Department of Computer Science
Series of Publications A
Report A-2018-6

Online Personalization in Exploratory Search

Joel Pyykkö

To be presented, with the permission of the Faculty of Science of the University of Helsinki, for public examination in Auditorium B123, Exactum, Kumpula, Helsinki on June 15th, 2018 at 12 o'clock noon.

University of Helsinki
Finland

Supervisor
Petri Myllymäki, University of Helsinki, Finland

Pre-examiners
Alexandros Iosifidis, Aarhus University, Denmark
Jorma Laaksonen, Aalto University, Finland

Opponent
Moncef Gabbouj, University of Tampere, Finland

Custos
Petri Myllymäki, University of Helsinki, Finland

Contact information

Department of Computer Science
P.O. Box 68 (Gustaf Hällströmin katu 2b)
FI-00014 University of Helsinki
Finland

Email address: info@cs.helsinki.fi
URL: http://cs.helsinki.fi/
Telephone: +358 2941 911

ISSN 1238-8645
ISBN 978-951-51-4303-7 (paperback)
ISBN 978-951-51-4304-4 (PDF)
Computing Reviews (1998) Classification: H.1.2, H.3.3, H.5.2
Helsinki 2018
Unigrafia

Online Personalization in Exploratory Search

Joel Pyykkö

Department of Computer Science
P.O. Box 68, FI-00014 University of Helsinki, Finland
joel.pyykko@cs.helsinki.fi

PhD Thesis, Series of Publications A, Report A-2018-6
Helsinki, June 2018, 101+63 pages
ISSN 1238-8645
ISBN 978-951-51-4303-7 (paperback)
ISBN 978-951-51-4304-4 (PDF)

Abstract

Modern society produces vast amounts of digital data related to multiple domains of our lives. We produce data in our free time when browsing the net or taking photos with various personal devices, such as phones or ipads. Businesses and governments also gather a lot of information related to our interests, habits or otherwise personal information (legal status, health data, etc.). The amount of data produced is growning too large for us to be handled manually, and so to assist the user, specialized information retrieval systems have been developed to allow efficient perusal of different types of data. Unfortunately, as using such systems often requires expert understanding of the domain in question, many users get lost in their attempt to navigate the search space. This problem will only be exacerbated in the future, as the amount of data keeps growing, giving us less time to learn about the domains involved.

Exploratory search is a field of research that studies user behaviour in situations, where users have little familiarity with the search domain, or have not yet decided exactly what their search goal is. Situations such as these arise when the user wishes to explore what is available, or is otherwise synthesizing or investigating the data. To assist the user in exploratory search and in finding relevant information, various methodologies may be employed, such as user modeling techniques or novel interfaces and data visualization techniques.

This thesis presents exploratory search techniques for online personalization and feature representations that allow efficient perusal of unknown datasets. These methods are showcased in two different search environments. First, we present a search engine for scientific document retrieval, which takes the user's knowledge level into account in order to provide the user with more or less diverse search results. The second search environment aims at supporting the user when browsing through a dataset of unannotated images. Overall, the research presented here describes a number of techniques based on reinforcement learning and neural networks that, compared to traditional search engines, can provide better support for users who are unsure of the final goal of their search or who cannot easily formulate their search needs.

Computing Reviews (1998) Categories and Subject Descriptors:
H.1.2 User/Machine Systems
H.3.3 Information Search and Retrieval
H.5.2 User Interfaces

General Terms:
Exploratory Search, Information Retrieval

Additional Key Words and Phrases:
Content-based Information Retrieval, Deep Learning, Bandit Algorithms

Acknowledgements

I would like to thank Business Finland/Center for Visual and Decision Informatics (CVDI) and Academy of Finland/Finnish Centre of Excellence in Computational Inference Research (COIN) for supporting this work, allowing me to focus on research full time, as well as funding my travels to conferences relevant to my research field.

Our team of researchers have been the backbone of the work I achieved during my PhD studies, from giving invaluable advice, to collaborating with me on publications. I extend my gratitude to my supervisors, Professor Petri Myllymäki and Dr Dorota Głowacka, who have been a steady supply of guidance through my journey into academia. I also thank my co-authors, Dr Alan Medlar, Sayantan Hore, Lasse Tyrväinen, Pedram Daee and Professor Samuel Kaski. Our collaboration taught me many skills, from theory of good science, to applying it in practice. Furthermore, I thank the community of DoCS, especially the coordinator Dr Pirjo Moen, and the staff at our faculty, who have facilitated my research with their work.

Finally, I would like to thank my family and friends for all the support I have gained throughout the years I have been at the university.

In Helsinki, Finland, May 7, 2018

Joel Pyykkö

Contents

Chapter 1

Introduction

The amount of digitally stored data has been steadily rising through the last decades, and is predicted to continue so in to the foreseeable future. The source is not only in government or business produced information, but also in the rising personal production, and most of it tends to be without annotation or other meta-data. Over the last three decades, this growth has made accessing and perusing these repositories an arduous task for all involved, and the need for methodologies to understand and explore these repositories have been in high demand. Several advances have been made in various directions to address this problem, from trying to ease the overload of information [87], to adding semantic structure to the data [104] or having adaptive, personalized search systems at our service [22].

Still, most modern search engines rely on look-up search, where the user's query is directly translated into results [75]. In these methodologies, the user is assumed to know what they are looking for, and it is assumed they will reach it with only a single query. This is a problematic assumption when a user is still just getting acquainted with the topic they are facing, or are otherwise interested in learning about the scope of available data: They might not know what they are looking for yet, and would prefer to peruse the available content in an efficient manner. Studies have shown that these kinds of scenarios may be relevant in as much as half of modern search sessions [48]. In these scenarios it is important for the system to zoom in onto the relevant parts as fast as possible, while guaranteeing a good coverage onto what is available.

There are also other considerations for the search process which arise during the session. Users are rarely able to give explicit feedback, which would lead to a singular correct answer, but rather accidentally or purposefully wander around the search space. Even in cases where the feedback is explicit, it might be very sparse, only highlighting positive examples from

the data. Furthermore, users usually want results immediately, rarely waiting for more than 4 seconds before they feel the system is slow [23]. In these cases knowing how to utilize the available feedback information is crucial.

Exploratory search [75] studies scenarios like these, where the user has not yet decided on a target or is still learning to navigate and understand the search topic. This field aims to give the user a better understanding of the available data, using a large set of tools from data visualization, to semantic understanding of the content or greater control over communicating the user's interest.

In this dissertation we focus on optimizing the results of search engines based on information that can be collected during the search session. The core question is thus, *how do we detect the user's needs in online settings, even before they do?* By creating models of user behaviour, we aim to develop methods for search engine parameter optimization that are applicable for existing information retrieval systems. During our research we built two content-based information retrieval engines that were designed specifically to test exploratory search scenarios where we can control the user's familiarity with the data: one is meant for scientific document retrieval, while the other is for image retrieval.

The main focus of the presented research is on dynamic exploratory search that reacts quickly to the changing needs of the user. Whatever the user's knowledge level, search context or their interests, the retrieval engine reacts quickly, giving a comprehensive view of the dataset. Each system utilizes a form of similarity measure, which is used to propagate the estimation of the relevance over the whole dataset. Thanks to this, the engines rely on content-based retrieval methods without the need for tagging the data. The personalization is further augmented with information gained from the user's behaviour, such as when assessing their knowledge level. Our findings open interesting venues for further development in information retrieval, especially for environments that lack pre-made annotations.

Another theme for our research is to find alternative ways to measure both the breadth and success of a single iteration of exploration. Evaluation of exploratory search has been notoriously hard to do, especially now that more and more ad hoc tools for learning are being developed. The success of exploratory search cannot be measured comfortably with the tools available for classical information retrieval methods, as the user still does not have a singular target within the search space. Hence, precision, recall and F1 scores become useful only after multiple search iterations, when a singular target has been formed in the user's mind. We explored a novel metric for measuring search space coverage, as well as a new user study setting.

The dissertation will start out by covering the background of these concepts: In Chapter 2 exploratory search, Chapter 3 relevance feedback and in Chapter 4 content-based information retrieval. Each of these fields used together form the basis for search engines that today allow users to tackle novel data. After this, in Chapters 5 and 6, we introduce new ways to preprocess the data for our scientific document search tool and the image retrieval framework, specifically regarding the similarity measures. Finally, we conclude the findings presented by our work in Chapter 7, and list the publications describing these systems.

1.1 Objectives

The objective of the dissertation was to develop frameworks with capacity for reactive exploratory search in the wild. These frameworks should elicit the ability to work in a varied set of domains, and require minimal effort from the user during application. My final work is based on four claims, which together have comprehensive implications for future exploratory search systems which have a low threshold for incorporation into a search engine.

Claim 1: It is possible to personalize the search parameters per session and user, thus accommodating users outside of population-wide tendencies.

Claim 2: People, who know what they do not know, give more reliable feedback, and this information can be utilized efficiently when optimizing the search system's parameters.

Claim 3: It is possible to use very general features, yet still catch a very specific target with contemporary transfer learning.

Claim 4: Users do not need to know exactly what they want at the beginning of the search session, but with appropriate support from the search engine, they can direct their search towards a desired direction.

Claim 1: I argue that personalization is possible to be done online, and learning the needs of each user, per session, yields superior results compared to population-based modeling. Contemporary personalization methods rely on population-level generic statistics, which often omit a large portion of users and use cases. Our results indicate that a combination of user information and generalizable features from multiple domains produce robust side information for this purpose.

Claim 2: The presented work posits that novice seekers give noisy feedback, while people with intermediate to expert knowledge give more precise feedback. This indicates that users with better knowledge levels on the topic give feedback that can be utilized in classical relevance feedback systems, while users completely new to the topic might gain more from conventional exploration strategies. The reasoning is that people who do not know what they do not know are unable to direct the search consciously, which renders most feedback systems moot.

Claim 3: I argue that modern deep learning has attained robust transfer learning capabilities, allowing us to create end-to-end online systems for information retrieval. This allows the real-time retrieval of items, which converges towards a group of relevant items within a reasonable number of iterations, with minimal feedback from the user. These systems can be implemented with low effort using existing, unannotated datasets, and require little domain expertize from the user to peruse comfortably.

Claim 4: I posit that it is possible for users to use their intuition when directing the search, as long as the search engine exhibits the necessary transfer learning capabilities. This means it is no longer necessary for the user to be acutely aware of their search target, but they can instead explore the search space by giving relevance feedback that indicates a general "preference" towards one item or another.

1.2 Author's Contribution

Publication I:

> Medlar, Alan and Pyykkö, Joel and Głowacka, Dorota, Towards Fine-Grained Adaptation of Exploration / Exploitation in Information Retrieval, Proceedings of the 22Nd International Conference on Intelligent User Interfaces, IUI '17, 2017, pages 623–627, ACM, [80].

Contribution:

> I was involved in designing the system and planning the user experiments. I conducted all of the user studies. Lastly, I was part of the writing process, writing the experimental setup section and helping in the revision of the whole manuscript. I took part in the analysis of the data both as an expert reviewer and when discussing the implications of our work.

Publication II:

Medlar, Alan and Pyykkö, Joel and Głowacka, Dorota, Fine-grained adaptation in exploratory search systems using simple user behaviour characteristics, Accepted in UMAP 2018, the 26th Conference on User Modeling, Adaptation and Personalization [81].

Contribution

I was involved in planning the user experiments. I conducted half of the user studies along with Evgenia Lyjina. Lastly, I was part of the writing process, writing the experimental setup section and helping in the revision of the whole manuscript. I took part in the analysis of the data both as an expert reviewer and when discussing the implications of our work.

Publication III:

A reinforcement learning approach to query-less image retrieval, Hore, Sayantan and Tyrväinen, Lasse and Pyykkö, Joel and Głowacka, Dorota, International Workshop on Symbiotic Interaction, pages 121–126, 2014, Springer [50]

Contribution

I participated in the theoretical discussion of the system development phase and the underlying algorithm along with Dorota Głowacka and Sayantan Hore. I was part of the design process of the various interface options.

Publication IV:

Pyykkö, Joel and Głowacka, Dorota, Interactive Content-Based Image Retrieval with Deep Neural Networks, Symbiotic Interaction: 5th International Workshop, Symbiotic 2016, Padua, Italy, September 29–30, 2017, Springer International Publishing, pages 77–88, [89]

Contribution

Along with Dorota Głowacka, we worked on the planning phase of the system. I developed and implemented the underlying system, algorithms and conducted the experiments. I and Dorota Głowacka wrote and revisioned the manuscript.

Publication V:

Pyykkö, Joel and Głowacka, Dorota, Dynamic Exploratory Search in Content-Based Image Retrieval, Image Analysis: 20th Scandinavian

Conference, SCIA 2017, Tromsø, Norway, June 12–14, 2017, Proceedings, Part I, 2017, Springer International Publishing, pages 538–549, [90]

Contribution

Along with Dorota Głowacka, I worked on the planning phase of the system. I developed and implemented the underlying system, algorithms and conducted the experiments. I and Dorota Głowacka wrote and revised the manuscript.

Publication VI:

Daee, Pedram and Pyykkö, Joel and Głowacka, Dorota and Kaski, Samuel, Interactive Intent modeling from Multiple Feedback Domains, Proceedings of the 21st International Conference on Intelligent User Interfaces, IUI '16, 2016, pages 71–75, ACM [32].

Contribution

With Dorota Głowacka and Samuel Kaski, I considered the potential for a joint Gaussian Process Bandits model which would pass relevance feedback information from two related domains. During the development of the theoretic foundation for this system, with Pedram Daee, I built the system for the user studies, and performed them together half and half. Pedram Daee wrote the initial draft of the manuscript, after which all the participants joined for revisions. With Pedram, I designed and built the system for the user studies.

Chapter 2

Exploratory Search

Research on exploratory search [121] has become a prevalent field over the last few decades, mostly due to the increasing need for navigating the growing online repositories of data. The field of exploratory search grew around the existing Information Retrieval (IR) research [102], where an extensive list of background work indirectly supported it. As the automated search tools from the early-80's developed, requirement to support activities such as learning and assessing all of the available data became prevalent [95]. This created a need for efficient data management, what the classical query-response paradigm can no longer supply, where the query is a one-time information need of the user.

Greedy search looks only for relevant documents. Whenever the user refines their query further, the best fitting items are shown for them, taking into consideration only the query word itself. Exploratory search, in contrast, often looks at the available information from multiple directions and balances the shown results with greater variety. With this in mind, common Web search could be characterized as precision oriented ranking, where the focus is on avoiding irrelevant items. This is usually manifested in the system avoiding less relevant objects from the results. Exploratory search in contrast is said to be recall oriented [27], where the focus is on not missing any relevant documents instead, usually showing a broader aspect of the data.

However, it is not enough to merely rerank the results to contain more recall oriented items: if the space and time for shown documents is the same as in the precision oriented setup, the coverage will still be lacking. Hence, exploratory search has focused a lot on the presentation of the data too, condensing and bringing the information in an optimal fashion to the user [17]. This can be done by simply having easy access to lowly ranked

hits, or offer various ways to group relevant hits and give overviews of them. Giving the user control over these view forms makes an even more refined system.

Exploratory search gives the user a more thorough view over these search spaces, and assists in learning and decision making. For example, it was found that query reformulation, the practice of rewording the original query word after learning more about the topic, was needed roughly on 50% of search sessions [48]. Furthermore, Teevan et al. [112] observed that users usually directed their search with small steps, instead of directly jumping to the end result. Each step used local knowledge, which was used to navigate the search space, and often helped them form an understanding of it on the way: Such behaviour happened even when the users knew what they were looking for. These findings are often unsupported in modern search engines that dominate the field.

Finally, taking into consideration the various needs of different users has been studied extensively. To speed up the search process, the system should take into consideration the users' knowledge level of the topic, as well as related topics in general. Also, the user's information needs, be they superficial or in depth, should be accounted for. Relating these needs to what is available is needed, as certain databases have different information to offer, and choosing the best presentation of data for a particular session is difficult.

For example, what if the user does not know what they are looking for, or are looking for multiple topics at the same time? Or what if they have a vague idea of the target, but would like to have a better look at what is available? In each scenario, a broader selection of documents helps the user to make up their mind sooner. The users usually want to form a cohesive opinion on the topic, requiring various aspects of knowledge to be present as soon as possible. Specifically, the documents missing from precision oriented ranking are the ones that discuss these differing views, as the focus of precision ranking is instead on showing merely the best hits to a query word. More often than not, the user is oblivious to the most fitting query words for the topic to begin with, and has to figure them out during the search.

Three Activities The various activities that happen during search sessions can be divided into three loose categories: lookup search, learning, and investigating ([75], see Figure 2.1). The focus of exploratory search is on the latter two, learning and investigation. Lookup is the straightforward retrieval of known items, where a query word or similar identification brings

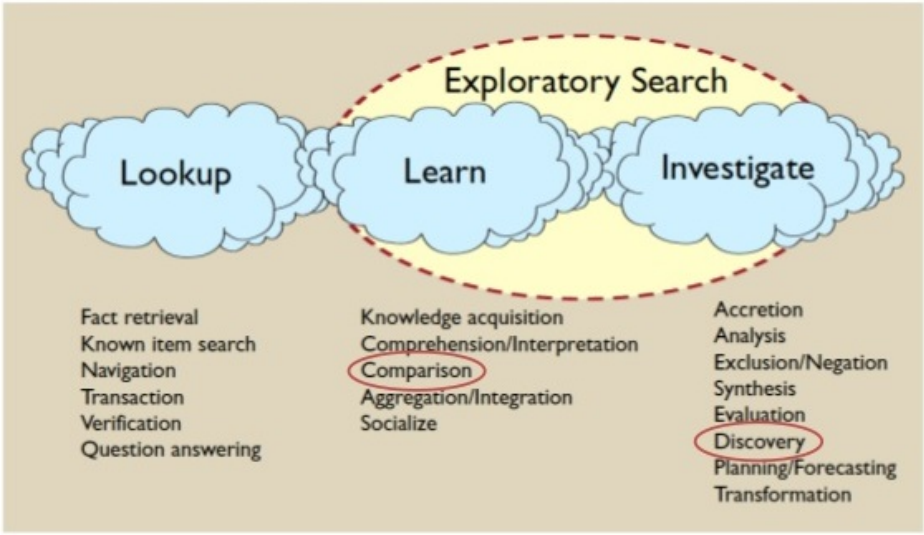


Figure 2.1: The three overlapping paradigms outlined by Marchionini [75], from lookup to learning and investigating.

precise results. This is good for situations where the user already knows the topic, and simply has to check the details for further reference, for example when a painter queries for Mona Lisa for reference. Learning on the other hand is more about comprehension of the data, aggregating relevant items and comparing them with each other for indepth understanding. It is about searching to learn, to acquaint oneself with the topic. Finally, investigation is about analysis, evaluation and synthesis of data, a more meta-data approach to what is actually available and how it relates to the whole picture.

Although the three activities are listed as separate, people often perform more than one of them within a session. The user might, for example, look up more information on a topic they have just learned to understand, or investigate a large number of items they have just retrieved individually. These types of activities are thus better thought of as "search tasks", and identifying what type of task the user requires at any given moment helps the search engine react to the user's needs.

Lookup search was originally conceived for database perusal, where the target was already known and easily found with a single query. These situations have grown increasingly rarer, as the Internet offers information in more varied settings for everyone. Users tend to be less knowledgeable about the topic and the contents of the database, and might not be inter-

ested in simple lookup activity in the first place.

Investigative search is more concerned with recall than precision, as finding more relevant objects is more important than avoiding irrelevant ones. Although experts know precise query words when looking for content, they also need robust annotation tools and other sophisticated resources to assess and analyze the data. In these cases, using relevance feedback [74] has been shown to give invaluable directive force for the search. However, people are often reluctant to interact enough with the system step by step, which has led to an increased interest in innovative interfaces [49, 62, 98, 113, 124].

Several authors [27] further elaborate on Marchionini's philosophy by noting a distinction in the context of structured data and unstructured data. For example, Herschel notes that for structured data the problem of exploratory search has a well defined solution in online analytical processing (OLAP) and data warehouses [31], where data integration, comparison, planning and forecasting are present in data reports. But when it comes to unstructured data, there are three use cases for further development in exploratory search: discovery, adaptation, and user centricity.

Exploratory search is all about exploring and discovering the new, as data that has not been labelled nor annotated is difficult to tackle. Adaptation on the other hand is the ability to break free from a set paradigm. The user has to be able to redefine the approach and *facet* [124] by which they wish to investigate the data. The third point is that much of the structure behind data is set and defined by the experts of the field. To be truly exploratory, the search has to accommodate those who are **still researching the topic**. For them, to explore the data means to learn the structure underneath as they peruse the data.

Exploratory Search and Information Retrieval Exploratory search is highly symbiotic with information retrieval [106], and originates from the same foundation. Indeed, one of the early appearances within the community was a workshop in an information retrieval conference [82]. Many of the widely used performance measures, user study setups and system designs are also familiar from IR literature.

Information retrieval is defined as a field that studies the efficient investigation and perusal of collections of data sources, such as text or multimedia [74, 102]. These searches may focus on documents, information within the documents themselves or metadata that describe the collection. The active engagement of the user gives feedback to direct the search engine, which in turn yields results for the user to peruse based on this feedback.

During a single session, there may be multiple communication iterations, where the user and the engine refine the results until the user is satisfied.

One early example of common history is Information-Seeking Support Systems (ISSS [77]), which were outlined as an extension of early information retrieval, spanning towards exploratory search, interactive search and human-computer information retrieval. The focus moved from the prevalent query-based search engines to cover planned behaviour and decision making, as well as to systems designed specifically for information seeking and learning.

As is evident, exploration is already an integral part of the information retrieval paradigm, and is implied throughout the definition. Exploratory search is thus the explicit formalization of the need and procedure by which information retrieval may be directed in a comprehensive manner [75, 121].

2.1 Interfaces

Interfaces play a major role in successful exploratory search and have been researched widely in the information retrieval field. A good layout makes the data more assessable, by visually condensing and abstracting information from the whole database. Sometimes less is more, as too much information may cause a cognitive overload [87], making the retrieval process tasking for the user. Furthermore, with the correct setup the user is able to give more informed feedback to the system. This can be done for example as a dial manipulating the available valences or features, or presenting the relation between documents as a spatial view.

An example of a minimalist interface is Google's search interface [2], which can be seen in Figure 2.2. Google's search engine retrieves websites from around the Internet based on queries, which may be given as a string of characters. The results are listed in descending order of importance, and the following is presented: The link's title, a description formulated either automatically or by the site maintainer, their fitness to the query and several other meta-data fields.

Google's search interface, and the manner by which it can be used, represents the classical query-based search paradigm. It requires careful formulation of queries to work effectively, requiring both knowledge of topic as well as expertise in using the search engine itself. In this case, the search algorithm responds better to certain kinds of wordings, which can be further refined via syntactical specifications. Furthering the search often requires learning more on the topic from the available links, from which consecutive, fresh queries have to be issued.

