We studied the correlation between relevance and freshness grades, and its implications on the training effectiveness. Our results revealed high correlation between relevance and freshness labels in temporal queries, suggesting that the choice of document labels is less important for training on that set. We modeled document importance by the likelihood of visiting each unique hybrid label, and surprisingly found that it can improve ranking performance, especially for temporal queries. However, in what way that such document weighting strategies influence ranking performance is still unclear. We will leave it as future work.

Our work can be considered as the simplest form of *multi-objective (multiple-criteria) optimization* [115], where multiple objective functions (freshness, relevance) are combined to form a single optimization goal (hybrid labels). These kinds of *aggregated* functions require the weight of each objective to be known in advance (γ in our case), and are incapable of finding all optimal solutions. Deploying more sophisticated multi-objective optimization techniques may lead to more significant improvements in relevance and freshness. Further work includes adopting other learning to rank architectures such as boosted decision trees [125] for multi-objective optimization of freshness and relevance.

Chapter 6

Conclusions and Future Work

So far we have presented how we incorporate the information of web dynamics into three search components, i.e., anchor text representation enhancement for retrieval, web authority estimation, and machine learning based ranking systems from Chapter 3 to Chapter 5. In this chapter, we conclude this dissertation. We start by recapping the threads of this dissertation, and then focusing on each individual part, we present its impact on other research directions or industry applications. We next analyze the deficiencies of this dissertation. We end by suggesting a few future research directions.

6.1 Recapitulation

The collective activities from billions of web users result in a dynamic web. The creation, revision and removal of web pages and hyperlinks characterize human's daily lives, suggest the authority and value of user-generated content, and reflect the temporal properties of users' information needs expressed through queries. Unfortunately, it is widely believed that the current commercial search engines fail to utilize much historical information concealed in web dynamics when generating document rankings for answering user queries [3]. Therefore, the purpose of this thesis is to explore effective ways of utilizing web dynamics to improve search quality. Given that query temporal characteristics imply the importance of search freshness on users' satisfaction, we emphasize the freshness and relevance of search results in the scope of this thesis.

In Chapter 3, we proposed an anchor text weighting strategy to enhance the representation of page content for improving search relevance. The essential idea is to aggregate anchor text weights at different time points, and this enables the state of anchor text (and its associated hyperlinks) at different time periods to influence each other. We observed that such a stabilized anchor text representation effectively improves the relevance of search results compared with the one only based on a single web snapshot.

Enlightened by the practical success of smoothing anchor text weights at different time points for anchor representation, we consider whether it also helps other applications that depend on analyzing the link structure. Web authority estimation is one such application. In Chapter 4, we proposed a temporal random surfer model for estimating web authority. This approach allows authority flows to distribute between web snapshots at different time points, and so the link structure at different time periods mutually influence the final estimation of web authority. The advantage is that the approach mitigates the deficiency that traditional web link analysis approaches unfairly favor old pages. Experimental results demonstrate that this approach is superior to the representative link analysis method—PageRank, and several state-of-the-art link-based algorithms that incorporate temporal information of web pages and hyperlinks, in terms that it significantly improves the freshness of search results without hurting search relevance for temporal queries.

Observing that both freshness and relevance can improve over baselines for temporal queries, we consider the interrelationship between freshness and relevance. Is there a positive correlation between the two ranking criteria? If so, is that correlation sensitive to query types? If so, can we utilize the correlation between freshness and relevance in optimizing ranking functions? These questions motivate the work of learning to rank for freshness and relevance, in which freshness and relevance are simultaneously optimized depending on queries' temporal characteristics. We present the design, implementation and evaluation of this system in Chapter 5. The observation is that freshness and relevance have high positive correlation with each other for temporal queries, but not for non-temporal ones. When optimizing freshness and relevance, adapting their trade-off according to queries' temporal characteristics, performance on both ranking criteria improved significantly.

6.2 Impact

We recapped the main parts of this dissertation, and now present the impact of our work on other research directions and/or industry applications.

Mining anchor text trends for retrieval. We proposed an anchor text weighting strategy that depends on the creation time of hyperlinks. The idea of propagating anchor weights over the time axis, letting such influence decay over time, has been shown effective for improving the field retrieval model BM25F in our experimental settings. While we focus on anchor text representation, a similar idea can be extended to other applications. The essential reason is that anchor text provides information complementary to target web pages, and these are reasonably viewed as part of the target page content.

- In professional search (such as people search in LinkedIn), the attributes of a person's profile at different time points, e.g., membership in an organization one year ago or five years ago, may influence her relevancy with respect to that organization. A "decayed" profile may better represent her characteristics.

- In academic search (such as finding the most influential researchers), the publication time affects the estimation on how active a researcher is, given a specific querying time. Decayed importance on older papers helps better quantify researchers' activity, when we use their publications to characterize them.

- In product rating systems (such as Amazon's product review system), customers usually refer to previous comments and ratings first, and then proceed with their purchase, and then optionally leave their own comments and/or ratings. Emphasizing more recent comments or ratings but demoting much earlier ones (e.g., three years ago or five years ago) helps better represent the preference of customers at the current time period.

- In microblog search (such as Twitter search), users typically generate consistent posts or tweets in terms of their topicality and/or polarity. Therefore, when utilizing such consistent content to enhance the representation of a target post or tweets, it is reasonable to demote temporally farther content more than temporally closer content.

This approach has the potential to be applied into real-world IR systems. Recall that the approach includes three steps. The time complexity of computing the anchor text weights for target pages within multiple snapshots (the first step) is $O(N_pe)$, where N_p is the number of past snapshots, and e is the number of edges. The time complexity of predicting future anchor text weights (the second step) is $O(N_fa)$, where N_p is the number of future snapshots, and a is the number of unique anchor-document pairs. The time complexity of anchor weighting propagation (the third step) is $O(Na)$, where $N = N_p + N_f$ is the total number of snapshots. Practically, commercial search engines crawl the web periodically, roughly less than one month per crawl. Each time when the newest version of the web is achieved, we first perform compute the anchor weights based on the current step ($O(N_pe)$), and then update the prediction of future anchor weights ($O(N_fa)$). We finally update the influence from anchor propagation ($O(N_f a)$). Thus, the proposed approach is no more cumbersome than existing efforts.