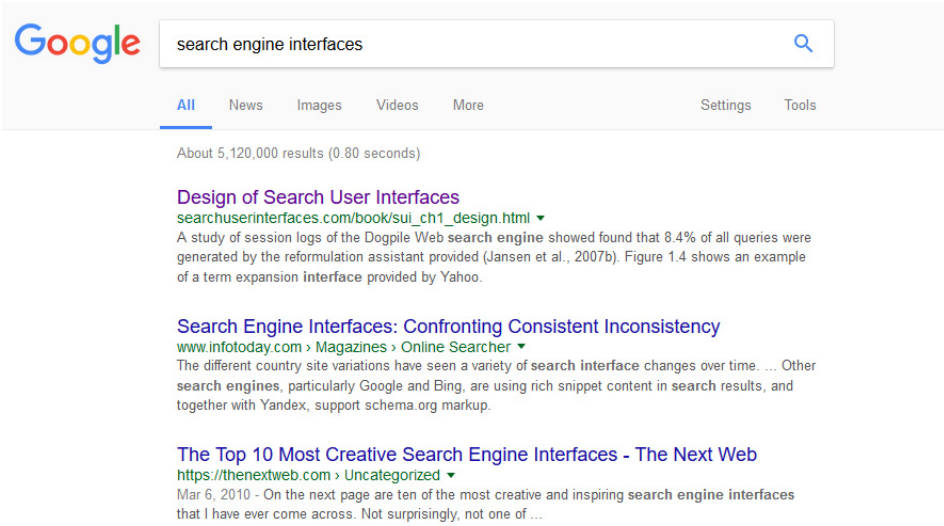


Figure 2.2: Classical query-based search interface [2]. The user defines a query as text, to which the best matching results are presented.

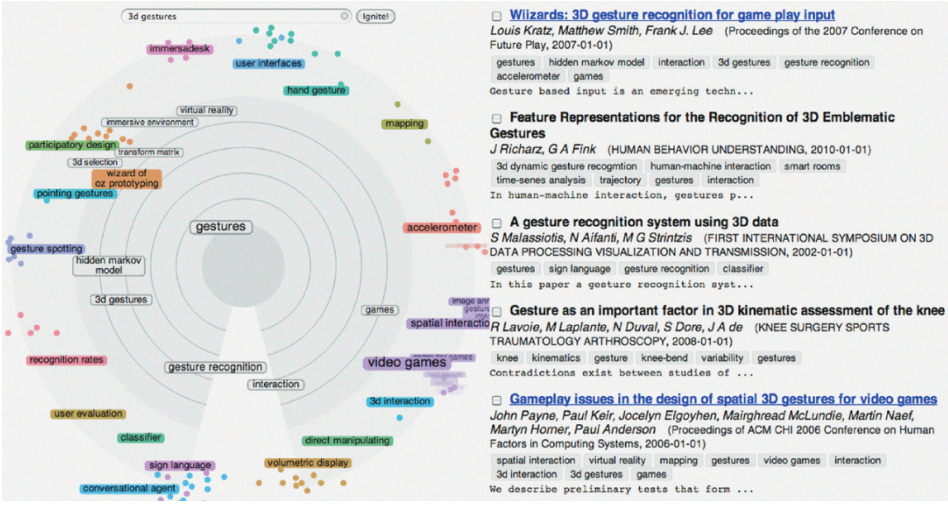


Figure 2.3: The Intent Radar interface [99], which was designed for assessing the user's intent during search. After querying the system with "3D gestures", the user is presented with a radar of keywords and a list of highest ranking documents. By manipulating the relevance of keywords on the radar, the user is able to fine-tune their search, repopulating the list of documents online.

Users have different starting points and goals every time they search for something, requiring flexibility from the search engine. They might have different knowledge levels, depending on the topic they explore. Similarly, the search tasks themselves might require various views of the data, where one view might help analyzing the seen data, while another helps navigating to relevant regions in the database. One such system can be seen in Figure 2.3, which presents the Intent Radar [99], an interface that helps the user better convey their intent to the system.

Finally, a given community itself produces and presents the data in a certain manner, and bringing this into a presentable form for people outside of the community might require extra work. For example, doctors might diagnose patients with a shorthand notation familiar to medicine, which would be uninterpretable for a patient accessing their own record online.

One large repository of useful work around interfaces has been done by the HCI (Human-Computer Interaction, [35]) community. Their work revolves around incentivizing users to provide more feedback, which is especially relevant for exploratory search. Instead of matching queries to results, HCI sees users as active agents with information needs and skills, while digital resources are maintained by communities, both of which evolve over time. This vantage point illuminates the complex environment a single procedure is a part of.

Many of the first interfaces were designed as menu views [75], which got their inspiration from restaurant menus. These views often break the information into a ready hierarchy as defined by the experts. Navigation in this category is mainly done with hypertext links, which were called embedded menus. Eventually, research found use for the refined feature representations, which enabled systems to react to singular details within the data.

Query-By-Example Query-By-Example interfaces (QBE, [130]) are systems based on non-textual queries in the multimedia domain. Contrary to lengthy descriptions or sets of key words, they instead provide actual examples from the database. This works well in domains that contain hard-to-annotate items, such as images, and has been used in various formats since its early formulation by Shneiderman in 1997 [105]. These environments are usually navigated via links from one query to another, and are thus well suited for the Web.

One earlier example is the Open Video Digital Library [76], which is a service for dynamic digital video viewing, as seen in Figure 2.4. It provides an easy interface for over-viewing available videos, as well as previewing

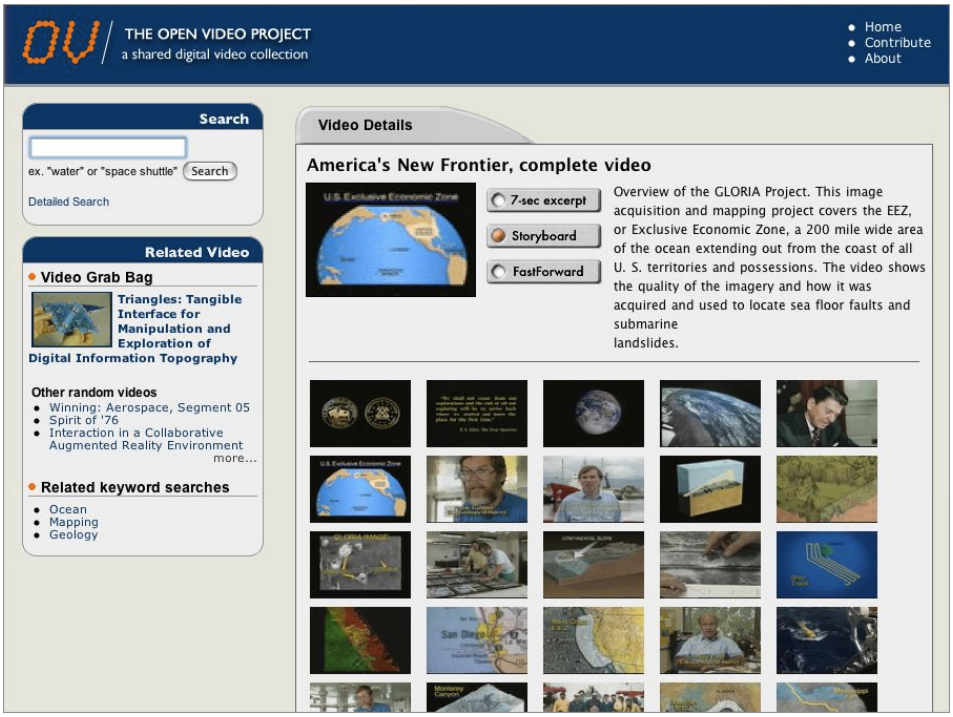


Figure 2.4: The Open Video Project's interface [76]. The items in the dataset contain meta-data that allows the user to search with a wide variety of options, such as genre or producer. Each query returns a list of video segments, as well as a detailed view of the selected item.

segments of their content. The service supplies alternative textual and visual representations too, supporting flexible control over the videos. The search can be further faceted with various ready-made partitions, and meta-data such as date, popularity, or title. All of this information together helps the user in choosing which parts of the database they want to download for further perusal, saving time and assisting decision making.

Faceted Search Large databases often contain multiple dimensions the data adheres to, any of which might be interesting to the user. For example, digital book libraries might contain meta-data from date of publication to writer, genre and excerpts from the books. These dimensions could be perused for information on publication density, content of the library and the productivity of a single author.

In faceted search [124], the user is allowed to add various filters over the dataset according to the chosen classification. The classification places each data point along a number of dimensions common to the database, which have been predefined, for example, by entity extraction. This requires analysis of either the meta-data of the dataset, or the data itself. Both are plausible for existing web pages, which is why they are often easy to implement.

For example, Relation Browser (RB) [40] was developed as a user interface for a variety of government statistical data, and their presentation to the public. The system specializes in exploring and combining information from different facets of public data, and was made generic enough for almost any kind of data set available. RB is a faceted search engine that may presents, for example, the topic, time, space or data format, with simple mouse-brushing actions. The facet may be moved along its axis, and frozen with a simple click, freeing the user to browse another facet along the partition.

Another system is Factic [115, 116], which presents an interface for personalized exploratory search, seen in Figure 2.5. The browser used a faceted approach to show semantic data of images (although applicable on other domains too), which used an ontological user model. Factic was later augmented with multi-paradigm exploration and a novel interface generation method that adjusted to the domain ontology in the various web pages where it was used [116].

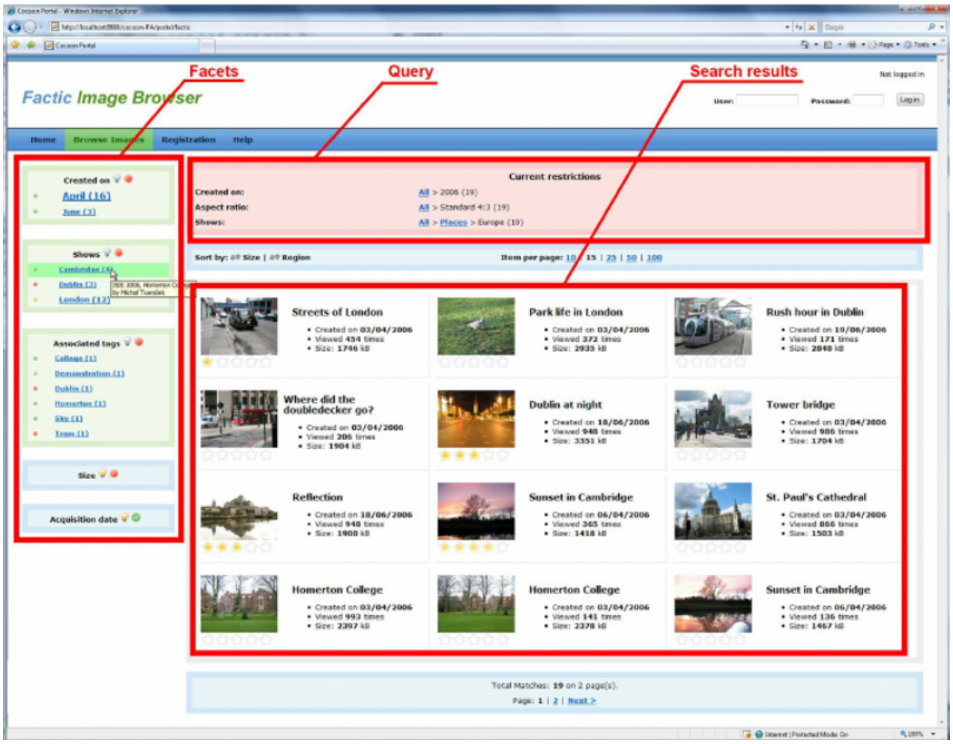


Figure 2.5: Factic's interface [115]. The initial query can be explored with the available facets of the data, which are for example tags, date or location of creation. Along with the thumbnail, a set of metadata is presented for each result.

2.2 User Modeling

In HCI, user modeling [38] is utilized to personalize the user experience by adapting the system's behaviour to suit the person's needs. This can be achieved by modeling anything from the user's skills or knowledge base, to their interests and demographic. By correctly predicting what the user needs at a given time, the system can speed up the search the user is undertaking, and avoid inefficient use.

When finding deciding factors for the model, data on the user can be gathered from multiple sources. Direct questions at the start of the session may yield quick adjustment in performance, such as asking for demographic information or their knowledge level on a given topic. For example in Factic [115], the various facets help in personalizing the search for the user, hence modeling their need better for the session. Each data source depends on both the available data, but also the purpose of the system. Users in certain settings will not tolerate long wait times, nor extended interaction with the interface. Thus, a less invasive way is to simply observe their behaviour, such as elapsed time on a page to clicked links.

The system may then adapt at a simple level, such as delivering particular search results for a certain user, or system-wide where every aspect in the environment is modified. The model of the user may be static, where once the profile is generated it no longer changes, or dynamic, where a lifelong adaptation occurs. For example, the same system could first act as an information retrieval browser for the available content and later on as a visualization tool that allows the user to better reflect on the found data.

Similarly, the *personalization* model may be based on stereotypes or classes of previous users, or be highly adaptive to a single user, making no assumptions from outside. In rule-based decision making, simple IF-THEN clauses are used to navigate the specific scenarios. On the other hand, regression models [80] may be used to track the user on a continuous valence. *Collaborative filtering* allows the comparison of different users' usage histories, which then indicates successful paths for the system when guiding the session.

Finally, the system may evaluate the user's *intent* in a given session. An example of this can be seen in SciNet [98], where the user's intent is modeled by their fine-tuning of the keywords. Having feeling of control has been shown to give users a sense of empowerment and satisfaction, thus helping them interact with the system until the desired results can be achieved.

Personalization Personalization [94] is a key component in many recommender systems, advertisement and social media in general; anywhere where tracking the user's static preferences may help tailor the service for them. When personalizing their experience, the system may target any visible aspect from the user.

Behaviour during the visitation, as well as contextual information from the entry location or time may be used to guess the needs and type of the user. Also, utilizing the actual history of the user (for example from external site cookies), or any similar user for the matter (collaborative filtering), gives an even more detailed profile. Such side information is gathered from any and all sources where a single user can be tracked. Various patterns of interest are followed, and at the best, every user gets assigned to a profile as soon as they enter the domain.

Using personalized data in explorative search has been a broad topic. For example, one study did prototyping to visualize Google search history using Timeline JS [13]. Their system collected data explicitly about the user's gender, age and relevance of the shown documents, while the user's knowledge on the issue was implicitly conveyed from their behaviour. In [8] Athukorala et al. devised a user study which analyzes how different exploration rates affect search performance and user satisfaction. Based on this, they presented a framework that recognizes whether the user is performing a lookup search, or an exploratory search [10].

Adapting to the user is nowadays one of the key components of search engines, and development of these methodologies has focused on inferring as much as possible from the little data available.

Recommender Systems Recommender systems [93] are specifically relevant in personalization of information retrieval scenarios. The purpose of such systems is to predict each user's preference, which can be used to recommend items for them. Profiles of similar interests and personal information are gathered en masse for this purpose, and then used to bring relevant objects for the user.

One of the most classical recommendation systems is for films (IMBD) [3], where entertainment preferences are tracked to match people with similar interests, and then recommend yet unseen movies to others. These recommenders can be seen ubiquitously elsewhere in the Web, from online shopping centers [1] to navigation (restaurants, shops, museums), life services (finance and insurance), news and online dating [4].

Recommendation is often based on the assumption that like-minded people exist and are prevalent throughout the society. When person A

likes a certain set of two movies, and another person B likes one of them, it is likely that B would also think similarly of the other movie. When this process is utilized over thousands of users and tens of thousands of objects, the statistical significance of profiling emerges. This allows the recommender systems to make more informed decisions when showing items to the users. And even if a large number of recommended objects would not be acted upon (such as in an advertisement), even a few successful recommendations can make or break an industry.

One major problem and subject of research is managing a *cold start* [93]. This is the issue of populating a recommendation list before anything is known of the user, such as the first time a user enters the service, there is no information regarding their preferences. Many solutions recommend the most likely set of items as their default set, such as news sites that show the user sports and generic global news.

Collaborative Tagging/Filtering Sharing information socially with collaborative tagging [38] has been recently researched as a way to navigate unlabelled content. As web users tag and document content as they pass by it, they produce valuable meta-data for future visitors and seekers. This information is collected in many sources explicitly, forming areas of expertise that are solely based on the thriving community, easing the management needs of the administrators. One such example is Wikipedia [5], an online free encyclopedia that is open for anyone to peruse and edit.

These sites also help people to add meta-data and personification profiles to often used data sets, helping new users navigate through the content. For example, collaborating in a search space was done with ResultSpace [28], which is an asynchronous collaborative information retrieval tool. Awareness is supported with a display for the history of queries, which is further enhanced by an aggregation of their ratings from other collaborators.

The same can be done for implicit user data too, well showcased in [52], where Hu et al. proposed a factor model for implicit feedback. They found unique properties from this feedback, which the factor model was able to formulate into explanations for previous recommendations.

Social tagging has been found to help exploratory search considerably. In [58] Kang et al. found that social tagging helped novices and experts alike. These tags were still more easily interpretable for the experts, but showed an overall improvement for both groups. This was presumed to be due to the community struggling and understanding similar problem descriptions together. Still, experts were found to have a clear advantage

in interpreting social tags. Their research highlights the interaction between knowledge-in-the-head and knowledge-in-the-social-web, and how domain expertise is an ever important aspect in the world.

One of the classical examples that rely on crowd-sourcing, a form of collaborative tagging, is Amazon's Mechanical Turk [26], which is a popular tool for public information refinement. In this service, customers may recruit the public to annotate data in a manner suitable for their needs, in exchange for a small monetary payment. It has been widely used to tag anything from images to videos and audio, adding information on their semantics and empowering large datasets. These have been tasks that are notoriously hard for algorithms to annotate.

Intent Modeling There is often a large discrepancy between what the user wants to search for, and how they end up formulating it for the search engine [113]. The target of the search also tends to change during the search session [48], as the user learns from intermediate results. As the user's knowledge of the topic increases, the target shifts closer to what they really want. Furthermore, the user's frustration is a concrete problem, affecting the time and resources spent on the search [48]. Even the best search engines might not get the chance to prove useful, if the user feels helpless with the tools provided.

Intent modeling [48] is a design that targets these situations in particular, giving the system a way to react to the current needs of the user. The user is better able to express their interest one way or another, usually through a valence or set of features. The interface also plays a crucial role here, as the data has to be accessibly presented for the user to understand how they are affecting it with their choices. Intent can also be modelled behind the scenes, capturing the particular information needs of the user for the session, and recognizing what their current goal is. At best, a good intent model will let the user feel in power of their choices, advocating easy and fast response to their actions.

Intent Radar [99], as seen in Figure 2.3, was designed to give users control over their search intent with intuitive and simple tools. The Radar presents the user with a spatial view of keywords relevant to their initial query. Keywords closer to the center are seen as more relevant, while similarity was measured as the angle. Users were allowed to modify the keywords as per their intent, which changed the view of relevant items. Moving the keywords closer to the middle, for example, showed interest in that item, which in turn prompted the system to recalculate a new orientation for the results as indicated by the new keyword constellation.

The system was compared to a simple list-type interface in task-based exploratory experiments. The user's task performance was better than in conventional interfaces, as seen from expert evaluations and the number of bookmarked items. This was attributed to the increased quality of displayed information, as well as improved support for directing the search.

Context Trap One more reason to advocate exploratory search is to avoid the context trap [62]. The context trap happens, when a search ends up in a dead end, from which the user cannot break free during the session anymore. In classical systems, this happens when the user has specified their feedback so far that the system is able to recommend only a restricted list of items, and no amount of new feedback lets the user escape from this context. Instead, they must restart the search, and any relevance feedback they gave until up to that point will be lost.

The problem is, common search engines tend to converge towards the correct solution over the search session, attempting to show only the most relevant information at each iteration. For this to work, the procedure requires the full attention of the user, enough patience that the convergence can happen, and enough knowledge on the topic so that each iteration the feedback is correct.

If on the other hand the user was not giving perfect feedback towards the goal they were trying to achieve, the search ends up in the wrong place of the search space. Now if the engine only retrieves the most relevant images given the feedback, stagnation of results occurs, and the user can no longer escape from the results they have at present. Managing the balance between greedy relevance exploitation and exploratory search is always a tough problem.

2.3 The Exploration - Exploitation Dilemma

One of key tenets of exploratory search is the diversity of results during a search session. Pure exploration is not interesting alone though, as maximizing the diversity of results would end up in a minimal number of relevant documents. This is why it is important to know when to keep exploring, and when to stop and utilize the resulting knowledge of what is relevant. This is called the exploration - exploitation dilemma [111], which describes the problem of choosing between exploring for better reward options, or playing the empirically best option we know thus far. The question is sometimes simplified to: when do we know enough of the problem at hand to start exploiting the system for the best rewards?

Solutions that consider exploration - exploitation tend to be utilized in the fields of recommendations and information retrieval, where the user's interests are initially unknown to the system. Tasks from filtering [127] to ad placement [85] and recommendation [15] itself have a large search space that requires mapping for lucrative matches.

One formalization for this dilemma is the multi-armed bandit environment [111], which has been used as a basis when developing exploratory systems. This method is well known both in statistics and in reinforcement learning, having a theoretically strong background there. In multi-armed bandits, an agent is presented with K one-armed bandits, each with unknown reward distributions. The agent has a certain number of game tokens to spend, which defines a horizon until which the game may continue. The agent's task is to maximize the reward from the bandits, but to do so it has to figure out which bandit yields the most reward. By spending several tokens in each bandit, the underlying reward distribution can be asserted, but certainty has to be balanced with the remaining tokens. Once the agent knows where the best rewards can be found, it can exploit that bandit for the remainder of the tokens. In exploratory search domains, a common interpretation is that the bandits represent the various recommendable items, the tokens are the act of recommendation and the horizon is measured as the user's patience.

2.3.1 Multi-armed Bandits

Multi-armed bandits [96] is the formalization of the exploration - exploitation dilemma, which by now consists of years of theoretical and practical background, and are popular even today. These methods have often been used in ranking [91], where they provide a solid framework for managing exploration.

Popularization of bandits has lead to multiple advances in information retrieval settings through the years, such as Dueling Bandits [125], which is based on pairwise comparisons that worked well with merely implicit feedback. The algorithm attained sublinear regret, as well as worked with discontinuous rankings, all features that show great promise in online settings.

Another such system was introduced by Radlinski et al. [91], which was an online ranking engine that maximizes the probability of finding a relevant document in the top k documents, and attains a polynomial total payoff while doing so. All of this happens without labels in the dataset, and by utilizing the similarities between documents and user feedback. Systems like these optimize the ranking of documents both for relevance and

diversity, granting users a more thorough set of items from the start.

Formally, K-armed bandits is defined by a set of random variables $R_{i,n}$, for $1 \leq i \leq K$ and $1 \leq n$, where i corresponds to a possible action (pulling an arm) and n corresponds to the number of times any arm has been tried. Pulling an arm i yields rewards $R_{i,1}, R_{i,2}, ...R_{i,n}$ which are i.i.d. across i and n, and are rewards from an unknown distribution and unknown expectation μ_i. The objective is to modify the policy to first learn the distributions of the variables (exploration), and at the same time maximize the expected reward (exploitation).

Let $t_i(n)$ be the number of times an arm has been pulled over the first n rounds. Now we may measure the regret after n rounds by

$$Regret(n) = \mu^* n - \mu_j \sum_{j=1}^{K} \mathbb{E}[t_j(n)], \tag{2.1}$$

where we define $\mu^* = \max_{1 \leq i \leq K} \mu_i$, and $\mathbb{E}$ is the expectation. Regret is thus the *expected loss* for not pulling the best known arm, given our current knowledge at the time. It was further proven by Lai and Robbins [67] that a lower bound for regret exists for any policy, and that the regret of the best algorithm is of the order $O(K \log(n))$. These bounds were later tightened on several attempts, most notably with Upper Confidence Bounds [11].

There are several exploratory paradigms for bandits, such as ϵ-greedy or explore-first tactics [120], which solve the problem adequately in most scenarios. ϵ-greedy picks the best choice it knows of with $1 - \epsilon | \epsilon \in [0, 1]$ chance, and takes a random option otherwise. Explore-first explores for the first l rounds, where $l < K$, to form an initial understanding of the dataset. Neither method comes with strong guarantees for success, as the parameters are independent of the environment's properties, and indeed, of what the algorithm discovers during exploration.

Upper Confidence Bounds A classical way to balance exploration and exploitation is to use Upper Confidence Bounds (UCB) [11], a method that further elaborates on the bounds defined above. By tracking the confidence of expected reward with confidence bounds, it is possible to solve the interplay of exploration and expected reward optimistically in an analytic form. Practically, this optimism is seen as the method assuming that any unexplored data point may contain maximal rewards. As exploration progresses, this assumption is relaxed towards the true expected reward, hence lowering the chance of these data points being tried again due to uncertainty.

Confidence bounds are defined by a normal distribution around a certain mean μ and variance σ, such that one tail of variance is the upper bound and the other the lower bound. A certain portion of the population will lie between these two bounds, ensuring that at high probability, using these bounds and the mean make for a powerful tool. The true expected reward falls into this confidence interval with overwhelming probability.

Confidence bounds define the statistically significant interval around the most likely outcome in the following manner:

$$CB = \mu_{\pm} Z_{\alpha/2} \cdot \frac{\sigma}{\sqrt{n}}, \tag{2.2}$$

where μ is the mean, α is the confidence level, σ the standard deviation, and n the sample size. Together $Z_{\alpha/2}$ defines the confidence coefficient.

UCB is based on a one-sided confidence bound, where only the upper bound is considered. The results in [11] show that there exists an allocation strategy, UCB1 (Algorithm 1), which achieves logarithmic regret uniformly over n and without any preliminary knowledge about the reward distributions. It is the sum of two terms, namely the current average reward and a one-sided confidence interval for the average reward which to expect with high probability.

Algorithm 1 UCB1 algorithm introduced by Auer et al. [11].

1: Initialization: play each arm j once.
2: **for** each t in n **do**
3: Play the arm j that maximizes

$$x_j + \sqrt{\frac{2 \ln n}{n_j}},$$

where x_j is the average reward obtained from arm j, n_j is the number of times arm j has been pulled thus far, and n is the number of pulls done in total thus far.
4: **end for**

A decision may be made by looking at the highest upper bound, giving us a well informed decision, combining both the expected reward and uncertainty under optimism. The point with the highest upper bound may then be used directly as the next prediction, as it balances the exploration and exploitation. UCB has uniform logarithmic regret for any set of reward distributions that elicit a known bounded support.

2.3.2 Contextual Bandits

Many practical applications have access to additional information, which can be used for better decision making within bandit settings. This is especially true in advertising content, where customers' behaviour can be monitored while they browse. These marketing strategies are visible everywhere from search engines and social media streams to any website which is based on ad revenue.

Utilizing this information was studied in Contextual Bandits [117], where the side information, or context, is considered as the features associated with the reward gained. This context was structured from the clicks, visitation times, website history or location. The model is a great match for online settings where search queries can be understood as the context, and relevant ads to be shown are the multi-armed bandit problem.

For example, in [70] a general contextual bandit was proposed for personalized news recommendation, which could be evaluated efficiently offline, based on previously recorded traffic. Their method reached a 12.5% click lift compared to the standard context-free bandit algorithm on Yahoo! Front Page Today Module dataset. Another work addressing the exploration - exploitation trade-off used Thompson sampling [29]. It was tested on the same news article recommendation problem, as well as displaying advertisement, where they reached higher click-through-rates than the previous state of the art, such as UCB. In both systems, accounting better for the context gives additional information on the needs of the user.

LinRel LinRel [12] was designed for solving exploitation-exploration decisions based on uncertain information, as an extension to upper confidence bounds. It is a deterministic algorithm that utilizes linear value functions and confidence bounds to measure the uncertainty of the environment.

Unlike in classical bandit scenarios, LinRel considers side information associated with each arm i in the form of a feature vector $X \in \mathbb{R}$, which describes the expected reward. It is assumed that there exists an unknown vector $f \in \mathbb{R}^d$, which is fixed and describes the relation between the feature vector and the expected reward, $f \cdot x_i(t) = \mathbb{E}[r_i(t)]$ for all $i \in \{1, ..., K\}$ and $t \in \{1, ..., T\}$.

LinRel (as seen in Algorithm 2) utilizes features X, which are listed as a matrix, where each column $x_i(t) \in \mathbb{R}^{d \times 1}$ represents a feature vector and $\Delta(\lambda_1, ..., \lambda_d)$ represents the diagonal matrix. In the eigenvalue decomposition $X(t) \cdot X(t)'$, $\lambda_1(t), ..., \lambda_k(t) \geq 1$, $\lambda_{k+1}(t), ..., \lambda_d(t) < 1$, and $U(t)' \cdot U(t) = \Delta(1, ..., 1)$. The purpose is to learn the upper confidence bounds for the means $\mathbb{E}[r_i(t)] = f \cdot x_i(t)$ from the weighted sum of the

Algorithm 2 LinRel algorithm, as introduced by Auer [12]. The aim is to learn the upper confidence bounds for the means of the weighted sum of previous rewards. From here the highest option is chosen as the next solution.

Parameters: $\delta \in [0, 1]$, the number of trials T.
Inputs: Selected features, $\psi(t) \subseteq \{1, ..., t-1\}$,
new features, $x_1(t), ..., x_K(t)$.

1. Let $X(m) = (x_{i(\Gamma)}(\Gamma))_{\Gamma \in \psi(t)}$ be the matrix of selected features and $R(m) = (r_{i(\Gamma)}(\Gamma))_{\Gamma \in \psi(t)}$ the vector of corresponding rewards.
2. Calculate the eigenvalue decomposition $X(t) \cdot X(t)' = U(t)' \cdot \Delta(\lambda_1(t), ..., \lambda_d(t)) \cdot U(t)$
3. For each $x_i(t)$ set $\hat{x}_i(t) = (\hat{x}_{i,1}(t), ..., \hat{x}_{i,d}(t))' = U(t) \cdot x_i(t)$ and $\hat{u}_i(t) = (\hat{x}_{i,1}(t), ..., \hat{x}_{i,k}(t), 0, ...)'$, $\hat{v}_i(t) = (0, ..., 0, \hat{x}_{i,k+1}(t), ..., \hat{x}_{i,d}(t))'$.
4. Calculate $a_i(t) = \hat{u}_i(t)' \cdot \Delta\big(\frac{1}{\lambda_1(t)}, ..., \frac{1}{\lambda_k(t)}, 0, ..., 0\big) \cdot U(t) \cdot X(t)$.
5. Calculate the upper confidence bounds and its widths, $i = 1, ..., K$, $width_i(t) = \|a_i(t)\|(\sqrt{\ln(2TK/\delta)} + \|\hat{v}_i(t)\|$, $ucb_i(t) = r(t) \cdot a_i(t)' + width_i(t)$.
6. Choose the alternative $i(t)$ which maximizes $ucb_i(t)$.

previous rewards. Thus, a feature vector $x_i(t)$ is a linear combination of some previously chosen feature vectors $x_i(\Gamma)$, where $\Gamma \in \psi \subseteq \{1, ..., t-1\}$,

$$x_i(t) = \sum_{\Gamma \in \psi(t)} a_i(\Gamma) x_{i(\Gamma)}(\Gamma) = X(m) \cdot a_i(m)' \tag{2.3}$$

for some $a_i(m) \in \mathbb{R}^{1 \times |\psi(t)|}$, where $X(m)$ is a matrix of previously chosen feature vectors.

Now, the exploration-exploitation trade-off can be managed by offering the largest upper confidence bound as the optimal solution. The exploitation is controlled by the estimation of the mean, while exploration is controlled by the width of the confidence interval.

Gaussian Process Bandits Gaussian Process Bandits (GPB) [36, 109] is a kernel method for bridging UCB and Gaussian Processes for tasks that have expensive objective functions. It also has a strong and well defined theoretic background, which has made it a popular topic for research [47, 108].

Gaussian Processes are used to model the reward distributions, such that each bandit is a point in the Gaussian Process. Kernels are used to

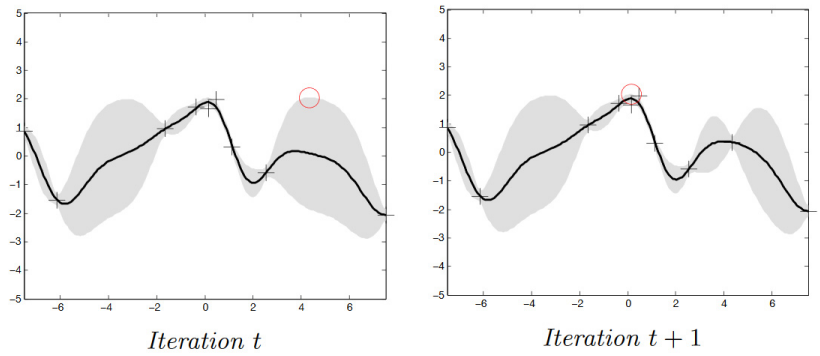


Figure 2.6: Gaussian Process Bandits [109] updates the neighbourhood around a new observation (circled red).

express pair-wise similarities in a feature space when updating the model. The similarity is measured as an expression of the data representations in the input space.

Similarly to Upper Confidence Bounds, in GPB the mean and variance are treated as the confidence bounds (see Figure 2.6). For the bandits setup, the upper bound gives a natural best guess for the optimal move, as it combines the information theoretic uncertainty and the estimate of a good reward into a single prediction. If a point is already explored and yields a high reward, then nearby points will be updated to be in higher uncertainty, relative to the distance. Thus, they will be explored sooner than regions where a nearby result has low reward.

Gaussian Process is then updated over the nearby region in the following manner: Pulling an arm will collapse the confidence bounds for an observation, at the same time affecting nearby, yet unobserved arms relative to their distance. In Figure 2.6 we can see how the uncertainty crashes around an updated point, as the knowledge is propagated to similar entries near the seen point. As more and more points are seen, the landscape starts to form towards the true probability function. There are proven regret bounds for Gaussian Processes for UCB (GP-UCB), which are sublinear for GPB optimization with commonly used kernels [47, 108].

The GP-UCB algorithm (Algorithm 3) combines Gaussian Processes with the UCB algorithm. Each point in the input space D corresponds to a Gaussian Process, with mean μ and variance σ. The variance and mean are set to maximum, for example in reinforcement learning the upper and lower bounds (tails of variance) to maximum and minimum rewards.

Each update changes all entries by the relation of their distance to the

Algorithm 3 GP-UCB algorithm introduced by [109]. The bandits are formulated as a joint distribution of variables $x \in X$, each regarded as a Gaussian distribution.

1: INPUT: Input space D: GP Prior $\mu_0 = 0$, σ_0, k
2: **for** each t in n **do**
3: Choose $x_t =_{x \in D} \mu_{t-1}(x) + \sqrt{\beta_t}\sigma_{t-1}(x)$
4: Sample $y_t = f(x_t) + \epsilon_t$
5: Perform Bayesian update to obtain μ_t and σ_t
6: **end for**

seen object. The updated item itself will have zero variance and the mean is set to y_t itself. Other items gain a relative change depending on their distance, with variance greater than zero.

Several improvements have been made on GP-UCB over the years, such as in Contextual bandits [117], which were devised to have a better representation of the mapped feature vector. In it a context vector is maintained at each iteration of the search, containing the current features associated with the shown content. This feature vector is then mapped to the gained reward, such that the context of each feature is learned over the session.

2.4 Challenges and Future Research

Exploratory search has recently received interest from multiple research communities related to information retrieval. The commonly seen query-response paradigm is becoming increasingly inefficient, as people wish to explore domains outside of their expertise more often. Exploratory search is a natural extension for the inquisitive behaviour people have a penchant for. One of the future challenges is to better define measures of exploration, both analytical and quantitative. At the moment, there are next to no objective functions that measure the quality of exploration itself, such as its coverage or expressibility. All studies have focused on the statistical analysis of subjective experience by the users, which although is needed for an end-product, is a bottleneck for quick development of new methodologies: Collecting new data from user studies is always time consuming and requires careful planning to succeed.

Assessing Exploratory Search Classical information retrieval systems rarely support exploratory search tasks. This is partially due to, and the cause of, having little empirical backing on how exploratory search and

lookup search differ from each other. The distinction is hard also because the user studies implemented in the field are applicable for regular search scenarios too.

Athukorala et al. [10] highlighted this difference by measuring behaviour patterns. With six search tasks, three exploratory and three lookup tasks, they showed that these tasks can be differentiated. This illustrates meaningful indicators for detection of exploratory tasks, which is an important aspect for cohesive study design. In their studies, the best indicators were query length, maximum scroll depth and task completion time: the longer these were, the more likely the associated task was exploratory in nature.

Another study was performed by Liu et al. [71] where they found that exploratory search is often associated with long query lengths, continuous query reformulations, careful search result evaluation and numerous search results click-through. They generated a multivariate regression model for tracking search interaction performance, which combined user search experiments, search query log analysis and search system development.

Additionally, in Publication V I have suggested a quantifiable coverage score that shows how widely a metric space has been explored. It allows for a search engine to know how much a certain set of retrieved objects covers from the whole dataset, and thus a way to optimize a system to increase exploration.

Assessment of exploration in search is a fledgling field, and is often approached by either user studies or classical content-based retrieval. We will theorize on its future over the span of this dissertation on a few occasions, and highlight lucrative opportunities on how to further the field.

Chapter 3

Relevance Feedback

Relevance feedback [14] is often used in information retrieval systems to evaluate the value of items for the user. This is done by extrapolating from existing knowledge onto items that have not yet been seen by the user, allowing more efficient retrieval. The various brands of feedback give different types of information, requiring the application of methodologies ranging from reinforcement learning to feature selection, and may be used to better understand the information needs of the user.

Feedback itself can be in any form from the explicit user score to the implicit visitation time [14]. For example, once a particular facet in the data is found [124], it is possible to show relevant results from the correct perspective. Tuning down to the correct subset of data as the user interacts with the system can be challenging.

One key challenge in information retrieval systems has been the users' ability to affect the search. The feeling of having control over the feedback's interpretation is important, as this is how the user is able to adjust and then direct the search. One problem stems from systems that take the user as a passive source for relevance feedback. In these cases both the user profiling and the state of search are seen as a single entity. This was further studied in [60], where the user was given extensive control over their environment. Their system interprets the user's action as having an intent towards a goal, mitigating the chance of misinterpretation. User studies indicate their system gives a better control over the search procedure, which in turn motivates the user to further the search.

3.1 Implicit and Explicit Feedback

User data collection in relevance feedback can be classically divided into two major categories, implicit and explicit [62]. Both methodologies have their advantages and disadvantages, but most contemporary systems tend to utilize them both. Hence, it is important to know what both are capable of, and how to interpret them given the interface, the user, and the engine's needs.

The most practical explicit feedback collection can be done when the user is directly queried for their preference, via questionnaires or simple text fields. More advanced interfaces supply various gauges [76], 2D or 3D representations of the search space [99] or tools to extract relevant features from the documents [76, 130]. Often useful in standard search, the user can fill in the query fields or give likes or numeric values to their preferences, and thus tell the system what is relevant in the dataset, or otherwise communicate their intent.

Explicit feedback is often easy to attain, is usually reliable and gives the user a sense of control, which further facilitates their search experience. The downside is that a lot of information that could have been used from implicit interaction, thus requiring extra effort on the part of the user to get on the same level. Users often get tired of giving lengthy amounts of feedback, often opting to stop after a few iterations of interaction [48]. Finally, the user might not always even know what they are really looking for, hence their feedback might be faulty, biased or lack information content regarding relevance.

Implicit feedback [14, 56], the more recent counterpart to the dichotomy, covers the usage of side information and behavioural cues to understand the user's needs. Here various hints can be gathered from the user's time spent on pieces of information, clicks or other ways to collect data on their attention. For example, implicit feedback may come in the form of gaze detection, reading time, scroll time, or any other cue that the user gives during their visit. With this data, it is possible to evaluate the user's acknowledgement of the data, their interests and needs.

Implicit information is easy to collect, and happens all the time without the user's direction. It is ubiquitous both where and when it appears, and may be exploited fast and easily. Often, the opportunity cost of using implicit feedback is also low, such as when the user has to only like a particular document, and the system then figures out what features were relevant for them. Such a system does not need specialized interfaces to work, nor does it take time from the user when giving the feedback. Existing machine learning methods can in theory extract enough data from such

feedback for informed decisions.

Unfortunately, implicit feedback at the same time tends to be very unreliable in what can be inferred from it. A system such as described above would need several iterations of feedback, before the correct subset of features could be targeted. An explicit targeting from the user would have given the information far earlier. Similarly, the user's gaze and attention may linger on an object or a page not because they are interested in it, but because they are lost in thought or fail to grasp what they are looking at. Clicks might be similarly missclicks, scrolling time may be different due to poorly functioning apparati, and making too precise predictions based on them is hard. Implicit feedback has to be treated as noisy information at all times.

Pseudo / Blind Feedback Pseudo-feedback [14], or blind feedback, is a method to automate the relevance measurement process locally for a single user. It removes the manual part of giving feedback, which improves retrieval performance, while allowing the user to skip some of the interaction with the system.

This methodology starts out with a regular retrieval setup, but after the first round of user supplied feedback, the system assumes that a number of the top ranked documents will be relevant without verification. These documents are then used as a basis for the following iterations, for example by extracting the most prominent features, and then propagating their relevance to the rest of the dataset. As many classical retrieval methods tend to have a high recall already, this assumption allows the system to bypass unnecessary steps with feedback acquisition.

There is compelling evidence that pseudo-feedback improves on global feedback techniques. For example, when developing the classical Cornell SMART system by Buckley et al. [25], they found out that on the TREC 4 dataset pseudo-feedback gave a rise of approximately 10% in retrieval precision, when compared to similar methods without pseudo-feedback. Results like this support the addition of pseudo-feedback as a tool for systems that need an edge in response time.

Negative Feedback The most common user behaviour often contains only positive instances of feedback, as the users' actions rarely indicate which products they actively disliked during their visit. This makes the data extremely one-sided, which is a problem for most machine learning methods that would learn to model the user.

There are several ways to augment this lack of negative information,

from making it easier to supply feedback, to making assumptions on what constitutes as an unwanted feature for the user, or even simply inferring more from missing data. For example, in [52] Hu et al. suggest a factor model to better include implicit negative feedback in the user modeling. They considered not only the valence for preference, but also confidence, which the factoring uses for prediction. Their method was applied for large-scale TV show recommendation, where surprisingly good levels of recall were seen for difficult situations.

Another work that studied negative feedback was done by Peltonen et al. [86], in which they introduced a continuous-value feedback system for an exploratory document retrieval engine. The interface supports both positive and negative feedback in an intuitive setting, as seen above in Figure 2.3. The studies showed superior performance over state-of-the-art positive feedback only systems, illuminating the information value of negative feedback.

3.2 Ranking with Relevance Feedback

Ranking of search items according to relevance feedback is one concrete way of formulating a response for information retrieval systems [14]. Ranking is a list that contains the search algorithms idea of best fitting items for that particular query, and the ordering may be used directly to populate the page with results for the user. Once a listing is done, each consecutive page can be generated simply by going further down the ranks.

There are several venues to expand upon with the inclusion of relevance feedback in ranking. For example, the original query might have already been exhausted of relevant hits after the first iteration, as the items down the list might be based on only a few relevant features. Furthermore, retrieval systems tend to have a limited capacity for relevant document recall. Only a subset of all relevant documents can be recalled at once, depending on the completeness of the original query. Thus it is often important to further refine the original query, such as by adjusting the term weights with information gleaned from relevance feedback.

Query Expansion One early development for refining ranking was query expansion [14], which populates new entries to the ranking list for the next iteration. Query expansion takes the user's original query, any other feedback there is available, and then generates a new query that furthers the search. For example, if the user queries the system with the word "dog", they often want to query with the word "hound", too. The goal

of this is to increase the recall of relevant items, although trading this for overall precision.

Query expansion is one way to avoid falling into the context trap [62], as the original query might be too narrow, or lead to irrelevant regions in the search space. Thus, instead of needing the user to redefine the query, the system can automatically expand on it and populate the response. Query expansion may also be made with several different methods [14], from local clustering to context analysis or term reweighting.

3.3 Ranking with Vector Space Models

Vector space models [14, 24] are algebraic constructs that form similarity measures between the objects of datasets. In these models, each document is represented as a vector of features, often generated in a domain-specific manner. This is a popular method in information retrieval, where a vector space can be used to propagate relevance from one seen object to all unseen objects. The user simply has to give feedback to a single entity in the dataset, and a natural ranking for all items can be generated by measuring the distance to this object.

When performing ranking, the given distance measure can be used to find items that are close to a relevant item, which are then assumed to be more relevant than distant items. Finding the right distance measure has to be based on the features that best describe the dataset. This also constrains the search to be effective only in the given description. Still, it is also possible to relearn the feature representation to fit the relevance information of the present user.

Cosine Similarity Cosine similarity [106] is a simple yet effective metric for measuring similarity, as it gives a measure of similarity between two data points that can be easily calculated in online settings. Cosine similarity is often used in text vector spaces, where the vector consists of, for example, the $TF - IDF$ vectors of the dataset [102].