Incorporating web freshness into page authority estimation. We proposed a temporal random surfer model for estimating web page authority. It incorporates web maintenance activities into web freshness, controlling the distribution of authority flows between pages. It mitigates the deficiency that traditional link based ranking algorithms favor old pages. In addition to the web search domain, a similar idea also inspires other research areas.

- In social network analysis, the importance of social players is typically estimated from their past activities. When quantifying how much a social player influences others, her interaction with others serves as an evidence of her importance. Emphasizing more recent interactions but demoting earlier ones (i.e., emphasizing the "freshness" of social interactions) helps better estimate social player importance.

- In publication search (such as finding the most influential papers), the importance of a paper is usually indicated by its citation number without taking into account when this paper was published. Decaying the influence of older citations but promoting the contribution from more recent ones (i.e., emphasizing the "fresher" citations) tends to provide more reasonable publication importance estimation.

- In computational advertising, one important problem is to target potential buyers. Researchers usually draw from analyzing users' past on-line activities. Especially, the transitions between diverse types of activities (e.g., `browsing`→`carting`, `carting`→`purchasing`) have different contributions on inferring users' purchase intent. In addition, the time points at which these transitions occur also significantly influence users' purchase prediction. Emphasizing the critical activity transitions within more recent time periods (e.g., emphasizing the "freshness" of activity transition) is likely to improve the prediction accuracy.

The approach has the potential to be applied in real-world search engine systems. First, the influence of web maintenance activities between successive time points is accumulated with the time complexity $O(e)$ (in-link freshness) and $O(n)$ (page freshness) respectively, where e (n) is the number of edges (pages) per snapshot. Second, the incremental in-link and page freshness by considering propagation has the time complexity $O(Ne)$ if we utilize the power iterative method, where N is the number of iterations before the convergence, e is edge number per snapshot. Third, the accumulative web freshness scores are then computed with the time complexity $O(n)$. These three steps can be done periodically per several crawls. Fourth, once we achieve the page and inlink freshness for all the pages at all time points, we next operate a temporal random surfer model, computing (1) the probability that a web surfer reaches one page, with the time complexity $O(NTN'e)$; and (2) the expectations of the surfer staying on one page, with the time complexity $O(TN'n)$. T is the number of snapshots within a temporal window, N' is the number of snapshots,

n is the number of pages per snapshot, and N is the number of iterations before the convergence assuming we use the power iterative method. Thus, the scale of data, i.e., the nodes and hyperlinks, processed by our proposed approach is approximately proportional to the number of snapshots, when compared with existing efforts that only utilize one (the current) web snapshot.

Learning to rank for freshness and relevance. We proposed a learning to rank system that optimizes freshness and relevance, adapting to queries' temporal characteristics. We observe that (1) temporal and non-temporal queries have different correlations between freshness and relevance; and (2) the benefits from optimizing bi-criteria depend on the correlation between these two criteria. While our system is proposed for ranking web documents, the ideas that involve optimizing for multiple criteria and building up adaptive learning systems can be extended beyond web search.

- In news search, freshness of search results is especially important. For each query, the correlation between freshness and relevance is sensitive to the time points at which users issue queries. While not a real-time system, our prototype helps handle the queries with diverse freshness-relevance correlation in an adaptive way.

- In information filtering and recommender systems, users rate items based on multiple criteria. These criteria correlate with each other depending on user profiles. Optimizing multiple criteria adaptive to users potentially benefits the prediction models of users' overall preferences on items.

- In search advertising, ad rankers are driven to optimize for revenue. However, over-optimizing for revenue hurts users' search experience, and so does harm to the revenue in the long term. Therefore, leveraging the trade-off between (short-term) revenue and users' search experiences in training ad rankers is not a trivial task and can potentially benefit from our system.

The approach can be applied in real-world web search systems. The time complexities in training and test stages are different. While the time complexity of the SVM algorithm depends on the actual technique for solving the quadratic convex optimization, it is well believed that the typical complexity of SVM is $O(n^2m)$, where n is the number of training instances, and m is the number of features. Here, our instances are preferential query-document pairs. For each ranker, the trade-off between freshness and relevance is learned using line search, and so the time complexity of our approach in training stage is about $O(Nkn^2m)$, where N is the number of rankers, and k is the number of trials within line search. In the test stage, the time complexity is linear with the feature size.

6.3 Caveats

We presented the impact of this dissertation; we now analyze the limitation and deficiencies of the dissertation projects respectively.

Mining anchor text trends for retrieval. While we empirically demonstrated that our proposed anchor text weighting strategy can enhance the baseline only using a single web snapshot, our method may suffer from the following limitations.

- We model the influence of hyperlink creation on anchor weighting. Other types of maintenance activities on hyperlinks, such as update and removal, are inevitably not modeled due to our experimental conditions, i.e., based on the current snapshots, we track backwards within the archival corpus to figure when each individual hyperlink was created. These additional activities could be very helpful.
- The existing archival web pages only cover a small portion of the historical web, which causes a large amount of missing anchors (only 2.57% anchors have archival copies in our data set) and thus limits the application of our proposed method. This also suggests a limitation to the usefulness of external archival web resources with respect to current search engines.
- The crawling policies used to collect the archival web page copies might not accurately record the history of web activities. For example, the updates of web pages are more frequent than crawling frequency. Therefore, inaccurate web maintenance activities may do harm to the accuracy of our proposed models.

One may consider how adversaries can utilize the proposed approach to hurt ranking performance. We argue that this approach is robust in the sense that anchor text weights are influenced by a series of their states on anchor text popularity and link structures in the past. One possible weakness is that we use linear regression for modeling the trends of anchor text weights over time in this work. Adversaries might consider changing the ways at which anchor weights for certain pages evolve. In addition, the evolution of anchor weights in general may be dynamic, i.e., it may be sensitive to the time periods, given that the motivation and behaviors of web creators maintaining web content may evolve over time. Therefore, the effectiveness of our approach may suffer from the inaccurate modeling on anchor weights.

Incorporating web freshness into page authority estimation. While the evaluation demonstrates that our proposed temporal random surfer that incorporates web freshness outperforms state-of-art link-based ranking algorithms, our approach may suffer from the following deficiencies.

- The crawling frequency influences the accuracy of web maintenance activities. In this work, we processed the corpus by removing pages with fewer than 5 snapshots. This reduces the size of our corpus from 158M unique pages to 3.8M unique pages. Even so, the historical copies of web pages are still sparse given that the time span is from January 2000 to April 2007, with month-based granularity. Therefore, the inaccurate web maintenance activities may do harm to the accuracy of our proposed models.

- The judgments on freshness for temporal queries may not be accurate. We choose to use April 2007 as our interested time point for ranking evaluation. We face the following difficulties: (1) human judges may not be able to remember the events closest to our interested time point, given that April 2007 is far from now; (2)

the unclear definition of freshness, i.e., it can be either how freshness the content recorded on the page is or how recent the last modified time is by page maintainer, etc.; (3) it is hard to obtain the freshness judgments for the web pages that do not contain an obvious time clue.

One may consider how adversaries can utilize the proposed approach to hurt ranking performance. We realized that our approach has the risk of promoting link spam. The in-link activities boost the in-link freshness of web pages, and so the search systems tend to favor the pages whose in-link popularity increases suddenly. As a result, the pages with link spam may be promoted. However, we also quantify another aspect within web freshness, i.e., page freshness, from the activities on the pages themselves and their outgoing linked web resources. We use the correlation between page and in-link freshness as a confidence to indicate the probability of the search systems favoring certain web pages, and so mitigate the negative effect from the potential link spam to some extent. Even so, it is worthwhile pointing out the risk of our approach on promoting link spam.

Learning to rank for freshness and relevance. While our proposed learning to rank system that optimizes freshness and relevance adaptive to queries' temporal characteristics can improve ranking performance on both freshness and relevance for queries with diverse temporal characteristics, it may suffer from the following deficiencies.

- The judgments on freshness for non-temporal queries may not be accurate. First, it is hard to build up a quantitative connection between users' search experience and the freshness of search results, i.e., the importance of freshness for non-temporal queries is doubtful. Second, the web pages for answering non-temporal queries typically do not contain time-sensitive information, and so how to rate the freshness score for these pages is still a controversial question in the research community.

- The granularity of temporal queries: we separate the queries used in ranking evaluation into temporal vs. non-temporal ones. The main criterion for temporal queries is whether there exists burstiness when keeping track of query popularity within search logs after 2008. Finer-grained query temporal characteristics is neglected, e.g., whether queries are seasonal or breaking-news, etc.
- Our methodology operates on the assumption that queries' temporal characteristics are differentiable. Representative research papers did demonstrate that commercial search engines are capable of classifying queries according to their temporal characteristics. Chien and Immorlica [32] suggested the similarity between search queries can be better inferred from the correlation between their query volume. Following this spirit, a few research works have been proposed to benefit search applications. Alfonseca et al. [4] used query temporal similarity to improve the applications of both query suggestion and query categorization. Shokouhi and Radinsky [114] predicted the future query frequencies to benefit query auto-completion systems. Jones and Diaz [74] classified queries according to their temporal distributions using the timestamps of pseudo-feedback documents, and suggested that the queries with different temporal characteristics indicate their diverse search difficulty.

- Our data-oriented assumptions may be too strong. We assume that (1) queries' temporal characteristics is long term, i.e., whether a query is temporal or non-temporal is consistent before and after 2008; and (2) queries' temporal characteristics are not sensitive to location, given that the archival web corpus is in `.ie` domain and the search log is US based.

6.4 Future Work

We reviewed the limitations and deficiencies of this dissertation; in this section we suggest a few future directions. These directions fall into two categories: (1) improving web search systems and search quality; and (2) analyzing web dynamics to benefit the applications of social media.

Improving Search Systems. Future work that aims to improve search systems includes:

- **The connection between search freshness and users' satisfaction**. Freshness has been recognized as one of the important facets within search quality, especially for temporal queries. How important search freshness is for a given query, when compared with relevance, is sensitive to the time points at which users issue that query. This suggests that when connecting users' search satisfaction with the freshness of search results, the strength of their connection depends on queries' temporal characteristics. For now, how much emphasis we should put on freshness in quantifying users' satisfaction, and how to determine the relative importance of freshness for each query, are still open questions in the research community.

- **Collecting groundtruth: click-through data versus freshness judgments**. Freshness judgements are difficult to obtain. First, freshness scores of web pages for answering a given query is sensitive to our time point of interest for conducting ranking evaluation. It is difficult to gain large scale freshness judgments in very short time periods. Second, the interpretation of page freshness is sensitive to topicality of page content. For example, a three-day-old story that keeps track of a tsunami is stale, while the week-old report introducing a recently appointed school president is still fresh. Based on the above two reasons, commercial search engines mine the click-through rates of URLs from search logs more often, and use that to guide the groundtruth that indicates the overall relevance of query-document pairs (i.e., assume the click-through implicitly conceal users' overall preference, leveraging all search quality facets). Its advantage is that it is cheap to gain large-scale groundtruth in a prompt manner, and so more convenient to keep track of how users' preference drifts over time. However, it also has its deficiencies, i.e., the position biases prevents one from objectively interpreting users' preference, since users are inclined to click the URLs at the very top positions. The comparison between these two ways of collecting ranking groundtruth suggests that (1) freshness judgments potentially enables us to investigate finer-grained search quality facets and how these facets interact with each other, and how such interaction evolves over time; (2) clickthrough data enables us to achieve large-scale groundtruth in a prompt way, but may suffer from the deficiency of position bias; in addition, the finer-grained search quality facets are not differentiable. Therefore, the future work might consider interpreting the ranking evaluation by using the groundtruth from different sources, and how they complement each other. In the worst case, when ranking evaluation results by using these two ways for collecting groundtruth contradict each other, how we can interpret the superiority of the compared ranking algorithms, etc.