The cosine similarity $\sim (A, B) = \cos(\theta)$ of two non-zero points, A and B from an inner product space, and may be calculated with the following formula:

$$\cos(\theta) = \frac{A \cdot B}{\|A\|_2 \|B\|_2} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}, \tag{3.1}$$

where A and B are vectors, with i indexing their components. Now, depending on angle between them, similarity is often interpreted to be either

maximally dissimilar, when $\cos(\Theta) = 0$, i.e. when $\Theta = 90°$, or maximally similar, when $\Theta = 0°$). When two vectors that are at a 90 degree angle to each other, they share no similar features between them.

Rocchio's algorithm Rocchio's algorithm [101] is one of the classical methods for creating ranking from relevance feedback, also stemming from the SMART research back from the 60's [24]. It is a method for generating queries from vector space models, which later became a popular baseline for many information retrieval systems.

Rocchio calculates a new query vector for each iteration, by which the next set of retrieved documents may be presented. It assumes that the user has approximate knowledge of which documents are relevant and which not. Rocchio's query vector is calculated as a sum of three sub-products:

$$\overrightarrow{Q_m} = (a \cdot \overrightarrow{Q_o}) + \left(b \cdot \frac{1}{\|D_r\|} \cdot \sum_{\overrightarrow{D_j} \in D_r} \overrightarrow{D_j} \right) - \left(c \cdot \frac{1}{\|D_{nr}\|} \cdot \sum_{\overrightarrow{D_k} \in D_{nr}} \overrightarrow{D_k} \right), \quad (3.2)$$

where $\overrightarrow{Q}$ is the query vector with m being the new, modified one and o the original one. The parameters a, b and c are weights by which the relevance is modified, D is the document vector, with j being the related documents, k the non-related ones and D_r and D_{nr} their respective sets.

In summary, the new modified query vector is recalculated such that the set of related documents move closer towards each other, while non-related documents move away from each other. The velocity of this change is directly affected by the three constants, for example a high c includes more non-relevant documents for the next iteration. One classical parametrization used extensively in the literature is $a = 1.0$, $b = 0.8$ and $c = 0.1$.

Chapter 4

Content-Based Information Retrieval

The increasing diversity of digital multimedia has forced the evolution of common information retrieval engines. Simple, text-based data can be easily queried with keywords, but what about unannotated photos, videos or music? The need to make such items available led to the advent of Content-Based Information Retrieval (CBIR) [69], which lets the user search for information with full or partial examples of what they are looking for. These systems do not require extensive analysis of the database, but rather descriptive feature representations that find semantically important aspects from the data.

To get salient, informative features associated with data items, the items usually need to be processed in a low level, such as pixels in an image or words in a document, that fits the domain in question. These feature representations require careful construction and planning, as the choice affects the possible targets the users can access.

Often, a higher level of abstraction has to be constructed from the initial features, which in turn explains them as a part of some semantic phenomenon. For example, Content-Based Image Retrieval [61] started out with combining pixels into shapes, textures and edges, forming higher level, local feature representations for bag-of-words models [123].

To get to a proper, comprehensible representation, several more layers of abstraction are often needed, after which these have to be translated to a summary of what the item stands for in terms of queryable words. Still, it is possible to use this preprocessing directly to figure out the relevant set of items for the user. It is not always necessary to identify and exhaustively explain what a particular item is, if the features underneath are directly used to make the decision.

Unfortunately, low-level feature representations come with a price for their cheap generation, as a semantic gap may emerge between them and the higher, human-level perception. This is why development has moved towards more advanced methodologies, such as more refined similarity measures and feature representations [118].

Similarity Measures Content-based information retrieval requires a representation that captures the essence of a given database, often found as a feature representation specifically made for the domain, or even for the dataset itself. One option is to utilize metric learning [16], which creates a distance function between any two items in the dataset to measure their similarity. This distance can be directly utilized as relevance: the user chooses one item from the dataset, and immediately every unseen item can be ranked according to their similarity to this item.

Metric learning methodologies have seen more and more success over the years, often becoming a key enabler for information retrieval tools [16, 51, 78]. Their strength is in a descriptive feature representation, where a metric can often be taught based on previous user feedback, or semantic knowledge of one form or another. These methods are often straightforward projections into the chosen metric, giving a fast and easy way to generate them.

There are several ways to generate similarity measures, often depending on the domain's parameters. One option is to form the metric from the pairwise distance [16] of any two items, where an external evaluation function tells whether two items are similar or dissimilar, which is then, on the basis of the items' features, used to learn the distance function. This works especially well when the domain consists of a free-form set of unrelated features, with no underlying structure present previously. This function may be the cosine similarity, a classic metric such as the Euclidean distance, or a measure specifically suited for the problem, such as the Mahalanobis distance [73] or the Hamming distance [83].

Deep learning offers a dynamic and robust environment for learning new metrics. Utilizing similarity measures with neural networks was seen for example in [30] and [51], where a network was trained to identify faces with different illumination, pose, obscuration or facial expression, based on a pairwise comparison of two faces. In both setups, the methods attempt to maximize the inter-class variation while minimizing the intra-class variation, which moves similar images into clusters within the target metric space. Their results worked well in face recognition tasks even in extreme conditions, such as highly varied pose, expression, lighting or position, and

showed promise in generalizing to other domains with large numbers of classes.

Features in Various Domains Successful content-based information retrieval is highly dependent on a dependable underlying feature representation. As each type of data has a specific way to formulate the features, it is imperative to have comprehensive domain knowledge when developing search engines for them.

As an example, text is often straightforward to handle on the character level, but requires extensive modeling and comprehension as the semantic meaning of the sentences is extracted. On the other hand, image features are hard to map to the semantics of the content, while after successful classification there is little need for further elaboration.

Nonetheless, one of the standing challenges for each new search engine is finding the correct feature representation that best encodes what the users might want to look for in the data. Having a structure that is precisely defined prevents the user from targeting special cases, and makes explorative search harder. If the focus is too much on abstraction, on the other hand, the accuracy of the prediction is severely reduced, making the convergence on a target hard. Having both might prove a difficult task, as it requires accumulating several feature representations and then careful interweaving them together to work. Hence, finding a good balance for the given task is important.

4.1 Features for Text

Formulating the structure, meaning and relation of textual data has been studied under the domain of natural language processing [57] for years. Systems employing various forms of natural language understanding have been widely adopted in the last decades, being a component for search engine infrastructures.

Although most natural languages conform to numerous rules, they still exhibit an infuriating number of exceptions. Working around these exceptions often requires special care in the definition of linguistic data structures, such as syntax trees. Natural language also tends to sprawl dynamically, forming recursive structures and having multiple correct interpretations at times (disambiguity). Due to this, many methodologies have relied on hand-crafted ontologies and rules from Part-Of-Speech (POS)-taggers to syntax trees and lexical semantics [57].

Machine learning has brought further advancements to the field, with the statistical properties of large datasets converting into powerful feature representations. Tools from topic modeling to language generation and sentiment analysis have all entered the field, moving dispositions towards systems that help in understanding large volumes of text [57]. Generating features out of text is often done statistically from a large corpus, often covering multiple different topics and fields.

Text features are often based on full words, although some character-level models do exist. These features are traditionally then subjected to multiple preprocessing steps, where disambiguity and low information content are removed. For example, as some words tend to appear throughout all text documents, a stop-word list is often used to exclude the ones with least information content. Salience-wise, removing the suffixes of words also helps the process.

From here, the full words may be collected into bag-of-words models [74], with which, although losing information on the ordering, we gain a descriptive feature representation for many applications. Bag-of-words models are one of the standard feature representations used in many natural language processing systems.

TF-IDF Many of the advanced methods used for understanding documents content is based on TF-IDF (Term Frequency - Inverse Document Frequency) [102], which is a numerical method for counting the occurrence of words in a set of documents. This frequency can be used to estimate what a single document represents, but also how it relates to the collection as a whole. The more mentions of single terms are found in a proportionally small number of documents, the better they describe and highlight the content common to them.

TF-IDF consists of two terms, the first being term frequency:

$$F(t,d) = 0.5 + 0.5 \cdot \frac{f_{t,d}}{\max(f_{t',d} : t' \in d)} \tag{4.1}$$

where t is the term, d the document $f_{t,d}$ the raw count of how many times the term is in the given document. The inverse document frequency is measured as:

$$IDF(t,D) = \log \frac{|D|}{|(d \in D : t \in d)|}, \tag{4.2}$$

where $|D|$ is the total number of documents in the corpus. When multiplied together, they show how strongly a single word describes a particular document: if it is found often in a small subset of documents, but rarely outside

of them, the score will be high, indicating a common theme between the documents. If a term on the other hand appears throughout the corpus, it holds little information value when trying to distinguish the documents.

TF-IDF also creates a natural distance between documents, as the TF-IDF values of a document can be used as a feature vector describing it. This feature has been used to a great effect in many occasions, such as in topic modeling, text mining or user modeling [102].

Okapi BM25 Okapi BM25 (Best Matching) was part of the Okapi probabilistic language processing suite [97], developed in London's City University in the 1980's. The original purpose of the system was to target short, similar-length abstracts and catalog records, with which it works well even today.

Okapi's BM itself is a retrieval function based on the bag-of-words model with TF-IDF, which ranks documents based on their content. It consists of a family of scoring functions (BM25, BM15, BM11, BM1, BM0), each being a different parametrization of the same function.

The BM25 score, which measures the keywords $q_1, q_2 ... q_n$ in common with the query Q, may be calculated as follows:

$$\text{score}(d, Q) = \sum_{i=1}^{n} IDF(q_i) \cdot \frac{TF(q_i, D) \cdot (k_1 + 1)}{TF(q_i, D) + k_1 \cdot (1 - b + b \cdot \frac{|d|}{avgdl})}, \qquad (4.3)$$

where $|d|$ is the length of the document and $avgdl$ the average document length. The free parameters k_1 and b are used in lower-bounding TF normalization. Now, much like in the common TF-IDF, the repetition of query words is compared against the commonality of them in the whole corpus. What is different is for one the term in the denominator, $TF + k_1 \cdot (1 - b + b \cdot \frac{|d|}{avgdl})$, which weighs repetitions as less important than differing query words. Additionally, the term $\frac{|d|}{avgdl}$ decreases the importance of lengthy documents.

By changing the parameters b and k_1, the function can be modified to change the importance of their corresponding terms. This allows one to for example ignore the importance of document length.

Topic modeling Topic models [18] are one of the most popular contemporary ways to formulate textual data statistically into semantic groups. These models formulate a corpus of text as distinct groups of discussions, or *topics*, describing the constituent parts of each individual document. Once topics are defined, multiple applications may be used to better analyze the

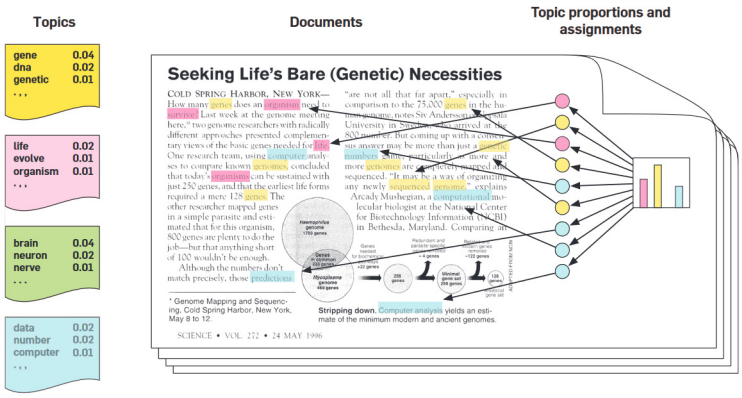


Figure 4.1: An example of the output of probabilistic topic models by Blei [19]. The topic proportions are assigned to each document based on their content, which then formulates a feature description for the document.

available data. Topic models are utilized in various information retrieval systems, where they can serve as advanced feature representations.

Much like with TF-IDF, topic consistency of a document may be thought of as a feature vector, giving each document a semantically relevant position in a vector space. Many modern topic modeling systems are based on TF-IDF, but have introduced probabilistic tools to the modeling procedure.

Such is for example Latent Dirichlet Allocation [20], as seen in Figure 4.1, which is a popular generative probabilistic model for discrete text data. LDA assumes that words arise from a mixture of k latent topics, each drawn from a distribution over the vocabulary. The joint distribution of the generative process can be written as

$$p(\beta_{1:K}, \Theta_{1:D}, z_{1:D}, w_{1:D}) = \prod_{k=1}^{K} p(\beta_i) \prod_{d=1}^{D} p(\Theta_d) \Big(\prod_{n=1}^{N} p(z_{d,n}|\Theta_d) p(w_{d,n}|\beta_{1:K}, z_{d,n}) \Big), \tag{4.4}$$

where the observed words in document d are w_d, n counting the word position in the document. $\beta_{1:K}$ is the set of topics, each β_k being a distribution over the words. The topic proportions are marked as Θ_d, and topic assignments as $z_{d,n}$.

Certain dependencies are the key of LDA, distinguishing it from other similar methods. Such are, for example, the dependency of topic assignment on topic proportions, or the dependency of observed words on the topic assignment. Distinct assumptions made by other topic modeling methods

define their usability, such as Correlated Topic Models (CTM [66]), which maximizes the correlation between topics.

4.2 Features for Images

The most popular feature representation for images has long been in the form of keywords, assigned to them manually. For example for Google [2], the side information on the websites and any textual annotations of the images has been in use for most of the search engine's existence. Unfortunately, unannotated images litter most databases, remaining unseen for efficient perusal of users. Several methodologies have since been invented to close this semantic gap between the pixels and the user's search queries.

For images, the first experiments for content-based retrieval date back to 1992 [61], where a query-by-example system was introduced for images. A variety of feature descriptors or local representations have been used to present the images, from colours to edges and textures, and furthermore with local feature representations, such as the bag-of-words models [123] in conjunction with local feature descriptors (e.g. Scale Invariant Feature Transform - SIFT based description [72]).

Since then, most of image classification methodologies have moved on to utilize neural networks [46], which work on an end-to-end basis between the images and predefined class labels. Content-based image retrieval methodologies followed on the same path [64, 118], as it was found that they can be applied in a highly dynamic manner for various tasks on the field. Another important aspect in these methodologies has been the high quality of features these networks produce. For example, it is possible to extract features from the middle layers of the network, which has been found to work well for many forms of image recognition tasks. Although the features suffer from the black-box -effect, they also benefit from it: human intervention is not required to gain state-of-the-art features for any complex problem.

Convolutional Neural Networks The latest breakthrough in visual recognition tasks has been largely thanks to convolutional neural networks (CNN, [68]), which utilize the same principles as the human visual cortex to assess visual scenes. The existing universal approximation capability of neural networks is further specialized for the visual domain, by the spatial assumption that pixels near each other correlate highly. Today, many ground-breaking discoveries in image recognition are based on these networks, be they simple classification or complex temporal activity recognition [54].

CNNs generate salient features on each layer, which resemble in their function anything from Gabor-filters on the first layers to advanced conceptual cells in later layers [6, 126]. Once trained, these features maintain strong, generic information about the visual world, and can be reused in completely different tasks later, not to mention retrained for new purposes (known as transfer learning [64, 84]). This has enabled the same methodology to be used in widely varying image recognition tasks, further spreading its popularity.

Classical CNNs are formed of a number of convolutional, max-pooling and fully-connected layers (see Figure 4.2), each containing a number of neurons chosen according to the task. These neurons are linked to a certain set of neurons on the previous and next layers by weights, which form the parameters of the learning task.

A convolutional layer has connections only on a certain filter range f, unlike in fully connected layers. This can be formulated as the convolution operation,

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n), \qquad (4.5)$$

where K is the two-dimensional kernel, I the input image, and the indices i,j,m and n denote the location of the kernel with respect to the image. In effect, when applying the convolution over an image, it does a static sliding window over it, presenting each offset of the kernel as its output to the next layer. The initialization of the network's parameters is done randomly, but as training progresses, they start to converge towards task-specific configurations.

Every neuron then has an activation function that non-linearly scales their output, based on the input. This activation function is used to propagate the presence of relevant features throughout the network, enabling differentiation necessary for gradient descent optimization. For convolutional neural networks, the most common activation functions are Rectified Linear Unit (ReLU), hyperbolic tangent (tanh) and the sigmoid function.

One popular layer between convolutional layers has been max-pooling. Max-pooling is used to down-sample the size of the learned representation within the hidden layers, highlighting the relevant features and reducing the number of parameters.

Training is conducted via backpropagation, which uses gradient descent to update the weights towards the defined natural outputs by propagating the derivatives of errors. The derivatives are calculated with regards to the error between the predicted and the true output, defined by the loss function. The loss function used plays a critical role here in finding coherency

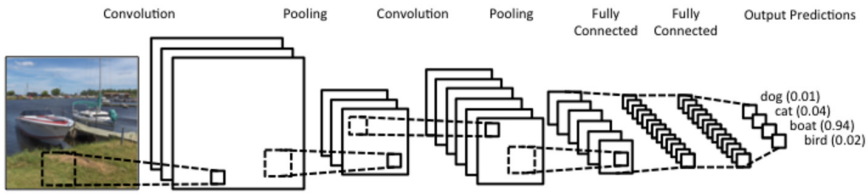


Figure 4.2: A convolutional neural network [68]. Each convolutional layer describes a kernel map from the previous layer, abstracting features onto a higher level of representation relevant for the final classification.

from the data, and is chosen according to each problem. For example, classification tasks may be solved with the softmax loss function, giving a probability distribution of the best fitting classes for each data item. On the other hand, if the loss function (as well as the underlying network) is contrastive, the output is a location in a similarity space, enabling clustering of data points.

Convolutional networks are further augmented with various methodologies, such as DropOut [110], which take care of regularizing the data. DropOut randomly remove connections from the input while training, which creates robustness towards noisy signals and eliminates dependence on singular connections.

Networks are often trained with a number of CNN layers followed by a number of fully connected ones. While the convolutions adhere to the visual features, the fully connected layers collect the final convolutional layer's features and form the final target representation (often the classification task). As per most machine learning methods, neural networks can utilize advanced iterative optimization techniques, such as regularization, batch learning and momentum.

Siamese Network Siamese networks [21] were originally developed for identifying signatures written on a pen-input tablet, but later became popular in other image recognition tasks, such as face recognition [30]. One of their novel qualities is that they project data points onto a metric space, generating a similarity measure between them. This enables inference for data points that have not yet been labelled, which is an important and powerful aspect for document retrieval purposes. Images that are close to each other in the new projection contain relevant details for the retrieval. This is especially true if the projection is changed during the training to

take into account relevant details.

The projection is learned by introducing a pair of images to the network, and depending on whether they are from the same category or not, they are then moved closer or further apart in the projection space. For this, a contrastive loss function was developed:

$$\mathbb{L}(W) = \sum_{i=1}^{P} L(W, (Y, X_1, X_2)^i), \tag{4.6}$$

and

$$(W, (Y, X_1, X_2)^i) = (1 - Y)L_S(D_W^i) + YL_D(D_W^i), \tag{4.7}$$

where $(Y, X_1, X_2)^i$ is the i-th labelled sample pair, L_S and L_D are the partial loss functions for a pair of similar or dissimilar points, respectively, and P is the number of training pairs. $Y = 1$ if the two images are from the same class, and $Y = 0$ if they are from different classes, D_W is the network's predicted distance between X_1 and X_2 given W. This function aims to maximize the intra-class similarity, in the case where X_1 and X_2 belong to the same class, and to minimize the inter-class similarity if they belong to different classes.

The siamese architecture (as seen in Figure 4.3) is able to find a new representation in the feature space that helps to distinguish between different aspects of the image, making it an ideal choice for our image retrieval engine. An important aspect of this architecture is that it generates a distance metric, which may be used to rank or generate relevance scores for all the images in a dataset.

The siamese network consists of two networks, $G_W(X_1)$ and $G_W(X_2)$, which share their weights W. A different image is introduced for both of them, and the both networks output a location in an initially arbitrary metric space. The contrastive loss function is told whether they are similar or dissimilar via the compatibility function E_W, and based on this, the networks are trained whether the images were supposed to be close or apart from each other.

Transfer learning Transfer learning [84] is a framework where reusing existing representations in new contexts allows faster learning of a new domain. It has seen increasingly more research within various machine learning fields, especially deep learning.

Neural networks are naturally good at learning meaningful patterns from data, and this principle allows relearning arbitrary sub-networks for

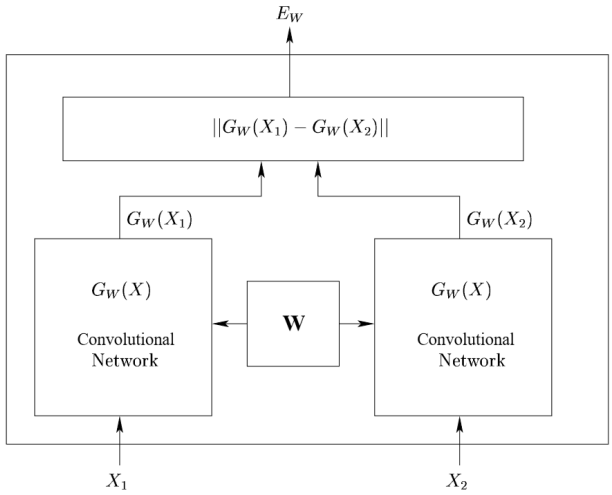


Figure 4.3: The siamese neural network [30]. Two images are fed to the network, and a contrastive loss function is used to distinguish whether the images belong to the same class or not. The weights W are shared by the networks, so that they learn the same representation for images. The purpose is to learn a distance metric that measures the similarity of images.

fine-tuning. Some of the latest research on the topic covers relearning object detectors to detect scenes instead [64], or using randomly downloaded, unlabelled entries to enhance the classification accuracy.

Transfer learning is the essential part of our method's dynamism. It allows the exploitation of good features learned in one domain in order to obtain good features for the target domain. By allowing layers to be reparameterized, it is possible to utilize the existing features, and build upon them a new representation. When retraining the network with what the user finds relevant, the resulting representation is able to project every item in the data set based on the user's preference, enabling online ranking.

Chapter 5

Personalization of Exploratory Search

Creating information retrieval systems that match the information need of an individual is a difficult task, as choosing what to show to the user is highly varied between search sessions. This was noticed to be an especially relevant problem in scientific literature search, where users would easily feel the system is unresponsive to their behaviour and needs [59, 60]. One reason for this problem is that each user, even each session, requires different results for the same query.

Previous research has shown that it is already possible to set the exploration rate effectively for a homogeneous population of users [8]. This was done by modeling the exploration - exploitation trade-off as a direct relationship between the exploration rate parameter, as well as the number of documents that are perceived by the user as interesting or relevant. Although this system works well for the average user, it does not account for the differences between individuals, who require varying levels of support during the search.