- **On-line ranking functions and temporal ranking features**. The freshness judgements on query-document pairs are sensitive to time points at which users issue queries. This suggests two possible directions of improving ranking performance: (1) training an on-line ranker that updates in real-time, leveraging the dynamic tradeoff between freshness and relevance; (2) incorporating the temporal features that characterize the query-document pairs for training on-line rankers in real time.

- **Hurt diversity?** When only focusing on freshness and relevance, we may inevitably neglect other ranking criteria, especially diversity which quantifies the richness of information that can be delivered from a document ranking list. Future work might investigate whether we could optimize for freshness and relevance, but at the same time leverage the diversity of search results.

- **Freshness label sparsity**. Only around 10% of queries are temporal queries. When training the rankers for answering temporal queries, one challenge is that the number of training instances is not enough to guarantee to train a confident ranker. Future work following this direction would focus on solving the problem of instance sparsity in the ranker training process.

Improving Social Media Applications. Social media web sites, such as Twitter, Facebook, LinkedIn, Myspace, etc., provide a variety of platforms to facilitate web users to interact with each other. Two representative research directions are social content and link (connection) recommendation and topical analysis.

- **Social content and link (connection) recommendation services** provide recommendations of items, users, or user generated content to web users. Research work on traditional information filtering and recommender systems only draw from the similarity between the users profiles and the recommended item candidates, without considering the recommendation from similar users. Collaborative filtering (CF) mitigated this problems by referring the preference of similar users. Representative approaches are matrix factorization and neighborhood based CF. Koren [79] found that the users' rating scores drift over time, and so incorporate a temporal factor into these two representative collaborative filtering approaches, and achieve significant improvements over the ones without. Nowadays, collaborative filtering approaches have been shown effective on recommendation tasks in industry. Modeling the dynamics of user interests continues to be an active area for research.

- **Topical analysis of on-line user generated content** is one way of understanding users' trace on the web. Traditional topical models do statistics on term occurrence within web documents, without considering the timestamp associated with documents. Therefore, it is difficult to model topic evolution, which is especially necessary for frequently updated on-line social media streams, such as Tweets. Research work following this direction aims to design temporal topical models that can recognize time-sensitive topics automatically, and differentiate the same/very similar topics within different time periods.

Bibliography

[1] Redundancy, Diversity, and Interdependent Document Relevance - SIGIR 2009 Workshop, 2008.

[2] A. Acharya, M. Cutts, J. Dean, P. Haahr, M. Henzinger, U. Hoelzle, S. Lawrence, K. Pfleger, O. Sercinoglu, and S. Tong. Information retrieval based on historical data. United States Patent 20050071741, USPTO, Mar. 2005.

[3] A. Acharya, M. Cutts, J. Dean, P. Haahr, M. Henzinger, U. Hoelzle, S. Lawrence, K. Pfleger, O. Sercinoglu, and S. Tong. Document scoring based on link-based criteria. US Patent 20070094255, Apr. 2007.

[4] E. Alfonseca, M. Ciaramita, and K. Hall. Gazpacho and summer rash: lexical relationships from temporal patterns of web search queries. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3 - Volume 3*, EMNLP '09, pages 1046–1055, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[5] O. Alonso and M. Lease. Crowdsourcing for information retrieval: principles, methods, and applications. In *SIGIR*, pages 1299–1300, 2011.

[6] G. Amati and C. J. Van Rijsbergen. Probabilistic models of information retrieval based on measuring the divergence from randomness. *ACM Trans. Inf. Syst.*, 20(4):357–389, Oct. 2002.

[7] Amazon Inc. Amazon mechanical turk. `http://www.mturk.com/`, 2011.

[8] E. Amitay, D. Carmel, M. Herscovici, R. Lempel, and A. Soffer. Trend detection through temporal link analysis. *Journal of the American Society for Information Science and Technology*, 55(14):1270–1281, 2004.

[9] E. Amitay and C. Paris. Automatically summarising Web sites — is there a way around it? In *Proceedings of the Ninth ACM International Conference on Information and Knowledge Management (CIKM)*, Washington, DC, Nov. 2000.

[10] A. Arasu, J. Novak, A. Tomkins, and J. Tomlin. PageRank computation and the structure of the Web: Experiments and algorithms. In *Proceedings of the Eleventh International World Wide*, 2002.

[11] R. A. Baeza-Yates, F. Saint-Jean, and C. Castillo. Web structure, dynamics and page quality. In *Proceedings of 9th International Symposium on String Processing and Information Retrieval (SPIRE)*, pages 117–130, 2002.

[12] Z. Bar-Yossef, A. Z. Broder, R. Kumar, and A. Tomkins. Sic transit gloria telae: Towards an understading of the web's decay. In *Proc. 13th Int'l World Wide Web Conf.*, pages 328–337, May 2004.

[13] K. Berberich, S. Bedathur, O. Alonso, and G. Weikum. A language modeling approach for temporal information needs. In C. Gurrin, Y. He, G. Kazai, U. Kruschwitz, S. Little, T. Roelleke, S. Rüger, and K. van Rijsbergen, editors, *Advances in Information Retrieval : 32nd European Conference on IR Research, ECIR 2010*, volume 5993 of *Lecture Notes in Computer Science*, pages 13–25, Milton Keynes, UK, 2010. Springer.

[14] K. Berberich, S. Bedathur, M. Vazirgiannis, and G. Weikum. Buzzrank... and the trend is your friend. In *Proceedings of the 15th International Conference on World Wide Web*, pages 937–938, New York, May 2006. ACM Press.

[15] K. Berberich, S. Bedathur, G. Weikum, and M. Vazirgiannis. Comparing apples and oranges: Normalized PageRank for evolving graphs. In *Proceedings of the 16th International Conference on World Wide Web*, pages 1145–1146, New York, May 2007. ACM Press.

[16] K. Berberich, M. Vazirgiannis, and G. Weikum. Time-aware authority ranking. *Algorithms and Models for the Web-Graph*, 3243/2004(3):32, 2004.

[17] K. Berberich, M. Vazirgiannis, and G. Weikum. Time-aware authority ranking. *Internet Mathematics*, 2(3):301–332, 2005.

[18] A. Bhattacharyya. On a measure of divergence between two statistical populations defined by their probability distributions. *Bulletin of the Calcutta Mathematical Society*, 35:99–109, 1943.

[19] J. Bian, X. Li, F. Li, Z. Zheng, and H. Zha. Ranking specialization for web search: a divide-and-conquer approach by using topical ranksvm. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 131–140, New York, NY, USA, 2010. ACM.

[20] J. Bian, T.-Y. Liu, T. Qin, and H. Zha. Ranking with query-dependent loss for web search. In *Proceedings of the third ACM international conference on Web search and data mining*, WSDM '10, pages 141–150, New York, NY, USA, 2010. ACM.

[21] J. Bian, Y. Liu, D. Zhou, E. Agichtein, and H. Zha. Learning to recognize reliable users and content in social media with coupled mutual reinforcement. In *WWW '09: Proceedings of the 18th international conference on World wide web*, pages 51–60, New York, NY, USA, 2009. ACM.

[22] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the seventh international conference on World Wide Web 7*, WWW7, pages 107–117, Amsterdam, The Netherlands, The Netherlands, 1998. Elsevier Science Publishers B. V.

[23] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. In *Proceedings of the 7th International World Wide Web Conference*, pages 107–117,

Brisbane, Australia, Apr. 1998.

[24] A. Broder. A taxonomy of web search. *SIGIR Forum*, 36(2):3–10, Fall 2002.

[25] C. J. C. Burges, R. Ragno, and Q. V. Le. Learning to Rank with Nonsmooth Cost Functions. In B. Schölkopf, J. C. Platt, T. Hoffman, B. Schölkopf, J. C. Platt, and T. Hoffman, editors, *NIPS*, pages 193–200. MIT Press, 2006.

[26] C. J. C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. N. Hullender. Learning to rank using gradient descent. In *ICML*, pages 89–96, 2005.

[27] V. Bush. As we may think. *Atlantic Monthly*, 176(1):101–108, July 1945.

[28] D. Cai, X. He, J.-R. Wen, and W.-Y. Ma. Block-level link analysis. In *Proc. of the 27th Annual Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 440–447, July 2004.

[29] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning*, ICML '07, pages 129–136, New York, NY, USA, 2007. ACM.

[30] D. Chakrabarti and C. Faloutsos. Graph mining: Laws, generators, and algorithms. *ACM Comput. Surv.*, 38(1):2, 2006.

[31] O. Chapelle, Y. Chang, and T.-Y. Liu. Future directions in learning to rank. *Journal of Machine Learning Research - Proceedings Track*, 14:91–100, 2011.

[32] S. Chien and N. Immorlica. Semantic similarity between search engine queries using temporal correlation. In *Proceedings of the 14th International World Wide Web Conference*. ACM Press, May 2005.

[33] J. Cho and H. Garcia-Molina. Estimating frequency of change. *ACM Transactions on Internet Technology*, 3(3):256–290, Aug. 2003.

[34] J. Cho, H. Garcia-Molina, T. Haveliwala, W. Lam, A. Paepcke, S. Raghavan, and G. Wesley. Stanford WebBase components and applications. *ACM Trans. on Internet Technology*, 6(2):153–186, 2006.

[35] J. Cho, S. Roy, and R. E. Adams. Page quality: In search of an unbiased web ranking. In *Proceedings of ACM SIGMOD*, Baltimore, MD, June 2005.

[36] Y.-j. Chung, M. Toyoda, and M. Kitsuregawa. A study of link farm distribution and evolution using a time series of web snapshots. In *AIRWeb '09: Proceedings of the 5th International Workshop on Adversarial Information Retrieval on the Web*, pages 9–16, New York, NY, USA, 2009. ACM.

[37] R. B. Cleveland, W. S. Cleveland, J. E. Mcrae, and I. Terpenning. STL: A seasonal-trend decomposition procedure based on loess. *Journal of Official Statistics*, 6(1):3–73, 1990.

[38] C. Cortes, M. Mohri, and A. Rastogi. Magnitude-preserving ranking algorithms. In *Proceedings of the 24th international conference on Machine learning*, ICML '07, pages 169–176, New York, NY, USA, 2007. ACM.

[39] C. Cortes and V. Vapnik. Support-vector networks. *Mach. Learn.*, 20(3):273–297, Sept. 1995.

[40] N. Craswell, D. Hawking, and S. Robertson. Effective site finding using link anchor information. In *Proceedings of ACM SIGIR*, pages 250–257, New Orleans, LA, Sept. 2001.

[41] N. Dai and B. D. Davison. Freshness matters: In flowers, food, and web authority. In *Proc. of SIGIR*, pages 114–121, Geneva, Switzerland, 2010.

[42] N. Dai and B. D. Davison. Mining anchor text trends for retrieval. In *Proceedings of 32nd European Conference on Information Retrieval (ECIR 2010)*, 2010.

[43] N. Dai, B. D. Davison, and Y. Wang. Mining neighbors' topicality to better control authority flow. In *ECIR*, pages 653–657, 2010.

[44] N. Dai, M. Shokouhi, and B. D. Davison. Learning to rank for freshness and relevance. In *SIGIR*, pages 95–104, 2011.

[45] W. Dakka, L. Gravano, and P. G. Ipeirotis. Answering general time sensitive queries. In *Proceeding of the 17th ACM conference on Information and knowledge management*, CIKM '08, pages 1437–1438, New York, NY, USA, 2008. ACM.

[46] B. D. Davison. Topical locality in the Web. In *Proc. of the 23rd Annual ACM SIGIR Int'l Conf. on Research and Dev. in Info. Retrieval*, pages 272–279, July 2000.