It was previously demonstrated that course-grained adaptation to two broadly defined search task types – exploratory and lookup, can be performed based on user behaviour characteristics. For example [10] presents an approach where the user's interaction data (clicks, reading time and scroll depth, etc.) from the first page of results is used to train a classifier to determine whether the user is conducting an exploratory or lookup search. This classification is then used to change the exploration rate of the system to either 0 (for lookup tasks) or 1 (exploratory tasks). This system, however, assumes that a single exploration rate is sufficient to support all users engaged in exploratory search, irrespective of their level of familiarity with the topic. Although this system can distinguish between lookup

and exploratory tasks, it fails to account for the fact that different types of users, or even the same user, may require varying degrees of exploration depending on their background or knowledge related to a specific task or query. Their work assumed that users are from a homogeneous group and only a single parameterisation of the system is necessary for all exploratory search tasks.

In this section I discuss my first and second claims of this dissertation, which tackle these problems in particular. The claims are based on our work on scientific article retrieval system ARES (Publications I and II), and the publication on Gaussian Process Bandits (Publication VI).

- Claim 1: It is possible to personalize the search parameters per session and user, thus accommodating users outside of population-wide tendencies.

- Claim 2: People who know what they do not know give more reliable feedback, and this information can be utilized efficiently when optimizing the search system's parameters.

In this work, we develop an approach to automatically set the level of exploration in a search system based on each user's individual information needs on a per-search basis. With this we show that by using measures that can be collected online during the search, it is possible to better match the information needs of the individual. This supports my first claim.

Our results indicate that taking the user's knowledge level on a given topic into account supports the user's exploration efficiently. This is especially true when the user has at least a basic knowledge on the topic, as this helps them set each piece of information into context. This supports my second claim.

Our approach is based on interval regression using the exploration rate as the response variable, which is, to our knowledge, the first time in the field. As we do not know the optimal exploration rate beforehand, we created censored intervals based on user feedback. In brief, each experiment is performed at a random exploration rate, γ, and we determine if the value should have been set higher (requiring a right-censored interval from γ to $+\infty$) or lower (left-censored from 0 to γ). Our explanatory variables are all based on simple-to-collect metrics based on user search behaviour and can be measured precisely.

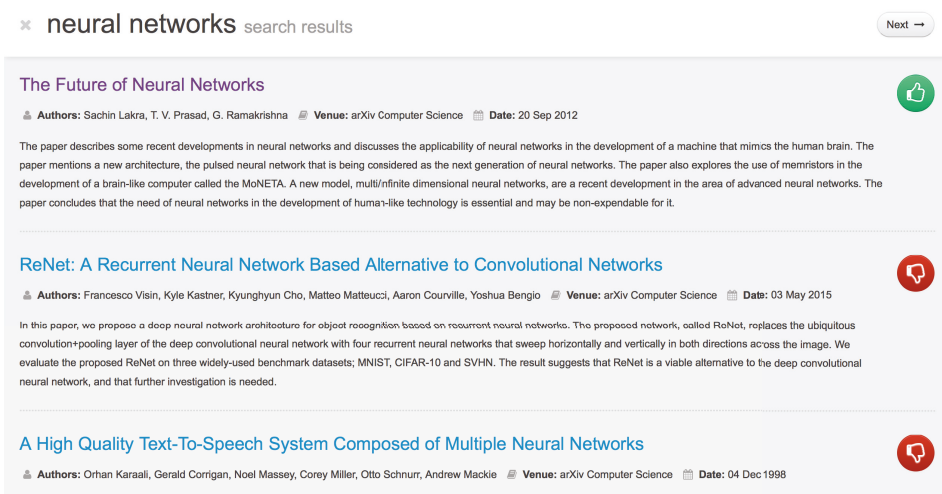


Figure 5.1: The interface of our system. After the query is given at the top, a list of scientific articles are presented for the user, complete with their abstract and meta-data. The user may then indicate relevant items with the like/dislike feature (thumbs up or down). The next button in the top right corner proceeds the search onto the next iteration.

5.1 System Description

The interface of the system is presented in Figure 5.1. After typing in the search query, the user is presented with 20 documents where seven documents are visible without scrolling. We display more documents than in traditional IR systems because in exploratory tasks users examine more results [121]. The initial set of documents is ranked based on the Okapi BM25 algorithm [107]. All documents initially have a "thumbs down" icon in the right margin. Users can indicate which documents interest them by toggling the icon to a "thumbs up". After clicking the "next" button in the top right corner of the page, a new set of documents is displayed based on all the feedback provided by the user. The users can provide feedback to as many documents as they wish. Documents that do not receive an explicit relevance feedback, retaining a "thumbs down", are assumed to receive relevance score of 0. The current version of the system is based on around 1.1 million documents from the arXiv repository.

To help the user explore the document space, we use LinRel [12], which was previously used in exploratory search systems [7, 8, 10, 79]. Suppose we have a matrix D, where each row d_i is a TF-IDF feature vector repre-

sentation of documents presented so far. Let $r = (r_1, r_2, ..., r_t)'$ be a column vector of relevance scores received from the user up to time t. We estimate the expected relevance r_i of a document d_i as $\mathbb{E}[r_i] = d_i \cdot w$, where the vector w is estimated from user feedback. LinRel estimates w by solving $r = D \cdot w$ and estimates the relevance score for each d_i as $r_i = d_i \cdot w$.

In order to deal with the exploration-exploitation trade-off, we present documents not with the highest score r_i, but with the largest upper confidence bound for the relevance score. Thus, if σ_i is an upper bound on standard deviation of relevance estimate r_i, the upper confidence bound of document d_i is calculated as $r_i + \gamma\sigma_i$, where $\gamma \geq 0$ is a constant used to adjust the confidence level of the upper confidence bound. In each iteration, LinRel calculates $s_i = d_i \cdot (D^\top \cdot D + \lambda I)^{-1} D^\top$, where λ is the regularization parameter which is set to 1 as each of the feature vectors sums up to 1 (following [12]) and the documents that maximize $s_i \cdot r + \frac{\gamma}{2}\|s_i\|$ are selected for presentation. The first term, $s_i \cdot r$, effectively ranks all the documents based on their similarity to the documents the user has selected so far and thus it narrows the area of the search space (exploitation). The second term, $\frac{\gamma}{2}\|s_i\|$, ensures that the user is presented with a more diverse set of results. The exploration rate is controlled by the γ parameter. The higher the value of γ, the more diverse, or exploratory, the results are.

5.2 User Studies

Two separate user studies were performed during our experiments, an initial study to assess the parameterization of a regression model, and another study to verify its applicability in fine-grain personalisation of exploration.

The goals of the user studies were to investigate how different exploration rates affect: 1. the number of clicks and the reading time, 2. subjective satisfaction, and 3. overall correlation between the user's knowledge of a given research topic, the exploration rate and the user search behaviour. For the initial study we recruited 20 MSc students who were in the process of writing their final dissertations. For the verification study we recruited 25 students working on their MSc dissertation or in the first year of their PhD, recruited separately from the previous study.

Prior to both studies we provided the subjects with a background questionnaire to assess their experience with literature search and their self-reported knowledge of a set of search topics on a 5-point Likert scale, where 1 = "no knowledge" and 5 = "very familiar". All participants reported experience in scientific literature search.

Design An expert researcher from the machine learning domain selected six topics that have sufficient data in the arXiv dataset: clustering, privacy, image processing, natural language processing, security, and neural networks. The machine learning domain was chosen because it was easier to find both domain experts for creating tasks, and a homogeneous group of participants who had taken at least an introductory machine learning course. Every participant performed two search tasks. For each task the exploration rate was randomly sampled without replacement from the set of values between 0.0 and 1.0 (inclusive) at intervals of 0.1. Thus, overall there were 11 possible exploration rates. As users engaged in exploratory search aim to acquire new knowledge [122], we specifically assigned to each participant two search topics that they reported to have some knowledge of but were not overly familiar with (scores 2, 3 and 4 on the Likert scale).

We described the tasks using a template that places participants in a scientific essay writing scenario, suitable for exploratory search tasks [122]. To preserve consistency among the tasks, all task descriptions followed the same template:

> *"Imagine that you are searching scientific literature to write an essay about topic X. We provide you with a search engine. Follow your natural literature search process and search for articles that help you to learn about this topic. You must prepare an abstract for your essay at the end."*

In order to ensure that all the studies for a given search topic were initiated from the same starting point, we provided the initial search query. Additionally, in order to ensure that all the participants went through an equal number of iterations, we limited each search session to five iterations. Based on our pilot study, five iterations provided the users with enough documents to perform the task, but at the same time kept the search session short enough to keep the users focused. There was no time limit set with regards to the duration of each search session or each iteration.

Measures and Procedure The initial studies were conducted in a controlled laboratory room, with a desktop computer with a 27-inch display. The verification studies used the same environment, but were conducted on a laptop computer with a 19-inch display. We automatically terminated the search session after five iterations. Prior to the study we informed the participants about the number of iterations required per task, introduced the search system and gave them one training task. We showed the participants how to select relevant articles and proceed to the next iteration. We

explained that by providing feedback they allow the system to learn about their interests and retrieve more suitable results in subsequent iterations. The participants were not aware of the different exploration rates per task. Participants were allowed to take notes using their comfortable technique – a word processing application or pen and paper. At the end of every task, we provided a web form to write the abstract of their essay.

We logged all the interactions with the search system: number of clicks, scroll data, details of all the displayed documents and selected documents. A domain expert performed a blind review of the abstracts and rated them from 0 to 5 (0 = failure and 5 = excellent). The aim of the assessment was to ensure that the participants were able to complete the task and find relevant documents. In order to assess user satisfaction, after each task the participants completed a short ResQue questionnaire [88], including a question related to their subjective satisfaction of the diversity of presented articles with reference to the initial search query:

> *"The search results recommended by the system contained documents closely related to the initial search query as well as articles related to other topics with varying degrees of relevance to the initial query. Based on the search session that you have just completed, would you prefer the search results to contain:*
> *a) more articles closely related to the initial search query*
> *b) more articles related to other topics with varying degrees of relevance to the initial search query"*

The responses provided to this question were later used in building our ARES interval regression model (the details will be presented in the next section). We included this question in our verification experiments to refit the model and compare it with the ARES model obtained in [80] to assess whether it can be replicated. Each study lasted approximately one hour. We compensated every participant with a movie ticket. At the end of the verification study we conducted a short interview with each participant to learn more about their experiences with each system.

5.3 Results for the Model Fitting

As our analysis is based on interval regression, we needed to combine the subjective satisfaction feedback with the randomly assigned exploration rate to determine an appropriate interval for fitting the model. If a user was randomly assigned the exploration rate, γ, and they stated in the post-experimental survey that they would have liked more articles closely related

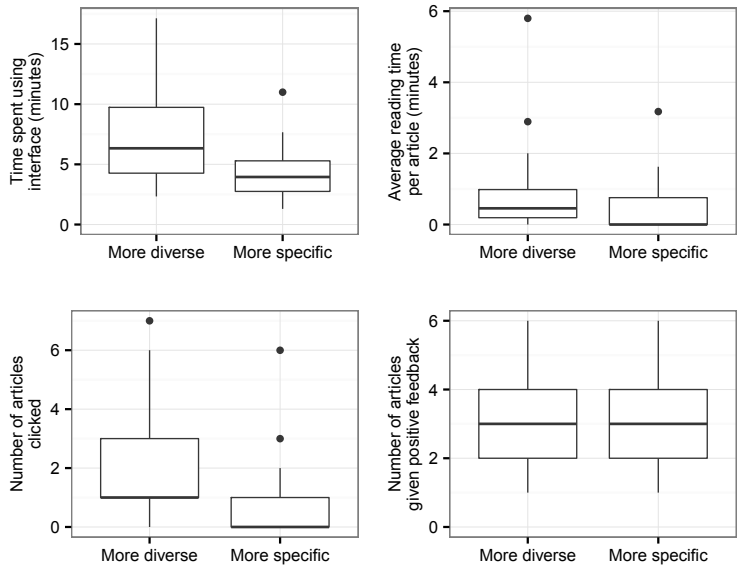


Figure 5.2: Boxplots showing the distribution of variables collected from implicit feedback. All variables except the number of articles given positive feedback vary with the experimental outcome.

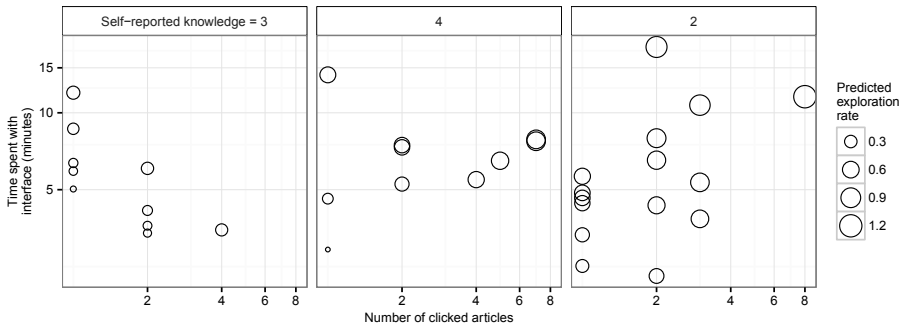


Figure 5.3: Graphical representation of regression model. Predicted exploration rate increases as a function of number of clicked documents, time spent with the interface and the level of self-reported knowledge in the order of the graphs (levels 3, 4 and 2). Area of circles is proportional to predicted exploration rate.

to the initial search query (from now on referred to as *more specific*), then this was encoded as a left-censored interval, $[0, \gamma]$. If, however, they stated they would have liked to see articles related to more diverse topics (*more diverse*), a right-censored interval, $[\gamma, +\infty]$, was used. In addition to the censored intervals we have to provide the distribution for the predicted variables, for which in this case we used a Gaussian distribution. Interval regression was performed using the Survival R package (ver. 2.38) [114].

A total of 40 experiments were performed, of which five were excluded from further analysis for the following reasons: in one experiment the post-experiment survey was incomplete. Another user had both experiments excluded as they appeared to have misunderstood the task. Further two experiments were excluded after being identified as outliers with principal component analysis.

Figure 5.2 shows the distributions of implicit variables collected during the first iteration of each experiment. Time spent with the interface excluding reading time (top-left), average reading time per article (top-right) and number of documents clicked (bottom-left) are higher on average when users want to see more diverse articles.

The number of articles given positive feedback (bottom-right) appears to be independent of user satisfaction. Self-reported knowledge level 2 had 15 data points; levels 3 and 4 had 10 data points each.

We performed model selection to find the simplest model that would enable us to predict an appropriate exploration rate. First, we fit a full model using all variables (average reading time per article, time spent with

the interface, number of clicked articles, number of articles with positive feedback, self-reported knowledge and the order of experiments). We encoded self-reported knowledge and the order of experiments as categorical variables (self-reported knowledge appears like it would be ordinal, but this is not the case). All other variables are strictly positive and were therefore log-transformed. Co-variates were dropped from the model if a χ^2 goodness-of-fit test comparing the current and nested models (the current model minus the co-variate being investigated) was not statistically significant. After dropping a co-variate, the nested model became the new current model.

After applying this model selection procedure, the model had only three significant predictor variables (p-values determined by χ^2 test): time spent with the interface ($p = 0.017$), number of clicked articles ($p = 0.025$) and self-reported knowledge ($p = 0.0024$). Contrasts between levels of self-reported knowledge was tested using general linear hypotheses tests. While the difference between 2-3 was highly significant ($p = 0.0004$), the differences between levels 2-4 ($p = 0.06$) and 3-4 ($p = 0.46$) was not significant. This might be due to the fact that some of the participants were overconfident when reporting their level of knowledge.

The final regression model to predict the exploration rate, γ, is:

$$\gamma = 0.29\ln(x_1) + 0.22\ln(x_2) - 0.44x_3 - 0.29x_4 + 0.06, \tag{5.1}$$

where x_1 = time spent with the interface, excluding document reading time, in minutes, x_2 = number of documents clicked, x_3 and x_4 are dummy variables for the self-reported knowledge levels 3 and 4, respectively.

Figure 5.3 shows how the predicted exploration rate changes as a function of each co-variate; increasing at a similar rate proportional to the log of both the number of documents clicked (x-axis) and time spent with the interface (y-axis). Self-reported knowledge is ordered by the coefficient magnitude (note: level 2 is the base-line in equation 5.1). We note that while our experiments only used exploration rates in the range [0,1], the model predicts exploration rates in the range $[0,+\infty]$.

Figure 5.4 shows whether our predictions are logically consistent with user feedback. Users that wanted documents more specific to their search query (blue dots) should have been predicted lower exploration rates than their experiments and therefore be under the $y = x$ dashed line. Symmetrically, users wanting more diverse documents (red dots) should be above $y = x$. The graph shows four blue dots and three red dots on the wrong side of the line, making 80% of predictions consistent with feedback. We note, however, that all of these inconsistent data points are close to the line $y = x$.

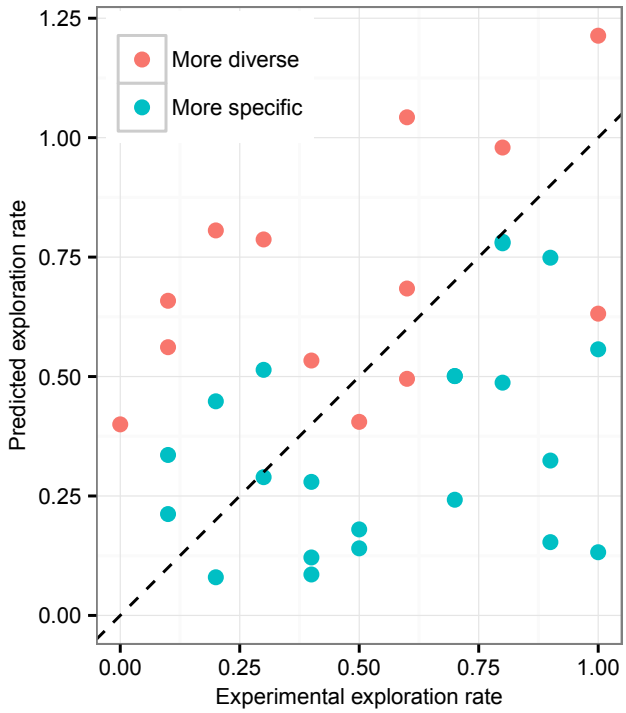


Figure 5.4: Predictions that are logically consistent with user feedback are red dots (users that wanted more diverse documents) above the line $y = x$ and blue dots (users that wanted to see more specific documents to the search query) below it. 80% of predictions were consistent with user feedback.

5.4 Incorporating the Regression Model into an IR System

The initial study's regression model was incorporated into a system which we call Adaptive Regression-based Exploratory Search (ARES). In this version of the system, the user first specifies their level of knowledge related to the subject area of their current search query. After examining the first page of results and clicking on the "next" button, the user interaction data and pre-specified knowledge level of the user are passed on to the regression model, which calculates the optimal exploration rate for the current search session (Figure 5.5).

The study involved 25 participants (9 female and 16 male), performing two experiments each. Six participants were first-year PhD students and

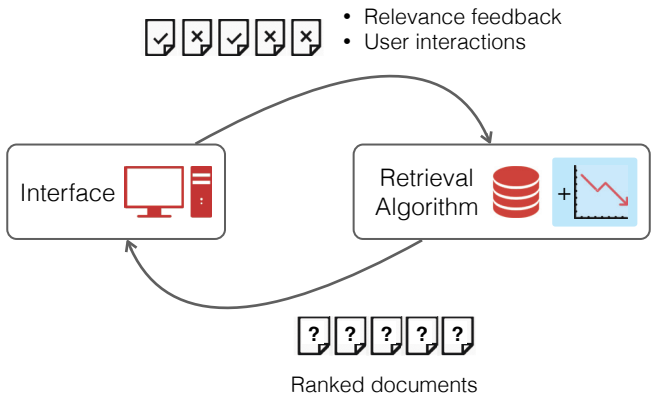


Figure 5.5: An overview of the ARES retrieval system. The user types in an initial query and specifies their level of knowledge of the search topic. The user is presented with a list of documents, clicks on the document(s) and proceeds to the next iteration by clicking the "next" button. The clicks, the self-reported knowledge level and the user interaction data, i.e. reading time or scroll depth, are passed on to the regression model, which calculates an appropriate level of exploration for the current search session. This exploration rate is unchanged for the remainder of the search session.

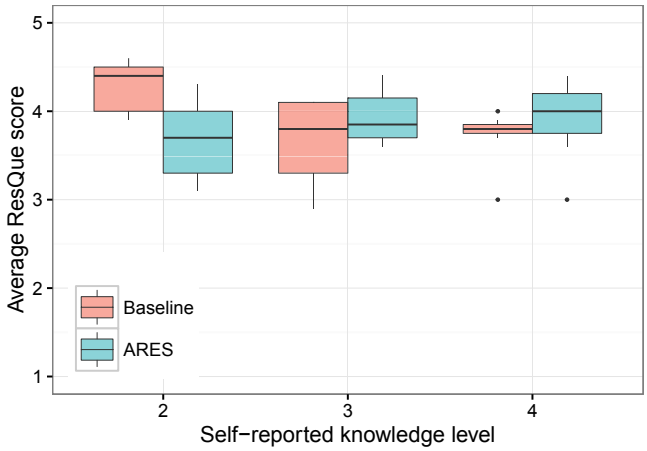


Figure 5.6: Mean ResQue score stratified by self-reported knowledge level and system. There are two regimes: users with level knowledge 2 prefer the baseline, while users with knowledge levels 3 and 4 prefer ARES.

19 were Masters students. The final number of experiments with ARES included in the analysis was 24, and with the baseline system 25. Of the 49 experiments, 14 were with a self-reported knowledge level of 2 (9 with ARES and 5 with the baseline), 21 with 3 (8 with ARES and 13 with the baseline), and 14 with 4 (7 with ARES and 7 with the baseline). Throughout this section, we used regression analysis due to the imbalanced nature of the data and to allow us to incorporate control variables, where appropriate.

5.4.1 User Perception

To understand whether there is a difference in user perception between the systems, we analyzed the user ratings from the ResQue questionnaire (Table 5.1). We also analyzed the qualitative feedback given by each participant during the post-study interviews and the expert assessment of the essay abstract written by each participant.

Users with self-reported knowledge levels 3 and 4 tended to prefer ARES, whereas those with knowledge level 2 appeared to prefer the baseline. The average ResQue score was 38.21 ($SD = 4.11$) for ARES and 38.04 for the baseline ($SD = 4.32$). There was no significant difference between systems ($t(40) = -0.028$, $p = 0.83$).

Closer inspection of the distribution of ResQue scores shows that users preferred different systems depending on their knowledge level (Figure 5.6). Rerunning our previous analysis for only knowledge levels 3 and 4 showed

there was a significant difference between ResQue scores given for each system ($t(25) = 2.488$, $p = 0.02$), with users preferring ARES ($M = 39.2$, $SD = 3.75$) to the baseline ($M = 36.85$, $SD = 3.75$).