[47] O. de Kretser and A. Moffat. Effective document presentation with a locality-based similarity heuristic. In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 113–120, New York, NY, USA, 1999. ACM.

[48] G. M. Del Corso, A. Gulli, and F. Romani. Fast pagerank computation via a sparse linear system (extended abstract). In S. Leonardi, editor, *WAW*, volume 3243 of *Lecture Notes in Computer Science*, pages 118–130. Springer, 2004.

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

[50] A. Dong, Y. Chang, Z. Zheng, G. Mishne, J. Bai, R. Zhang, K. Buchner, C. Liao, and F. Diaz. Towards recency ranking in web search. In *Proceedings of the third ACM international conference on Web search and data mining*, WSDM '10, pages 11–20, New York, NY, USA, 2010. ACM.

[51] A. Dong, R. Zhang, P. Kolari, J. Bai, F. Diaz, Y. Chang, Z. Zheng, and H. Zha. Time is of the essence: improving recency ranking using twitter data. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 331–340, New York, NY, USA, 2010. ACM.

[52] Z. Dou, R. Song, J.-Y. Nie, and J.-R. Wen. Using anchor texts with their hyperlink structure for web search. In *Proceedings of SIGIR*, pages 227–234. ACM, 2009.

[53] N. Eiron and K. S. McCurley. Locality, hierarchy, and bidirectionality on the Web. In *Second Workshop on Algorithms and Models for the Web-Graph (WAW 2003)*, Budapest, Hungary, May 2003. Extended Abstract.

[54] J. L. Elsas and S. T. Dumais. Leveraging temporal dynamics of document content in relevance ranking. In *WSDM*, pages 1–10, 2010.

[55] H. Fang and C. Zhai. An exploration of axiomatic approaches to information retrieval. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '05, pages 480–487, New York, NY, USA, 2005. ACM.

[56] M. Farah and D. Vanderpooten. An outranking approach for rank aggregation in information retrieval. In *Proceedings of the 30th annual intl' ACM SIGIR conf.*, pages 591–598, New York, NY, USA, 2007. ACM.

[57] D. Fetterly, M. Manasse, M. Najork, and J. L. Wiener. A large-scale study of the evolution of web pages. In *Proceedings of the 12th International World Wide Web Conference*, pages 669–678. ACM Press, May 2003.

[58] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. *J. Mach. Learn. Res.*, 4:933–969, Dec. 2003.

[59] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *Proc. 13th International Conference on Machine Learning*, pages 148–156, 1996.

[60] A. Fujii. Modeling anchor text and classifying queries to enhance web document retrieval. In *Proceeding of the 17th international conference on World Wide Web*, pages 337–346, New York, NY, USA, 2008. ACM.

[61] J. Gao, H. Qi, X. Xia, and J.-Y. Nie. Linear discriminant model for information retrieval. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '05, pages 290–297, New York, NY, USA, 2005. ACM.

[62] X. Geng, T.-Y. Liu, T. Qin, A. Arnold, H. Li, and H.-Y. Shum. Query dependent ranking using k-nearest neighbor. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, pages 115–122, New York, NY, USA, 2008. ACM.

[63] S. Gollapudi and A. Sharma. An axiomatic approach for result diversification. In *Proceedings of the 18th international conference on World wide web*, WWW '09, pages 381–390, New York, NY, USA, 2009. ACM.

[64] Google Blog. Outpouring of searches for the late michael jackson. `http://googleblog.blogspot.com/2009/06/outpouring-of-searches-for-late-michael.html`, 2009.

[65] Google Inc. Google trends home page. `http://www.google.com/trends`, 2010.

[66] J. D. Hamilton. *Time-series analysis.* Princeton Univerity Press, 1 edition, Jan. 1993.

[67] T. Haveliwala, S. Kamvar, A. Kamvar, and G. Jeh. An analytical comparison of approaches to personalizing pagerank. Technical report, Stanford University, 2003.

[68] T. H. Haveliwala. Topic-sensitive PageRank. In *Proc. of the 11th Int'l World Wide Web Conf.*, pages 517–526. ACM Press, May 2002.

[69] L. Hong, Z. Yang, and B. D. Davison. Incorporating participant reputation in community-driven question answering systems. In *CSE (4)*, pages 475–480, 2009.

[70] Internet Archive. The Internet Archive. http://www.archive.org/, 2011.

[71] K. Jarvelin and J. Kekalainen. IR evaluation methods for retrieving highly relevant documents. In *Proc. of the 23rd Annual Int'l ACM SIGIR Conference*, pages 41–48, July 2000.

[72] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD)*, pages 133–142, New York, NY, 2002. ACM Press.

[73] K. S. Jones, S. Walker, and S. E. Robertson. A probabilistic model of information retrieval: development and comparative experiments. *Inf. Process. Manage.*, 36(6):779–808, Nov. 2000.

[74] R. Jones and F. Diaz. Temporal profiles of queries. *ACM Trans. Inf. Syst.*, 25(3):14, 2007.

[75] I.-H. Kang and G. Kim. Query type classification for web document retrieval. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, SIGIR '03, pages 64–71, New York, NY, USA, 2003. ACM.

[76] N. Kanhabua and K. Nørvåg. A comparison of time-aware ranking methods. In *SIGIR*, pages 1257–1258, 2011.

[77] K. Kise, M. junker, A. Dengel, and K. Matsumoto. Passage retrieval based on density distributions of terms and its applications to document retrieval and question answering. volume 2956 of *LNCS*, Berlin/Heidelberg, 2004. Springer.

[78] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA-98)*, pages 668–677, San Francisco, CA, Jan. 1998.

[79] Y. Koren. Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09, pages 447–456, New York, NY, USA, 2009. ACM.

[80] R. Kraft and J. Zien. Mining anchor text for query refinement. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 666–674, New York, NY, USA, 2004. ACM.

[81] A. Kulkarni, J. Teevan, K. M. Svore, and S. T. Dumais. Understanding temporal query dynamics. In *Proceedings of the fourth ACM international conference on Web*

search and data mining, WSDM ’11, pages 167–176, New York, NY, USA, 2011. ACM.