Analysing knowledge level 2 in isolation shows the average ResQue score for ARES is 36.56 ($SD = 4.36$) and the baseline is 42.8 ($SD = 3.11$). The difference is significant ($t(5) = -2.646$, $p = 0.046$), however, the sample size is low (14 observations: 9 ARES + 5 baseline) suggesting it is too underpowered to draw conclusions.

We further analyzed individual ResQue questions for knowledge levels 3 and 4 using ordinal logistic regression (ordinal R package, version 2015.6.28). This is similar to multinomial logistic regression with the exception that response variables are ordered, as in Likert scale responses. Again, we controlled for the ordering of experiments, the search query and self-reported knowledge level. We report the logit values of the system coefficient together with P-values from one-tailed z-tests (Table 5.1).

From the results, we can see that ARES suggested papers that were more appropriate for the users' level of knowledge than the baseline (Q5, $p = 0.035$). Documents shown by ARES were less similar to each other (Q10, $p = 0.044$) and, therefore, more diverse (Q1, $p = 0.059$) than the baseline. ARES provided users with more novel papers than the baseline (Q8, $p = 0.019$), which users felt were good suggestions (Q4, $p = 0.072$). While the baseline always uses an exploration rate of 1, in these particular experiments ARES used lower exploration rates, especially for knowledge levels 3 and 4. It is, therefore, interesting that users perceive the system as giving more diverse results including more novel articles. This is suggestive that participants define documents as "diverse" or "novel" in reference to the initial query, but not more random documents from elsewhere in the search space.

Qualitative feedback Overall 15 out of 25 participants preferred searches performed with the ARES system. Some of the comments about the system included: "the papers were informative" [P22], "the system seemed to find relevant things" [P1], "the system went into different directions really well" [P4]. Participants who expressed a preference for ARES complained about the baseline system because "it went off the track" [P25] or "it converged on too specific details too soon" [P11]. There was a clear correlation between the level of knowledge and system preference. In general, participants with knowledge level 2 did not rate the ARES system highly. On the other hand, the majority of participants with knowledge level of 3 and 4 who conducted the search with ARES preferred it over the baseline (13 out

#	*Question*	*ARES*	*Baseline*	*Logit*	*P-value*	
1	The papers suggested by the system were diverse	4.33	3.85	1.16	0.08	·
2	The suggested papers helped me to learn more about the topic	4.07	4.0	0.39	0.313	
3	The topic was covered comprehensively during the search session	3.2	3.2	0.60	0.221	
4	The system gave me good suggestions	3.8	3.55	1.12	0.08	·
5	The papers suggested by the system were appropriate for my level of knowledge	3.93	3.65	1.63	0.033	∗
6	Some of the suggested papers are familiar to me	2.07	2.3	0.36	0.33	
7	The papers suggested by the system were interesting	4.33	4.15	0.96	0.14	
8	The papers suggested by the system were novel	3.93	3.6	2.09	0.013	∗
9	The system helped me to discover new papers	4.4	4.45	-0.91	0.22	
10	The papers suggested by the system are similar to each other	2.73	3.3	-1.44	0.035	∗

Table 5.1: Each question from the *Quality of Recommended Items* section of ResQue was analysed independently using ordinal logistic regression. The final column states whether P-values were significant at 0.05 (∗) or 0.1 (·). Papers suggested by ARES were appropriate for participants' level of knowledge (Q5), were novel (Q8) and not similar to each other (Q10, logit sign is negative).

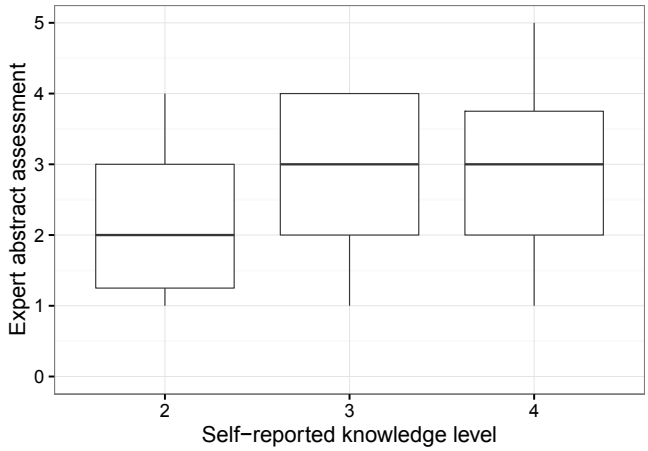


Figure 5.7: Boxplots showing the distribution of expert abstract assessments for each self-reported knowledge level. Expert assessments correlated well with self-reported knowledge level.

of 15 participants). This is concordant with our analysis of the responses given in the ResQue questionnaire by these participants. The preference for ARES at higher knowledge levels was unaffected by the knowledge level of their other search query conducted with the baseline.

Expert assessment According to a blind assessment by a third expert, there was no significant difference in the quality or length of abstracts produced using either system, nor in abstract assessment or word count for either system when compared using only knowledge levels 3 and 4. Figure 5.7 shows the expert assessments correlate well with the participants' self-reported knowledge levels. The mean abstract assessment score was 2.8 ($SD = 0.94$) for participants using ARES and 2.75 ($SD = 1.25$) for the baseline.

5.4.2 User Behaviour

During the user study, we logged information related to user behaviour, including the total time spent using the interface, the number of documents clicked and given positive feedback, and the time spent reading each document.

The analysis of user perception data has shown that users with higher knowledge levels preferred ARES over the baseline. We analysed user behaviour data for these participants (knowledge levels 3 and 4) with regres-

sion models for count data. Our goal was to understand whether the user experience results are reflected in user behaviour metrics.

Interactions The number of documents that were clicked or received positive feedback was independent of which system was being used. The mean number of documents clicked for reading was 4.93 ($SD = 2.60$) with ARES and 6.0 ($SD = 4.21$) with the baseline. While there appears to be a difference between the systems, it was not statistically significant ($LRT = 3.524$, $p = 0.061$). The mean number of documents given positive feedback was 12.47 ($SD = 7.30$) with ARES and 15.8 ($SD = 12.77$) with the baseline. The difference between the systems was not statistically significant ($LRT = 0.547$, $p = 0.46$).

Interface and reading time While the time spent with the interface was independent of the search system, users on average spent much longer reading each document with ARES compared to the baseline. The mean time spent with the interface was 823.85 seconds ($SD = 511.91$) with ARES and 1129.61 seconds ($SD = 846.12$) with the baseline. There was no significant difference between systems ($LRT = 2.47$, $p = 0.116$). The mean reading time per document was 589.14 seconds ($SD = 291.41$) with ARES and 625.23 seconds ($SD = 495.04$) with the baseline. While the raw mean reading times are very similar, when control variables are taken into account, participants spent on average $\sim$1.7 times longer reading each document with ARES ($LRT = 6.93$, $p = 0.008$). It is unlikely that the documents returned by ARES were inherently better, but the phenomenon was probably due to users quickly inspecting some of the more diverse documents returned by the baseline.

5.4.3 System Behaviour

Despite clear differences in user experience between ARES and the baseline at higher knowledge levels, there does not appear to be a difference between the diversity of documents presented to users in either system. We measured document diversity using the mean pairwise cosine distance between TF-IDF transformed bag-of-words representations of document abstracts. Figure 5.8 shows how document diversity decreases over iterations as participants give positive feedback to documents to narrow down their search. We compared the diversity of articles presented by each system in the final iteration of the experiment using linear regression, controlling for the ordering of experiments, the search query and the self-reported knowledge level. The mean pairwise distance between documents returned by ARES

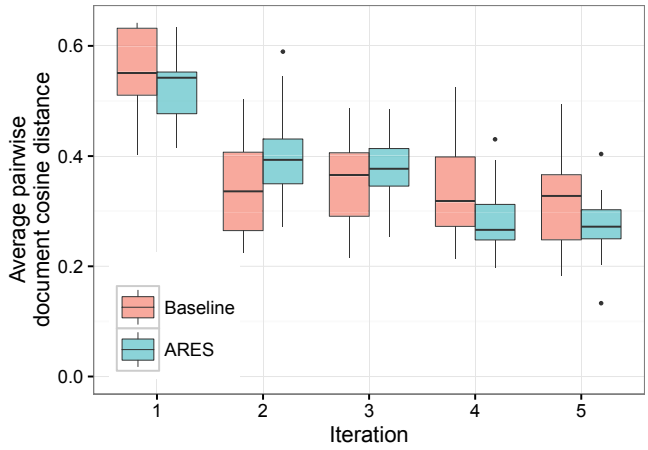


Figure 5.8: Document diversity measured using mean pairwise cosine distance between documents over iterations for experiments performed at knowledge levels 3 and 4. As participants progress through iterations giving positive feedback to documents, the diversity of search results decreases.

was 0.63 ($SD = 0.11$) and by the baseline was 0.62 ($SD = 0.12$). There was no significant difference between systems ($p = 0.561$).

5.4.4 Refitting ARES

The ARES regression model is robust and can be replicated using data from this study. The experiments from [80] used randomised exploration rates and user feedback to derive the ARES regression model. To restate, the model is as follows:

$$\gamma = 0.29 \ln(x_1) + 0.22 \ln(x_2) - 0.44x_3 - 0.29x_4 + 0.06, \tag{5.2}$$

where x_1 = time spent with the interface, excluding document reading time, in minutes, x_2 = number of documents clicked, x_3 and x_4 are dummy variables for the self-reported knowledge levels 3 and 4, respectively.

During the post-experimental questionnaire in this user study, the participants were again asked whether they felt the search results were too diverse or too specific with respect to the initial search query. This time we encoded intervals not using random exploration rates, but the exploration rates predicted by ARES. The interval regression model was fit using the Survival R package (ver. 2.38). The refit regression model is:

$$\gamma = 0.31 \ln(x_1) + 0.22 \ln(x_2) - 0.43x_3 - 0.29x_4 + 0.02. \tag{5.3}$$

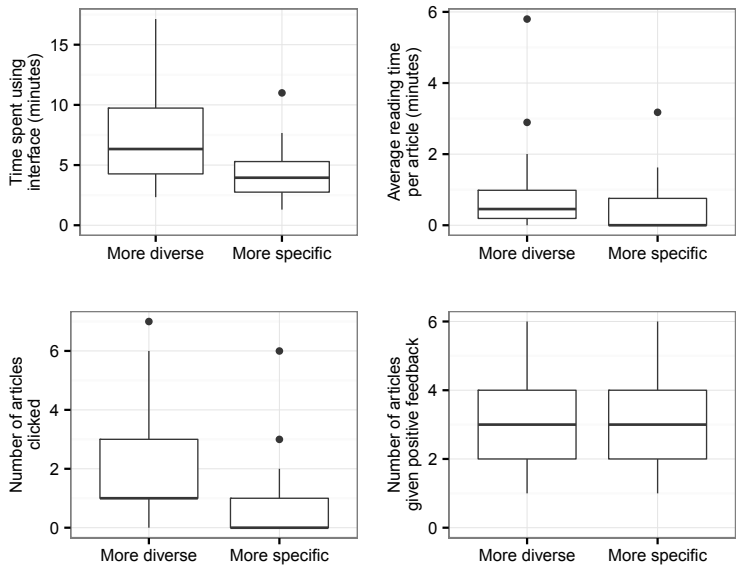


Figure 5.9: User behaviour characteristics, stratified by knowledge level. While the number of clicked articles appears to be correlated with knowledge level, most metrics that can be collected during a search session only distinguish between knowledge level 4 and all other levels.

All coefficients in this slightly refined model are within the standard errors of the ARES model that was derived in [80]. Conversely, all coefficients of the ARES model [80] are within the standard errors of the current refined model, with the exception of the intercept (the intercept, however, is not significant in either model).

In opposition to the results from the ResQue questionnaire, we note that users with knowledge level 2 are evenly split on whether the search results should be more diverse or more closely related to the search query (five out of nine wanted results to be less diverse). While we were able to replicate the ARES model, more nuanced subjective workload assessments have shown that feedback from less knowledgeable users is unreliable.

5.5 Discussion

The aim of this study was to validate the effectiveness of incorporating a regression model into an IR system in order to provide fine-grained support to users preforming exploratory search tasks in an interactive setting. Our

implementation, ARES, requires users to provide a search query along with an estimate of their knowledge level for the topic of that query. Additional user behaviour metrics required by the model (number of clicked documents and time spent with the interface, excluding reading time), were collected automatically by the system. Using these variables, a regression model infers the most appropriate exploration rate to parameterize a reinforcement learning algorithm that is used to build a user model interactively.

Our findings highlight that modeling subjective user expectations of system behaviour succeeds for users with higher knowledge levels, and when it fails for users with the lowest knowledge level. Users with knowledge levels 3 and 4 reported significant differences in novelty among the search results and found that papers suggested by the system were appropriate for their level of knowledge. These two findings in particular were the most important to us because we are performing exploratory search (and therefore want to encourage serendipitous discovery of novel papers) and because the model explicitly includes the knowledge level (and was statistically significant enough to warrant inclusion). At the lowest knowledge level 2, however, users appeared to prefer the baseline. While analysing this subset of users in isolation is statistically underpowered, their preference for the baseline is backed up by the qualitative analysis of post-experiment interviews.

Exploratory search support ARES provides fine-grained support for exploratory search by mapping the information needs of the user to an appropriate exploration rate using mostly implicit feedback. ARES was successful in supporting exploratory search, improving user experience for a wide range of users (Section 5.4.1). The model is not perfect, but those it failed to support (knowledge level 2) could be accommodated with the baseline system, which is a simple distinction to implement.

Another important feature was the low training cost of the model, as fitting it required only 35 observations [80]. This, combined with the transparent nature of linear models, makes it an easy choice for any practical search environment.

Finally, the covariates used by ARES (clicked documents, interface time and the knowledge level) are highly robust. We found no significant differences in the number of clicked documents or time spent with the interface, despite the underlying system being different (Section 5.4.2). Furthermore, knowledge level, which is self-reported by the participant, is concordant with the expert abstract assessments (Figure 5.7).

Epistemic modeling failures Several participants commented they felt their ratings of each system were related to their knowledge of the query topic. For example, one participant stated that "the level of knowledge affected the rating here very highly" [P22]. From one perspective, this may not be important because ARES and the baseline had identical interfaces. These users could therefore have been attributing their perceived differences to things they were actually aware of, i.e. the search query and their own knowledge of the topic. However, given that ARES failed to support users with the lowest knowledge level, this could be related to failures in the modeling process.

There are two types of uncertainty: aleatoric and epistemic. Aleatoric uncertainty is the kind of uncertainty that we can model statistically, i.e. the unknown factors that result in random variation when experiments are repeated. Epistemic uncertainty, however, is the "unknown unknowns" and represents the variability present because we have not asked the right questions, lack sufficient knowledge of the problem or information is purposefully hidden. ARES attempts to overcome aleatoric uncertainty: we are modeling users' expectations about how the system should feel and predict the exploration rate from behaviours that are associated, but not contingent on, how exploratory users want the system to be. However, we have introduced epistemic uncertainty by asking lower knowledge users whether the system should present documents that are more or less diverse. These users, by their own admission, lack knowledge on the search topic, making their feedback potentially unreliable.

Confusingly, when we refit the model using new data from this study, the resulting model was almost identical to the ARES model (Section 5.4.4). Despite knowledge level 2 users preferring the baseline – which always gave a higher exploration rate than ARES – approximately half of these participants (five out of nine) stated that they wanted the search results to be more closely related to the initial search query. We would therefore expect this model to be correct, but the more in-depth subjective workload assessments reveal otherwise. Despite good model fit and all covariates in the model being highly significant, the participant's own lack of knowledge on the topic – the epistemic uncertainty that we do not model – prevents us from inferring appropriate exploration rates.

Inferring knowledge level ARES only uses simple, easy to collect interaction data with the exception of knowledge level, which we need to explicitly ask the user to provide. Ideally, the search engine would only use implicit feedback to parameterize itself for different users. Figure 5.9

shows boxplots of six metrics that are easy to collect, stratified by knowledge level. Of all the variables that can be collected during a search session, only the number of clicked documents correlates with the knowledge level (Figure 5.9, top row, centre graph). There is no correlation with the number of documents given positive feedback (Figure 5.9, bottom row, centre graph) and a partial correlation with interface time and average reading time, with knowledge level 4 taking longer than others (Figure 5.9, left column, top and bottom graphs, respectively). Metrics that can only be evaluated after the search session (abstract word count) or offline (expert abstract assessment) correlated with the knowledge level. These could be easily used by a search system with persistent user accounts, where information about interests and exam grades could be stored.

The positive correlation between the number of clicked documents and knowledge level (Figure 5.9, top row, center graph) is suggestive that users with greater knowledge might be more curious. That is, their increased knowledge of the field, e.g. specialized terminology, may enable higher knowledge users to identify documents worth considering for relevance. Highlighting words that are associated with terms enriched in the set of documents already given positive feedback, may increase users' curiosity of seemingly unrelated documents. Alternately, all users may simply be good at filtering out documents not appropriate for their level of knowledge, with lower knowledge users filtering out proportionally more.

5.6 Conclusions

Our findings demonstrate that it is possible to build a system that provides fine-grained personalised support for users conducting exploratory search in the scientific domain using only simple and easy to collect user behaviour characteristics. The derived regression model is simple to implement and could so be easily deployed in large-scale commercial search engines. Additionally, our approach does not require any special equipment, such as eye trackers, that would need to be deployed on the client-side. The main drawback of our approach is that users have to specify their knowledge level, which might make the resultant system somewhat cumbersome for some users. However, we suggest this obstacle could be overcome with persistent user accounts.

We also show that different modeling approaches might be required for users with a higher knowledge level of a given subject versus less knowledgeable users: the more knowledgeable the user is, the more reliable their feedback is, which, in turn, leads to a higher level of personalisation with

very little feedback required. Less knowledgeable users provide less reliable feedback, which prevents the system from fully catering to their information needs. Our suggestion for personalising exploratory search for this type of users is to base it on population-level statistics. However, further studies are needed to investigate other approaches for modeling users with a lower level of knowledge.

Chapter 6

Exploratory Image Retrieval

Large-scale image retrieval has been dominated by methods that employ textual annotations for the identification of content. These techniques are dependent on high quality meta-data, and work best in scenarios where the target is exactly described by them. Unfortunately the growth of data has far surpassed our ability to maintain tags on every new image, and even when this has been done, the search can only find them within the context of the tags. It might be easy for a user to define their query if they are looking for an image of a cat, but how do they specify that the cat should be playing with a ball of yarn? A solution to this is content-based image retrieval (CBIR) [33], especially in combination with relevance feedback [129], that actively involves the user into the search loop and utilizes his knowledge in the iterative search process [8, 9, 10, 43, 60].

The interactive nature of CBIR poses additional difficulties for the search engine, from the requirement for fast responsiveness to the lack of directing feedback. The response time itself may not exceed 4 seconds, or else it interferes with the user's experience [23]. Similarly, the systems needs to learn what the user is interested in from a very small amount of feedback – the users tend to indicate only a few images as relevant per iteration [41, 44, 50, 63].

Earlier CBIR systems utilized a variety of feature descriptors that were readily extracted from pictures, from texture detectors to local feature representations, such as the bag-of-words models [123] in conjunction with local feature descriptors (e.g. SIFT [72]). The downside of these methods is their distance to human understanding and perception, forming a semantic gap between them and what the user wants. The latest advances in deep neural networks has made it possible to further close this gap [46]. The first application of deep learning was for image classification, where they gained fame in the ImageNet competition [100]. The network's abil-

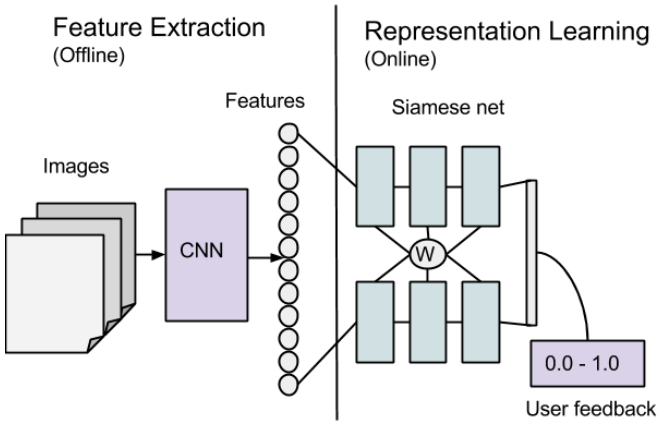


Figure 6.1: The siamese architecture with the image feature preprocessing step. The online component accepts two feature vectors, one per image, and user feedback as the label.

ity to form high-level representations has proven an asset in various image recognition tasks, even in tasks for which they were not trained, such as scenes or CBIR [45, 92, 118].

Learning deep hierarchies for fast image retrieval was considered before by using autoencoders [65] or creating hash codes based on deep semantic ranking [128]. While both methods are fast, neither is flexible enough to learn the image target based on the small amount of relevance feedback obtained from the user. Similarity measures were studied by Wan et al. [118], where deep learning was applied to learn a similarity measure between images in a CBIR setting. Unfortunately, no consideration was given to the time requirements of the learning task, which is an important aspect of interactive retrieval systems. The reported training procedure uses entire datasets and the training itself can take days. Similarity learning can also be used to find new metrics between faces by maximizing the inter-class difference, while minimizing the inner-class difference [30, 51], however, the method was not tested with a broader set of images and features. Two recent studies [39, 119] took into consideration the training time requirements. However, their system setting relies on using thousands of images for training, which is too large for a user to tag over the span of a single search session.

The third and fourth claims of this dissertation are based on the three publications on our exploratory CBIR system [50, 89, 90]:

- Claim 3: It is possible to use very generic features, yet still catch a

very specific target with contemporary transfer learning.

- Claim 4: Users do not need to know exactly what they want at the beginning of the search session, but with appropriate support from the search engine, they can direct their search towards a desired direction.

Our goal for the system was to show how to learn a definite representation of the user's target with only a few training examples, as well as to reach this solution in less than 4 seconds. Furthermore, we will show how this can be done with a set of generic and accessible deep learning features, over which we learn the representation that the current user needs. The system consists of an interface specialized for quick content-based selection of images, a similarity-based backend for learning the representation of relevance, as well as algorithms to optimize the exploration of the learned metric space more efficiently. Together, these systems make the proposed system interactive enough, keeping the user engaged in the search loop, answering the third claim.

The ultimate purpose of our system is to assist the user in finding images that cannot be easily described using tags, such as an image of "beautiful sky". By testing the system over several datasets representing targets from abstract and generic to specific targets, we test the speed and versatility of transfer learning. Together with the exploratory system presented in Publication IV, the results indicate a user should be able to merely click through relevant images, and soon arrive at their point of interest. As our claim four posits, this requires little conscious effort from the user, helping them explore unknown territories with ease and celerity.