[82] U. Lee, Z. Liu, and J. Cho. Automatic identification of user goals in web search. In *Proceedings of the 14th World Wide Web Conference (WWW)*, pages 391–400, New York, NY, 2005. ACM Press.

[83] L. Li, F. Liu, and W. Chou. An information theoretic approach for using word cluster information in natural language call routing. Technical Report ALR-2003-014, Avaya Labs Research, Apr. 2003.

[84] P. Li, C. J. C. Burges, and Q. Wu. Mcrank: Learning to rank using multiple classification and gradient boosting. In J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, editors, *NIPS*. Curran Associates, Inc., 2007.

[85] Y. Li and J. Tang. Expertise search in a time-varying social network. In *Web-Age Information Management, 2008. WAIM ’08. The Ninth International Conference on*, pages 293–300, July 2008.

[86] Y. Liu, B. Gao, T.-Y. Liu, Y. Zhang, Z. Ma, S. He, and H. Li. Browserank: letting web users vote for page importance. In *SIGIR ’08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 451–458, New York, NY, USA, 2008. ACM.

[87] Y. Lv and C. Zhai. Positional language models for information retrieval. In *SIGIR ’09: Proceedings of the 32nd international ACM SIGIR conference on Research and*

development in information retrieval, pages 299–306, New York, NY, USA, 2009. ACM.

[88] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval.* Cambridge University Press, 2008.

[89] N. Manouselis and C. Costopoulou. Analysis and classification of multi-criteria recommender systems. *World Wide Web*, 10:415–441, December 2007.

[90] D. Metzler, R. Jones, F. Peng, and R. Zhang. Improving search relevance for implicitly temporal queries. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '09, pages 700–701, New York, NY, USA, 2009. ACM.

[91] D. Metzler, J. Novak, H. Cui, and S. Reddy. Building enriched document representations using aggregated anchor text. In *Proceedings of the 32nd SIGIR conference*, pages 219–226. ACM, 2009.

[92] R. Nallapati. Discriminative models for information retrieval. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '04, pages 64–71, New York, NY, USA, 2004. ACM.

[93] L. Nie and B. D. Davison. Separate and inequal: Preserving heterogeneity in topical authority flows. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 443–450, July 2008.

[94] L. Nie, B. D. Davison, and X. Qi. Topical link analysis for web search. In *Proc. of the 29th Annual Int'l ACM SIGIR Conference*, pages 91–98, Aug. 2006.

[95] L. Nie, B. D. Davison, and B. Wu. From whence does your authority come? Utilizing community relevance in ranking. In *Proceedings of the 22nd National Conference on Artificial Intelligence (AAAI)*, pages 1421–1426, July 2007.

[96] L. Nie, B. D. Davison, and B. Wu. Ranking by community relevance. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 873–874, July 2007.

[97] NIST. Text REtrieval Conference (TREC) home page. http://trec.nist.gov/, 2008.

[98] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: Bringing order to the Web. Technical report, Stanford University, 1998. Available from http://dbpubs.stanford.edu/pub/1999-66. Accessed 29 March 2008.

[99] M.-H. Peetz, E. Meij, M. de Rijke, and W. Weerkamp. Adaptive temporal query modeling. In *ECIR*, pages 455–458, 2012.

[100] D. Petkova and W. B. Croft. Proximity-based document representation for named entity retrieval. In *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 731–740, New York, NY, USA, 2007. ACM.

[101] X. Qi, L. Nie, and B. D. Davison. Measuring similarity to detect qualified links. In *AIRWeb*, 2007.

[102] T. Qin, X.-D. Zhang, M.-F. Tsai, D.-S. Wang, T.-Y. Liu, and H. Li. Query-level loss functions for information retrieval. *Inf. Process. Manage.*, 44(2):838–855, 2008.

[103] K. Radinsky, K. M. Svore, S. T. Dumais, J. Teevan, A. Bocharov, and E. Horvitz. Modeling and predicting behavioral dynamics on the web. In *WWW*, pages 599–608, 2012.

[104] S. Robertson, S. Walker, M. M. Beaulieu, M. Gatford, and A. Payne. Okapi at trec-4. In *The Fourth Text REtrieval Conference (TREC-4)*, pages 73–96, 1995.

[105] S. Robertson and H. Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389, Apr. 2009.

[106] S. Robertson, H. Zaragoza, and M. Taylor. Simple BM25 extension to multiple weighted fields. In *Proceedings of the thirteenth ACM CIKM Conference*, pages 42–49, New York, NY, USA, 2004. ACM.

[107] S. E. Robertson. Overview of the OKAPI projects. *Journal of Documentation*, 53:3–7, 1997.

[108] S. E. Robertson. Readings in information retrieval. chapter The probability ranking principle in IR, pages 281–286. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1997.

[109] S. E. Robertson and K. Sparck Jones. Document retrieval systems. chapter Relevance weighting of search terms, pages 143–160. Taylor Graham Publishing, London, UK, UK, 1988.

[110] S. M. Ross. *Introduction to Probability Models, Ninth Edition*. Academic Press, Inc., Orlando, FL, USA, 2006.

[111] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620, Nov. 1975.

[112] R. L. T. Santos, C. Macdonald, and I. Ounis. Exploiting query reformulations for Web search result diversification. In *Proceedings of the 19th International Conference on World Wide Web*, pages 881–890, Raleigh, NC, USA, 2010. ACM.

[113] G. Shen, B. Gao, T.-Y. Liu, G. Feng, S. Song, and H. Li. Detecting link spam using temporal information. In *Proc. of IEEE International Conference on Data Mining (ICDM)*, pages 1049–1053, 2006.

[114] M. Shokouhi and K. Radinsky. Time-sensitive query auto-completion. In *SIGIR*, pages 601–610, 2012.

[115] R. Steuer. *Multiple Criteria Optimization: Theory, Computation and Application*. John Wiley, 546 pp, 1986.