6.1 System Overview

Our system needs to learn what the target of the search is through relevance feedback obtained on the small number of images displayed at each iteration. As the user's final search target may be an image containing any combination of features, our method utilizes a distance metric between images to learn what features or combination of features might be of interest to the user. This allows the system to learn a representation based on the user feedback on the presented images, and show the user more relevant images as the search progresses. The system differs from a classifier in that it does not predict which particular classes the user is interested in, but instead tries to learn what features or combination of features might be of interest to the user.

The system described in Publications III, IV and V adheres to a search procedure that can be briefly summarised as follows. The search starts with a random selection of images presented to the user. At each search iteration, the user is presented with k images and indicates which images are relevant to their search by clicking on them. The remaining images in the set of k images that did not receive any user feedback are treated as irrelevant. Based on this feedback, all the images in the dataset are re-ranked using a distance measure and the top k images are presented to the user at the next iteration. Images that were presented to the user so far are excluded from future iterations. If no images are selected as relevant by the user, we assume that all the presented images are irrelevant, and images that are maximally distant from the presented ones are shown to the user at the next iteration. The search continues until the user is satisfied with the presented images.

Additional to this, Publication V presents an exploratory system which works as follows. Based on the feedback, a *central target* is located, indicating the image with the estimated highest potential relevance for the user, and is suggested as the first result for the next search iteration. As this estimation is done over the now changed relevance space, all images closest to the central target are also the most relevant, relevance lowering as distance grows. The neighbourhood around this image is then clustered, and the centroids of these clusters are added to the results. As each image comes from a distinct group, they are *iconic* representations of the neighbourhood of images, ensuring a fair exploration into the region.

Below, we describe the feature extraction process and the architecture of the system in more details.

6.1.1 Feature Extraction

In order to obtain a good base representation, we use CNNs to extract image features. CNNs generate high quality classification results end-to-end from low, pixel-level data to image labels by utilizing deep non-linear architectures. The higher level features from these networks have been successfully used in tasks involving classification of images that were not in the initial training set. This can be achieved by fine-tuning the parameters of the network based on information related to the user's interests. For our tests, we use features extracted with OverFeat [103] and relearn only the last few fully connected layers for the target representation. OverFeat is a publicly available CNN trained on the ILSVRC13 dataset [100], on which it achieved an error rate of 14.2%. ILVSRC13 contains 1000 object classes from a total of 1.2 million images. OverFeat has been shown to be

Layer	Input size	Output size	
$FC1$	4096	100	Fully connected layer
$ReLU$			Rectified Linear Unit
$FC2$	100	20	Fully connected layer
$ReLU$			Rectified Linear Unit
$Feat$	20	6	Final feature layer
CLF			Contrastive loss function

Table 6.1: Composition of the neural architecture used in our system.

successful at various image recognition tasks from fine-grained classification to generic visual instance recognition tasks [92]. The chosen features were a set of hidden nodes as the fully connected graph begins from layer 7 (19 within the architecture), corresponding to 4096-dimensional image representations. The images were shrunk and then cropped from all sides to produce images of equal size of 231×231 pixels. Table 6.1 shows the composition of the neural architecture used in our system, after the 4096-dimensional OverFeat vectors were used as inputs.

6.1.2 System Architecture

Our system employs the siamese architecture [30], which is used for learning similarities between images by labeling pairs of images as similar or dissimilar, and maximizing the distance between different image groups. We employ user relevance feedback to divide the presented images into the two classes, i.e. images with positive feedback (relevant class) and images with negative feedback (non-relevant class). The overview of the system's architecture can be seen in Figure 4.3 in Section 4.2.

The siamese similarity metric aims to find a function that maps the input into a new space, where the target distance measure, such as the Euclidean distance, may be used to determine the proximity of two data points. This similarity function, E, is parameterized with weights W, which the system tries to learn to form the similarity metric:

$$E_W(X_1, X_2) = \|G_W(X_1) - G_W(X_2)\|, \tag{6.1}$$

where X_1 and X_2 are paired images.

This metric aims to minimize the intra-class similarity, in the case where X_1 and X_2 belong to the same class, and to maximize the inter-class similarity if X_1 and X_2 belong to different classes. The algorithm accepts a pair of observations, which when the loss function is minimized, minimizes or

maximizes the similarity metric $E_W(X_1, X_2)$ depending on whether these observations belong to the same class.

The contrastive loss function used in the siamese architecture is:

$$\begin{aligned} &L(W, (Y, X_1, X_2)^i) = \\ &(1 - Y)L_G(E_W(X_1, X_2)^i) + YL_I(E_W(X, 1, X_2)^i), \end{aligned} \tag{6.2}$$

where $(Y, X_1, X_2)^i$ is the i-th sample, which is composed of a pair of images and a label (inter- or intra-class), L_G is the partial loss function for an intra-class pair when $Y = 0$, and L_I is the partial loss function for an inter-class pair when $Y = 1$ [30].

The siamese architecture (Figure 6.1) can find an angle in the feature space that helps to distinguish between different aspects of the image, such as different position of the face or different facial expressions, making it an ideal choice for our application. An important aspect of this architecture is the fact that it generates a distance metric, which may be used to rank or generate dynamic relevance scores for all the images in a dataset.

6.1.3 Exploratory Search

In Publication V we introduced a method for exploratory search that attempts to cover the search space over the first few iterations of the session, at the same time balancing the user's interests. As the siamese neural network produces a metric representation for the images, the system is able to separate data points spatially into regions of interest. The exploratory methodologies we present here affect two variables in the framework: first, what the primary point of interest in the search space is, and second, how to explore the region around this point. For our primary location, we evaluate a focal point that is far from irrelevant images and close to the center of the cluster of relevant images. From here we explore nearby groups of images to find images that cover as large a portion of images as possible.

Central target γ is the starting point of the exploration in our method. It is chosen to be the center of the cluster of images rated as relevant, which should be close to the highest estimated relevance according to the siamese network. This point is found by minimizing the distance for each positively ranked image $x_+ \in X$, while maximizing the distance to all the negatively ranked images $x_- \in X$. The central target is thus selected by the following function:

$$\gamma = argmin_{x \in X}\big(\sum_{x \in X}(\|x, x_+\|) + \sum_{x \in X}(C - \|x, x_-\|)\big), \tag{6.3}$$

where x is an image that may or may not have yet been ranked, $\|x, x_+\|$ is the distance measure between x and x_+, and C is a suitably selected constant. The purpose of the constant is to work as the exploratory term – moving more aggressively away from the edges of positive clusters allows the method to find the center of each relevant region faster.

After the central target has been located, we define the local region as the nearest n images. This neighbourhood is then clustered to locate images that describe the content best. Defining a range parameter such as this one allows our method to utilize the relearned metric to increase the relevance. The parameter may be chosen as a proportion of the total size of the dataset balanced with the processing capabilities of the whole system.

In our system we used DBScan [37] for clustering as it generates clusters based on the form of the data itself, creating as many clusters as there are concentrated regions. This phase handles the exploratory phase of the search. As DBScan does not generate centroids, we use the image closest to the center as the centroid. This image is presented to the user if there are enough exploratory slots left. Depending on the precision rate of the previous iteration, we explore more or fewer items close to these centroids. If the previous iteration resulted in only relevant images, no exploration is done, but rather the primary central region is exploited until it is exhausted. If, on the other hand, the precision is low, we look for more images from the nearby clusters.

Our algorithm moves around the image representation space due to the changes the neural net imposes on the representation. It reconfigures the center of interest at each iteration, assuming that interesting images were successfully separated from the rest.

We tested our method against three other CBIR setups that work in comparable settings, e.g., with the same set of CNN features. Each of them is based on similarity measures and assists the user in exploring a given dataset. First, Rocchio's algorithm [101], which is a widely used ranking method for vector space settings. It finds a vector from around which images are shown to the user. The relevance score given to the method directs this vector towards a space with more related documents. We also paired Rocchio's algorithm with a classical exploratory method from multi-armed bandit literature: ϵ-greedy exploration [120]. Here, a certain number of actions are randomized to avoid policy stagnation. More precisely, the estimated optimal action is taken with a chance $1 - \epsilon$, and a random choice with chance ϵ. The initial ϵ was set to 0.5, which was annealed linearly to 0 after 10 simulation iterations with steps of 0.05. Finally, AIDE [34] is a recent exploratory framework for information retrieval that attempts to

provide a good coverage of the whole dataset. It partitions the search space into subspaces, from where it attempts to find all the relevant regions by presenting to the user samples from each of them.

To test the quality of exploration, we measured the precision of the retrieved images as well as the coverage of the search space. We measured coverage $\mathbb{C}$ as the average of distances between all retrieved items compared to the dataset size:

$$\mathbb{C} = \sum_{i,j} \left(\frac{\|x_i^s, x_j^s\|}{maxDist(X) \mid x^s \mid}\right), \tag{6.4}$$

where x^s is the set of retrieved images, $\mid x^s \mid$ its size, $\|x_i^s, x_j^s\|$ is the distance between the i:th and j:th member in the set averaged over the number of retrieved images. The term $maxDist(X)$ is the maximum distance between two points within the dataset, scaling the sum to be between 0 and 1. The greater the average sum of these distances, the further apart the data points are in the similarity space, and thus the larger the view over the data set is.

6.2 Experiments

We conducted two sets of simulation experiments to evaluate the applicability of the proposed system in interactive CBIR. For the initial study in Publication IV, we identified the following aspects of the system's performance to be crucial:

1. The system needs to be trained with only a few training examples, i.e. at each search iteration, the user is presented with only a small number of images and often provides feedback to a subset of these, and the system needs to be able to "learn" what the user is looking for based on this limited feedback;

2. The search target maybe very concrete, e.g. "red rose", or very abstract, e.g. "happiness", and the system needs to support all types of searches with varying degrees of abstractness;

3. Training time has to be below 4 seconds for the system to be interactive.

For the follow-up study in Publication V, we compared the performance of our new central exploration method to other exploratory search algorithms with the coverage score $\mathbb{C}$, as presented in Equation 6.4. The simulations we ran to assess the performance of our system were similar in

both publications, but there were differences in the evaluation. These were due to the different focus in the studied phenomenon. In the initial test we were assessing the feasibility of transfer learning the user's target in online settings. Hence, we measured the elapsed time, as well as the F1-score, which combines the classical retrieval precision with recall, which is used more in the exploratory search literature [121]. For the follow-up study, on the other hand, we were also tracking the new cumulative coverage score, which replaces the recall with a more informative measure. We also opted to follow the cumulative precision, which is measured at each iteration as the precision of all of the iterations thus far, instead of the precision of a single iteration. This metric is more informative, as the user is able to peruse all of the retrieved items at once, instead of the last result only. Furthermore, cumulative precision is still able to highlight if the search has reached a saturation point where the local context has been depleted by the search.

6.2.1 Experimental Setup

We conducted the tests in Publication IV with small training set sizes ranging from 10 up to 150 presented images. This is the average number of images in a typical CBIR search session [42], when the user is presented with 10 images in a single iteration. The same number of presented images was used in Publication V to verify the precision and time taken. For the main exploratory tests we simulated the user for 20 iterations, yielding a total of 200 images. All the reported results are averaged over five training runs for each of the existing classes in the datasets.

The target of each search is a class of images with a given label, e.g. "dogs", and the simulated user "clicks" on relevant images from a given target class at each iteration, i.e. the user feedback is 1 for images with a relevant label and 0 for the remaining images in the presented set. The number of relevant images in each iteration can vary from 0 to 10, depending on the number of relevant images in a given dataset and on the accuracy of the user throughout the search session. We assume that the user clicks only on images with the relevant label and that the user clicks on all the relevant images presented in a given iteration. To test whether the system can generalize, we also included as search targets images whose labels were not included in the training set. The search starts with a random selection of nine images from a given test dataset plus one image with the label of the target class for a specific search – this setting allows us to ensure that all the simulation experiments have a comparable starting point. In summary, our system supports the user in finding an image that best matches their

Figure 6.2: Example images from the three datasets used in our experiments.

ideal target image(s) in the manner described below. In each iteration, k images from the database $\mathbb{D}$ are presented to the user and the user selects the relevant image(s) from this set, according to the following protocol:

For each iteration $i = 1, 2, \ldots$ of the search:

- The search engine calculates a set of images $x_{i,1}, \ldots, x_{i,k} \in \mathbb{D}$ to present to the user.
- If one or more of the presented images are of interest to the user, then the user clicks on them thus providing a relevance score of 1 to the clicked images. All the remaining images in the presented set automatically receive relevance feedback of 0.
- If none of the presented images is of interest to the user, then the user proceeds to the next iteration and all the presented images automatically receive relevance feedback of 0.
- The search continues until the user finds their ideal target image.

We used Caffe [55] to produce the modifiable network described above. The simulation experiments were run on a machine with an Intel Core $i5-4430$ CPU 3.00 $\times 4$ GHz and a GeForce GTX 660 Ti.

We used three different datasets (Figure 6.2):

1. 1096 images from the MIRFlickr dataset [53] with various combinations of the following labels: mammals, birds, insects, locomotives. This dataset allowed us to test whether the learned metric is able to generalize to abstract concepts. The arbitrary nature of these classes with regards to the model of the feature extractor is perfect to demonstrate the robustness of our system: the features extracted from the images may be widely different within each label class but as long as each label can be distinguished with a set of features, the system should be able to learn it.

2. Our own collection of 294 images consisting of six different dog breeds, of which only four are included in the OverFeat classification list. This dataset allows us to test whether the model is able to learn the target in the presence of semantically related images, some of which are not included in the original scope of features used for training. Such a scenario is quite common in a CBIR setting as the search gradually narrows down towards a very specific set of images, e.g. the user starts a search for images of dogs and gradually narrows down the search to images of black dogs with pointy ears and bushy tails.

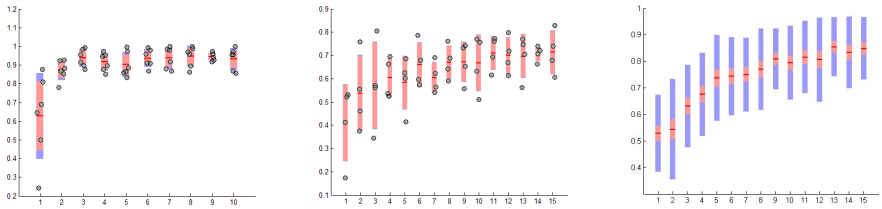
3. 300 classes from the ILSVRC2013 dataset [100], totalling 385412 images. We used this dataset to show that even if the presented images could potentially lead to hundreds of different target images, the learned representation is still able to detect the relevant features and steer the search towards the most relevant images.

Exceptions to the above were in Publication IV, where the experiments with the dog breeds dataset were simulated only for 10 search iterations due to the small size of the dataset. Furthermore, with Publication V, we sampled evenly 20000 images from 100 classes of the ILSVRC2013 dataset (from a total of 128894 images). This was due to space and time constraints set by some of the baseline methodologies.

Before running the simulations in the initial study, we conducted a number of experiments to configure our system and to learn what effect various networks parameters have on the overall performance. By varying the number of layers between one to three, we noticed smaller gains in the ILSVRC2013 dataset, while with the other datasets the accuracy improved when extra layers were added. We varied the number of training iterations and noticed no significant improvement after a thousand iterations. We settled for 1500 iterations for the final simulations. With these results, we chose a structure that takes at most 4 seconds to train, while maximizing gains from the network structure. For the siamese architecture, the training time was already closer to 4 seconds with two hidden layers, thus we chose a smaller structure, as seen in Table 6.1.

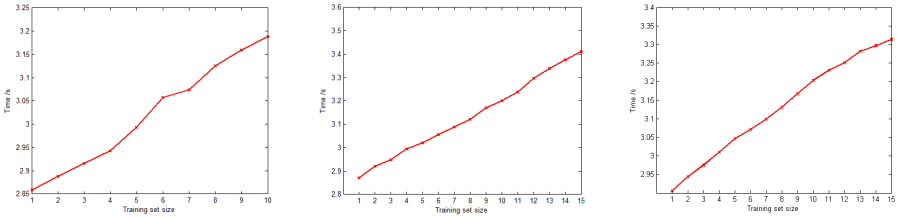
6.2.2 Experimental Results for Publication IV

The aim of the initial experiments was to test whether the system is able to find the target image or class of images with a relatively small number of training examples and whether the training time for each iteration is short enough to allow the system to be used interactively. The test results are shown in Figure 6.3 and Figure 6.4. We show the F1 measure and



(a) Dogs (b) MIRFlickr (c) ILSVRC2013

Figure 6.3: Test F1-scores (with confidence intervals) for each of the three datasets used in our experiments. The F-1 score increases with the number of iterations and thus more user feedback provided to the system.



(a) Dogs (b) MIRFlickr (c) ILSVRC2013

Figure 6.4: Training times for the three datasets used in our experiments. For all the three datasets, the training time is less than 4 seconds.

Initial test: Precision				
Data set/#Images	10	50	100	150
Dogs	0.691	0.921	0.953	0.958
MIRFlickr	0.482	0.611	0.690	0.722
ILSVRC2013	0.709	0.810	0.835	0.846

Table 6.2: Average precision with the three datasets.

the training time for each dataset. The system is able to retrieve relevant images from all the datasets within the first few iterations. Initially, the confidence intervals are wide, which reflects the uncertainty of the system with regards to the user's search target. However, as the search progresses and the system receives more and more training points and user feedback, the confidence intervals are getting narrower, indicating that the system is gradually zooming in on a specific area of the search space.

In Figure 6.4 we show the average training time for each search iteration. For each dataset, the average duration of each search iteration is below the 4 seconds required to make the system interactive from the usability perspective. This is the case even when the number of the training datapoints grows with each iteration.

6.2.3 Experimental Results for Publication V

In the follow-up study of Publication V we measured the exploratory capabilities of our algorithm compared to three baseline exploratory search methodologies. The initial average precision results are shown in Table 6.2. As can be seen, our system is able to retrieve relevant images with high accuracy even within the first few iterations. As the training set is increased to 150 images, the precisions become comparable to modern ranking methodologies.

In Table 6.3 we show the average training time for each training set size. For each dataset, the average duration for each search iteration is below 4 seconds. This makes the system interactive from the usability perspective, and grows linearly even as the number of the training data points grows larger.

In the second set of experiments we look at the performance of the various exploratory methods (Figure 6.5). We report the cumulative precision until a given point with the previous iterations acting as the context for the user throughout the search session.

Our centroid-based method gains clear advantage after approximately five iterations as the system learns the target representation. Due to the

Initial test: Time taken (s)				
Data set/#Images	10	50	100	150
Dogs	2.861	2.903	3.181	3.302
MIRFlickr	2.872	2.997	3.196	3.414
ILSVRC2013	2.911	3.042	3.208	3.315

Table 6.3: Training time in seconds for the three datasets.

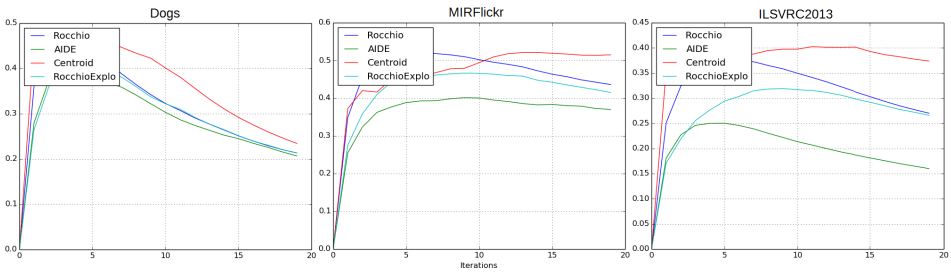


Figure 6.5: Exploration tests for the three datasets, shown with cumulative precision for the following methods: Rocchio's algorithm, Rocchio's with ϵ-greedy, central exploration, and AIDE.

small number of images present in the dog dataset, the curves for this dataset turn downwards for most methods as the search progresses, because all of the relevant images have been exhausted early on in the search. Still, our method finds the relevant images sooner and finds a larger portion of them at the end of the search. For the MIRFlickr dataset, we can see how Rocchio's exploits an early local cluster of good images, but fails later on in the search as it is unable to break out of the initial context. Meanwhile our method sacrifices a number of attempts early on and gradually achieves a larger number of correctly retrieved images.

The cumulative average coverage shows interesting trends with different methodologies. The baseline methods proceed steadily through the dataset adding relatively small gains throughout the search. Our method, on the other hand, keeps finding new regions for a long time until slowing down after approximately seven iterations. The overall coverage with our method is significantly larger in the different settings, showing how exploring the changing metric spaces helps us to find new regions of interest faster.

Finally, with the ILSVRC2013 dataset the effect of sparse targets highlights the efficiency of our method. With 100 target classes present, the local space of the initial target quickly exhausts valid images with centroid exploration. It is likely that if the first image is on the edge of the valid clus-

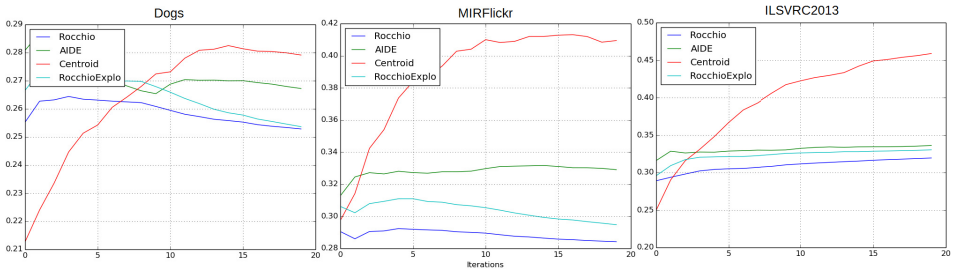


Figure 6.6: Exploration tests for the three datasets, shown with cumulative average of coverage for the following methods: Rocchio's algorithm, Rocchio's with ϵ-greedy, central exploration and AIDE.

ter of images, the nearby images will quickly present neighbouring classes. In Figure 6.6, we see the coverage for each method and dataset. The more exploratory methods keep covering a larger section of the datasets faster, while the greedy Rocchio's lags behind.

6.3 Conclusions

We presented a deep neural network framework for learning new representations in an online interactive CBIR setting. The experimental results of our initial study [89] show that it is possible to build CBIR systems that can dynamically learn the target from very limited user feedback. The system allows users to conduct searches for abstract concepts even though the system may not have been initially trained with abstract image classes. This aspect is also of high importance for symbiotic interactive systems, which can automatically detect what type of images the user might be looking for without the need on the part of the user to specify beforehand what image features would best satisfy their needs. The results showed that it is possible to produce near-instant image metrics with only a few training examples. Previous studies show that CNNs are able to abstract and discriminate beyond their original use. The descriptive value of the original features was not diminished by the small training set size used in our system, which is a promising step for using these in a CBIR setting.