[116] M. Taylor, J. Guiver, S. Robertson, and T. Minka. Softrank: optimizing non-smooth rank metrics. In *Proceedings of the international conference on Web search and web data mining*, WSDM '08, pages 77–86, New York, NY, USA, 2008. ACM.

[117] M.-F. Tsai, T.-Y. Liu, T. Qin, H.-H. Chen, and W.-Y. Ma. Frank: a ranking method with fidelity loss. In W. Kraaij, A. P. de Vries, C. L. A. Clarke, N. Fuhr, and N. Kando, editors, *SIGIR*, pages 383–390. ACM, 2007.

[118] L. Wang, J. Lin, and D. Metzler. Learning to efficiently rank. In *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '10, pages 138–145, New York, NY, USA, 2010. ACM.

[119] L. Wang, J. J. Lin, and D. Metzler. A cascade ranking model for efficient ranked retrieval. In *SIGIR*, pages 105–114, 2011.

[120] L. Wang, D. Metzler, and J. J. Lin. Ranking under temporal constraints. In *CIKM*, pages 79–88, 2010.

[121] S. Whiting, I. A. Klampanos, and J. M. Jose. Temporal pseudo-relevance feedback in microblog retrieval. In *ECIR*, pages 522–526, 2012.

[122] S. Whiting, Y. Moshfeghi, and J. M. Jose. Exploring term temporality for pseudo-relevance feedback. In *SIGIR*, pages 1245–1246, 2011.

[123] S. R. Wolfe and Y. Zhang. User-centric multi-criteria information retrieval. In *Proceedings of the 32nd intl' ACM SIGIR conf.*, pages 818–819, New York, NY, USA, 2009. ACM.

[124] B. Wu, V. Goel, and B. D. Davison. Propagating trust and distrust to demote web spam. In *Proceedings of the WWW2006 Workshop on Models of Trust for the Web (MTW)*, Edinburgh, Scotland, May 2006.

[125] Q. Wu, C. Burges, K. Svore, and J. Gao. Adapting boosting for information retrieval measures. *Information Retrieval*, 13, 2010.

[126] F. Xia, T.-Y. Liu, J. Wang, W. Zhang, and H. Li. Listwise approach to learning to rank: theory and algorithm. In *Proceedings of the 25th international conference on Machine learning*, ICML '08, pages 1192–1199, New York, NY, USA, 2008. ACM.

[127] J. Xu and H. Li. Adarank: a boosting algorithm for information retrieval. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, pages 391–398, New York, NY, USA, 2007. ACM.

[128] Yahoo!, Inc. Yahoo! site explorer. `http://siteexplorer.search.yahoo.com/`, 2009.

[129] L. Yang, L. Qi, Y.-P. Zhao, B. Gao, and T.-Y. Liu. Link analysis using time series of web graphs. In *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 1011–1014, New York, NY,

USA, 2007. ACM.

[130] Z. Yang, L. Hong, and B. D. Davison. Topic-driven multi-type citation network analysis. In *Adaptivity, Personalization and Fusion of Heterogeneous Information*, RIAO '10, pages 24–31, Paris, France, France, 2010.

[131] J. Y. Yeh, J. Y. Lin, H. R. Ke, and W. P. Yang. Learning to Rank for Information Retrieval Using Genetic Programming. In T. Joachims, H. Li, T. Y. Liu, and C. Zhai, editors, *SIGIR 2007 workshop: Learning to Rank for Information Retrieval*, July 2007.

[132] X. Yi and J. Allan. A content based approach for discovering missing anchor text for web search. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '10, pages 427–434, New York, NY, USA, 2010. ACM.

[133] P. S. Yu, X. Li, and B. Liu. On the temporal dimension of search. In *Proc. 13th Int'l World Wide Web Conf.*, pages 448–449, May 2004.

[134] Y. Yue, T. Finley, F. Radlinski, and T. Joachims. A support vector method for optimizing average precision. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, pages 271–278, New York, NY, USA, 2007. ACM.

[135] H. Zaragoza, N. Craswell, M. Taylor, S. Saria, and S. Robertson. Microsoft cambridge at trec 13: Web and hard tracks. In *TREC '13: Proceedings of the thirteenth Text REtrieval Conference*, 2004.

[136] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.*, 22(2):179–214, Apr. 2004.

[137] Z. Zheng, K. Chen, G. Sun, and H. Zha. A regression framework for learning ranking functions using relative relevance judgments. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, pages 287–294, New York, NY, USA, 2007. ACM.

[138] Z. Zheng, H. Zha, T. Zhang, O. Chapelle, K. Chen, and G. Sun. A general boosting method and its application to learning ranking functions for web search. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 1697–1704. MIT Press, Cambridge, MA, 2008.

Vita

1984 Born in Nanjing, China.

2001 Graduated from the Science Experimental Class in the Experimental High School Attached to Beijing Normal University, Beijing, China.

2005 B.E. in Computer Science and Technology, Beijing University of Technology, China.

2010 M.S. in Computer Science, Lehigh University.

2007 - 2013 Graduate study in Department of Computer Science and Engineering, Lehigh University.

2010 Na Dai and Brian D. Davison. Mining Anchor Text Trends for Retrieval. In *Proceedings of the* 32nd *European Conference on Information Retrieval (ECIR)*, Milton Keynes, UK, March 2010.

2010 Na Dai and Brian D. Davison. Freshness Matters: In Flowers, Food, and Web Authority. In *Proceedings of the* 33rd *Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Geneva, Switzerland, July 2010.

2010 Na Dai and Brian D. Davison. Capturing Page Freshness for Web Search. In *Proceedings of the* 33rd *Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Geneva, Switzerland, July 2010.

2011 Na Dai, Milad Shokouhi and Brian D. Davison. Learning to Rank for Freshness and Relevance. In *Proceedings of the* 34th *Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Beijing, China, July 2011.

2011 Na Dai, Milad Shokouhi and Brian D. Davison. Multi-objective Optimization in Learning to Rank. In *Proceedings of the* 34th *Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Beijing, China, July 2011.