The average duration of a search iteration with our pipeline is close to the 4 seconds required in interactive systems, and can be further reduced with more fine tuning of the system and improved hardware. In the future, we are planning to run more extensive simulation experiments as well as conduct extensive user studies to test the system for its applicability in

various search scenarios. Additionally, decreasing the sampling size and parallelizing the framework with GPUs are the next steps in our system's development. The goal is to reduce the processing speed to below 3 seconds in a system that is able to converge to the target image in a user study within a reasonable number of iterations.

For our Publication V, we presented a deep exploratory search framework for online interactive CBIR settings, which reacts to the users feedback dynamically and covers a larger portion of the search space than conventional retrieval tools. The system allows users to conduct searches for concepts outside of the initially used features. We showed that this transfer learning is able to extend to abstract targets, robustly learning concepts that were not originally intended for the starting features.

The system is highly dependent on good initial image features. For cases where the dataset has not been annotated but presents natural images, a good object classification CNN is required. In the case of specialized image datasets, such as medical imaging, a separate neural network should be trained just for that purpose, after which the presented methodologies are able to learn the various combinations required to identify the target. Furthermore, efficient sampling (or better hardware) is required to process more images. Fortunately, computational times with modern GPU-based neural networks scale well with larger datasets given an adequate amount of memory.

Chapter 7

Discussion

Modern search engines have come far from simple indexing systems. They use a multitude of meta-data and well designed features when bringing relevant documents for the user. Unfortunately, the most popular algorithms tend to go straight for the target, requiring familiarity either in the topic, or expertize on search engine usage to work comfortably. When using conventional search tools, one must always wonder if they truly retrieved the most important documents and information available.

Exploratory search is a relevant field for these modern problems, as it focuses on finding results that support decision making and learning. It is yet to be fully utilized in publicly available search engines, one of the reasons being the lack of qualitative analysis tools for the field, which has made it difficult to assess the performance of the new systems.

At the beginning of this thesis I made four claims:

Claim 1: It is possible to personalize the search parameters per session and user, thus accommodating users outside of population-wide tendencies.

Claim 2: People who know what they do not know give more reliable feedback, and this information can be utilized efficiently when optimizing the search system's parameters.

Claim 3: It is possible to use very general features, yet still catch a very specific target with contemporary transfer learning.

Claim 4: Users do not need to know exactly what they want at the beginning of the search session, but with appropriate support from the search engine, they can direct their search towards a desired direction.

To investigate these claims, we developed three exploratory search methodologies, two for scientific articles and one for image retrieval. Each system was developed as information retrieval tools for practical use, studying the various environments and situations users encounter when they are still learning to understand the available data.

In the Publications I, II and VI we showed that it is possible to personalize search parameters in online environments such that a wider audience of users can be modeled. This satisfies the description of my first claim. The work in [81] further investigates our method, and the results of our user studies suggest that the more users know about a given topic, the better we can utilize their feedback. This is as we asserted in my second claim.

Based on the system described in Publication III, our work in Publications IV and V showed that generic deep features, combined with transfer learning, can be utilized in online image retrieval settings to learn the intent of the users. These are in accordance with our third and fourth claim.

We also studied several avenues for better assessment of exploration during these papers, from user study settings to qualitatively measuring coverage of a search in the dataset. Indeed, part of our focus was to improve the research on exploratory search in general. Our intent is to extend this work in the future, and thus give researchers of the field better tools to evaluate the quality of their systems.

References

[1] Amazon. http://www.amazon.com/. Accessed: 2017-07-30.

[2] Google. `http://www.google.com`. Accessed: 2017-07-30.

[3] The internet movie database, imdb. http://www.imdb.com/. Accessed: 2017-07-30.

[4] Okcupid. http://www.okcupid.com/. Accessed: 2017-07-30.

[5] Wikipedia. `http://www.wikipedia.com`. Accessed: 2017-07-30.

[6] Pulkit Agrawal, Ross Girshick, and Jitendra Malik. Analyzing the performance of multilayer neural networks for object recognition. *European Conference on Computer Vision 2014 (ECCV-2014)*, 2014.

[7] Salvatore Andolina, Khalil Klouche, Jaakko Peltonen, Mohammad Hoque, Tuukka Ruotsalo, Diogo Cabral, Arto Klami, Dorota Głowacka, Patrik Floréen, and Giulio Jacucci. Intentstreams: smart parallel search streams for branching exploratory search. In *Proceedings of the 20th International Conference on Intelligent User Interfaces*, pages 300–305. ACM, 2015.

[8] K. Athukorala, A. Medlar, K. Ilves, and D. Głowacka. Balancing exploration and exploitation: Empirical parameterization of exploratory search systems. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management*, 2015.

[9] Kumaripaba Athukorala, Dorota Głowacka, Giulio Jacucci, Antti Oulasvirta, and Jilles Vreeken. Is exploratory search different? a comparison of information search behavior for exploratory and lookup tasks. *Journal of the Association for Information Science and Technology*, 67(11):2635–2651, 2016.

[10] Kumaripaba Athukorala, Alan Medlar, Antti Oulasvirta, Giulio Jacucci, and Dorota Głowacka. Beyond relevance: Adapting exploration/exploitation in information retrieval. In *Proceedings of the 21st International Conference on Intelligent User Interfaces*, pages 359–369. ACM, 2016.

[11] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47:235 – 256, 2002.

[12] Peter Auer. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3:397–422, 01 2002.

[13] Daniel T. J. Backhausen. Personalized support in exploratory search. In *Proceedings of the 4th Information Interaction in Context Symposium*, IIIX '12, pages 323–323, New York, NY, USA, 2012. ACM.

[14] Ricardo Baeza-Yates, Berthier Ribeiro-Neto, et al. *Modern information retrieval*, volume 463. ACM press New York, 1999.

[15] Marko Balabanović. Exploring versus exploiting when learning user models for text recommendation. *User Modeling and User-Adapted Interaction*, 8(1):71–102, Mar 1998.

[16] Aurélien Bellet, Amaury Habrard, and Marc Sebban. A survey on metric learning for feature vectors and structured data. *CoRR*, abs/1306.6709, 2013.

[17] Ofer Bergman, Ruth Beyth-Marom, Rafi Nachmias, Noa Gradovitch, and Steve Whittaker. Improved search engines and navigation preference in personal information management. *ACM Trans. Inf. Syst.*, 26(4):20:1–20:24, October 2008.

[18] David M. Blei. Introduction to probabilistic topic models. *Communications of the ACM*, 2011.

[19] David M. Blei. Probabilistic topic models. *Commun. ACM*, 55(4):77–84, April 2012.

[20] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March 2003.

[21] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a "siamese" time delay neural network. In *Proceedings of the 6th International Conference*

on Neural Information Processing Systems, NIPS'93, pages 737–744, San Francisco, CA, USA, 1993. Morgan Kaufmann Publishers Inc.

[22] Peter Brusilovsky, Alfred Kobsa, and Wolfgang Nejdl, editors. *The Adaptive Web: Methods and Strategies of Web Personalization*. Springer-Verlag, Berlin, Heidelberg, 2007.

[23] J. D. Brutlag, H. Hutchinson, and M. Stone. User preference and search engine latency. In *JSM Proceedings, Qualtiy and Productivity Research Section.*, 2008.

[24] Chris Buckley. Implementation of the smart information retrieval system. Technical report, Ithaca, NY, USA, 1985.

[25] Chris Buckley, Amit Singhal, and M Mitra. New retrieval approaches using smart: Trec 4. pages 25–48, 01 1995.

[26] Michael Buhrmester, Tracy Kwang, and Samuel D. Gosling. Amazon's mechanical turk. *Perspectives on Psychological Science*, 6(1):3–5, 2011. PMID: 26162106.

[27] Selcuk Candan. Exploratory search: New name for an old hat?, 2014.

[28] Robert Capra, Jaime Arguello, Annie Chen, Katie Hawthorne, Gary Marchionini, and Lee Shaw. The resultsspace collaborative search environment. In *Proceedings of the 12th ACM/IEEE-CS Joint Conference on Digital Libraries*, JCDL '12, pages 435–436, New York, NY, USA, 2012. ACM.

[29] Olivier Chapelle and Lihong Li. An empirical evaluation of thompson sampling. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 2249–2257. Curran Associates, Inc., 2011.

[30] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546 vol. 1, June 2005.

[31] E. F. Codd, S. B. Codd, and C. T. Salley. Providing OLAP (On-Line Analytical Processing) to User-Analysts: An IT Mandate. E. F. Codd and Associates, 1993.

[32] Pedram Daee, Joel Pyykkö, Dorota Głowacka, and Samuel Kaski. Interactive intent modeling from multiple feedback domains. In *Proceedings of the 21st International Conference on Intelligent User Interfaces*, IUI '16, pages 71–75, New York, NY, USA, 2016. ACM.

[33] R. Datta, J. Li, and J. Wang. Content-based image retrieval: approaches and trends of the new age. In *Multimedia information retrieval*, pages 253–262. ACM, 2005.

[34] Kyriaki Dimitriadou, Olga Papaemmanouil, and Yanlei Diao. Explore-by-example: An automatic query steering framework for interactive data exploration. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, SIGMOD '14, pages 517–528, New York, NY, USA, 2014. ACM.

[35] Alan Dix, Janet Finlay, Gregory Abowd, and Russell Beale. *Human-computer Interaction*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1997.

[36] Louis Dorard, Dorota Głowacka, and John Shawe-Taylor. Gaussian process modelling of dependencies in multi-armed bandit problems. In *Int. Symp. Op. Res*, 2009.

[37] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. pages 226–231. AAAI Press, 1996.

[38] Gerhard Fischer. User modeling in human& computer interaction. *User Modeling and User-Adapted Interaction*, 11(1-2):65–86, March 2001.

[39] Xingyu Gao, Steven C.H. Hoi, Yongdong Zhang, Ji Wan, and Jintao Li. Soml: Sparse online metric learning with application to image retrieval. 2014.

[40] Marchionini Gary and Brunk Ben. Toward a general relation browser: A gui for information architects. *JOURNAL OF DIGITAL INFORMATION*, 4, 2003.

[41] D. Głowacka and J. Shawe-Taylor. Content-based image retrieval with multinomial relevance feedback. In *Proc. of ACML*, pages 111–125, 2010.

[42] Dorota Głowacka and Sayantan Hore. Balancing exploration-explotation in image retrieval. In *Proc. of UMAP*, 2014.

[43] Dorota Głowacka, Tuukka Ruotsalo, Ksenia Konuyshkova, Kumaripaba Athukorala, Samuel Kaski, and Giulio Jacucci. Directing exploratory search: Reinforcement learning from user interactions with keywords. In *Proceedings of the 2013 International Conference on Intelligent User Interfaces*, IUI '13, pages 117–128, New York, NY, USA, 2013. ACM.

[44] Dorota Głowacka, Yee Whye Teh, and John Shawe-Taylor. Image retrieval with a bayesian model of relevance feedback. *arXiv preprint arXiv:1603.09522*, 2016.

[45] Yunchao Gong, Liwei Wang, Ruiqi Guo, and Svetlana Lazebnik. Multi-scale orderless pooling of deep convolutional activation features. *CoRR*, abs/1403.1840, 2014.

[46] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[47] S. Grunewalder, J.-Y. Audibert, M. Opper, and J. Shawe-Taylor. *Regret bounds for Gaussian process bandit problems*, pages 273–280. 2010.

[48] M. Hearst. *Search user interfaces*. Cambridge University Press, 2009.

[49] Sayantan Hore, Dorota Głowacka, Ilkka Kosunen, Kumaripaba Athukorala, and Giulio Jacucci. Futureview: Enhancing exploratory image search. In *IntRS@ RecSys*, pages 37–40, 2015.

[50] Sayantan Hore, Lasse Tyrvainen, Joel Pyykko, and Dorota Głowacka. A reinforcement learning approach to query-less image retrieval. In *International Workshop on Symbiotic Interaction*, pages 121–126. Springer, 2014.

[51] J. Hu, J. Lu, and Y.-P. Tan. Discriminative deep metric learning for face verification in the wild. In *Prof. of CVPR*, 2014.

[52] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08, pages 263–272, Washington, DC, USA, 2008. IEEE Computer Society.

[53] M. J. Huiskes and M. S. Lew. The mir flickr retrieval evaluation. In *Proc. of MIR*, 2008.

[54] Moustafa Ibrahim, Srikanth Muralidharan, Zhiwei Deng, Arash Vahdat, and Greg Mori. A hierarchical deep temporal model for group activity recognition. *CVPR 2016*, 11 2015.

[55] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv:1408.5093*, 2014.

[56] T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting clickthrough data as implicit feedback. In *SIGIR*, pages 154–161, 2005.

[57] Daniel Jurafsky and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1st edition, 2000.

[58] Ruogu Kang, Wai-Tat Fu, and Thomas George Kannampallil. Exploiting knowledge-in-the-head and knowledge-in-the-social-web: Effects of domain expertise on exploratory search in individual and social search environments. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, pages 393–402, New York, NY, USA, 2010. ACM.

[59] Antti Kangasrääsiö, Yi Chen, Dorota Głowacka, and Samuel Kaski. Interactive modeling of concept drift and errors in relevance feedback. In *Proceedings of the 2016 Conference on User Modeling Adaptation and Personalization*, pages 185–193. ACM, 2016.

[60] Antti Kangasrääsiö, Dorota Głowacka, and Samuel Kaski. Improving controllability and predictability of interactive recommendation interfaces for exploratory search. In *Proc. IUI*, pages 247–251. ACM, 2015.

[61] T. Kato, T. Kurita, N. Otsu, and K. Hirata. A sketch retrieval method for full color image database-query by visual example. In *Pattern Recognition. Computer Vision and Applications*, pages 530–533, 1992.

[62] Diane Kelly and Xin Fu. Elicitation of term relevance feedback: An investigation of term source and context. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '06, pages 453–460, New York, NY, USA, 2006. ACM.

[63] Ksenia Konyushkova and Dorota Głowacka. Content-based image retrieval with hierarchical gaussian process bandits with self-organizing maps. In *ESANN*, 2013.

[64] Markus Koskela and Jorma Laaksonen. Convolutional network features for scene recognition. In *Proceedings of the ACM International Conference on Multimedia*, MM '14, pages 1169–1172, New York, NY, USA, 2014. ACM.

[65] A. Krizhevsky and G. E. Hinton. Using very deep autoencoders for content-based image retrieval. In *Proc. of ESANN*, 2011.

[66] John D. Lafferty and David M. Blei. Correlated topic models. In Y. Weiss, P. B. Schölkopf, and J. C. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 147–154. MIT Press, 2006.

[67] T.L Lai and Herbert Robbins. Asymptotically efficient adaptive allocation rules. *Adv. Appl. Math.*, 6(1):4–22, March 1985.

[68] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[69] Michael S. Lew, Nicu Sebe, Chabane Djeraba, and Ramesh Jain. Content-based multimedia information retrieval: State of the art and challenges. *ACM Trans. Multimedia Comput. Commun. Appl.*, 2(1):1–19, February 2006.

[70] L. Li, W. Chu, J. Langford, and R.E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proc. of WWW*, 2010.

[71] Fei Liu. *The Search Performance Evaluation and Prediction in Exploratory Search*. ProQuest Dissertations Publishing, 2016.

[72] D. G. Lowe. Object recognition from local scale-invariant features. In *ICCV*, pages 1150–1157, 1999.

[73] P. C. Mahalanobis. On the generalised distance in statistics. In *Proceedings National Institute of Science, India*, volume 2, pages 49–55, April 1936.

[74] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[75] Gary Marchionini. Exploratory search: From finding to understanding. *Commun. ACM*, 49(4):41–46, April 2006.

[76] Gary Marchionini and Gary Geisler. The open video digital library. *D-Lib Magazine*, 8(12), 12 2002.

[77] Gary Marchionini and Ryen W. White. Information-seeking support systems. *Computer*, 42(3):30–32, March 2009.

[78] Brian McFee and G.R.G. Lanckriet. Metric learning to rank. *Proceedings of the 27th annual International Conference on Machine Learning (ICML)*, 2010.

[79] Alan Medlar, Kalle Ilves, Ping Wang, Wray Buntine, and Dorota Głowacka. PULP: A system for exploratory search of scientific literature. In *Proc. SIGIR*, pages 1133–1136. ACM, 2016.

[80] Alan Medlar, Joel Pyykkö, and Dorota Głowacka. Towards fine-grained adaptation of exploration/exploitation in information retrieval. In *Proceedings of the 22Nd International Conference on Intelligent User Interfaces*, IUI '17, pages 623–627, New York, NY, USA, 2017. ACM.

[81] Alan Medlar, Joel Pyykkö, and Dorota Głowacka. Fine-grained adaptation in exploratory search systems using simple user behaviour characteristics. 2018.

[82] Microsoft. Workshop on exploratory search interfaces, April 2005.

[83] Mohammad Norouzi, David M. Blei, and Ruslan Salakhutdinov. Hamming distance metric learning. In P. Bartlett, F.c.n. Pereira, C.j.c. Burges, L. Bottou, and K.q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1070–1078. 2012.

[84] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, Oct 2010.

[85] Sandeep Pandey, Deepak Agarwal, Deepayan Chakrabarti, and Vanja Josifovski. Bandits for taxonomies: A modelbased approach. In *Proc. of ICDM*, 2007.

[86] Jaakko Peltonen, Jonathan Strahl, and Patrik Floréen. Negative relevance feedback for exploratory search with visual interactive intent modeling. In *Proceedings of the 22Nd International Conference on*

Intelligent User Interfaces, IUI '17, pages 149–159, New York, NY, USA, 2017. ACM.

[87] Peter Pirolli and Stuart Card. Information foraging. pages 643–675, 1999.

[88] Pearl Pu, Li Chen, and Rong Hu. A user-centric evaluation framework for recommender systems. In *Proceedings of the Fifth ACM Conference on Recommender Systems*, RecSys '11, pages 157–164, New York, NY, USA, 2011. ACM.

[89] Joel Pyykko and Dorota Głowacka. Interactive content-based image retrieval with deep neural networks. In *International Workshop on Symbiotic Interaction*. Springer, 2016.

[90] Joel Pyykkö and Dorota Głowacka. Dynamic exploratory search in content-based image retrieval. In *Scandinavian Conference on Image Analysis*, pages 538–549. Springer, 2017.

[91] Filip Radlinski, Robert Kleinberg, and Thorsten Joachims. Learning diverse rankings with multi-armed bandits. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 784–791, New York, NY, USA, 2008. ACM.

[92] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. CNN features off-the-shelf: an astounding baseline for recognition. *CoRR*, abs/1403.6382, 2014.

[93] Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor. *Recommender Systems Handbook*. Springer-Verlag New York, Inc., New York, NY, USA, 1st edition, 2010.

[94] K. Riemer and C. Totz. The many faces of personalization - an integrative economic overview of mass customization and personalization. In *1st World Congress on Mass Customization and Personalization*, 2001.

[95] C. J. Van Rijsbergen. *Information Retrieval*. Butterworth-Heinemann, Newton, MA, USA, 2nd edition, 1979.

[96] Herbert Robbins. Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society*, 58(5):527–535, 1952.

[97] Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389, April 2009.

[98] Tuukka Ruotsalo, Kumaripaba Athukorala, Dorota Głowacka, Ksenia Konyushkova, Antti Oulasvirta, Samuli Kaipiainen, Samuel Kaski, and Giulio Jacucci. Supporting exploratory search tasks with interactive user modeling. *Proceedings of the American Society for Information Science and Technology*, 50(1):1–10, 2013.

[99] Tuukka Ruotsalo, Jaakko Peltonen, Manuel Eugster, Dorota Głowacka, Ksenia Konyushkova, Kumaripaba Athukorala, Ilkka Kosunen, Aki Reijonen, Petri Myllymäki, Giulio Jacucci, and Samuel Kaski. Directing exploratory search with interactive intent modeling. In *Proceedings of the 22Nd ACM International Conference on Information & Knowledge Management*, CIKM '13, pages 1759–1764, New York, NY, USA, 2013. ACM.

[100] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge, 2014.

[101] G. Salton. *The SMART Retrieval System—Experiments in Automatic Document Processing*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1971.

[102] Gerard Salton and Michael J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA, 1986.

[103] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *Proc. of ICLR*, 2014.

[104] Nigel Shadbolt, Tim Berners-Lee, and Wendy Hall. The semantic web revisited. *IEEE Intelligent Systems*, 21(3):96–101, May 2006.

[105] Ben Shneiderman. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 3rd edition, 1997.

[106] Amit Singhal. Modern information retrieval: A brief overview. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, pages 35–42, 2001.

[107] K. Sparck Jones, S. Walker, and S. E. Robertson. A probabilistic model of information retrieval: Development and comparative experiments. *Info. Proc. & Manag.*, 36(6):779–840, 2000.

[108] N. Srinivas, A. Krause, S. M. Kakade, and M. W. Seeger. Information-theoretic regret bounds for gaussian process optimization in the bandit setting. *IEEE Transactions on Information Theory*, 58(5):3250–3265, May 2012.

[109] Niranjan Srinivas, Andreas Krause, Matthias Seeger, and Sham M. Kakade. Gaussian process optimization in the bandit setting: No regret and experimental design. In Johannes Furnkranz and Thorsten Joachims, editors, *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 1015–1022. Omnipress, 2010.

[110] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

[111] Richard S. Sutton and Andrew G. Barto. *Introduction to Reinforcement Learning*. MIT Press, Cambridge, MA, USA, 1st edition, 1998.

[112] Jaime Teevan, Christine Alvarado, Mark S. Ackerman, and David R. Karger. The perfect search engine is not enough: A study of orienteering behavior in directed search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '04, pages 415–422, New York, NY, USA, 2004. ACM.

[113] Jaime Teevan, Susan T. Dumais, and Eric Horvitz. Personalizing search via automated analysis of interests and activities. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '05, pages 449–456, New York, NY, USA, 2005. ACM.

[114] Terry M Therneau. *A Package for Survival Analysis in S*, 2015. version 2.38.

[115] Michal Tvarožek and Mária Bieliková. *Personalized Faceted Navigation in the Semantic Web*, pages 511–515. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[116] Michal Tvarožek and Mária Bieliková. Factic: Personalized exploratory search in the semantic web. In *10th International Conference on Web Engineering*, pages 527â–530. LNCS, 2010.

[117] Lu Tyler, Pál Dávid, and Pál Martin. Contextual multi-armed bandits. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 485–492, 2010.

[118] Ji Wan, Dayong Wang, Steven Chu Hong Hoi, Pengcheng Wu, Jianke Zhu, Yongdong Zhang, and Jintao Li. Deep learning for content-based image retrieval: A comprehensive study. In *Proceedings of the 22Nd ACM International Conference on Multimedia*, MM '14, pages 157–166, New York, NY, USA, 2014. ACM.

[119] Ji Wan, Pengcheng Wu, Steven C. H. Hoi, Peilin Zhao, Xingyu Gao, Dayong Wang, Yongdong Zhang, and Jintao Li. Online learning to rank for content-based image retrieval. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, pages 2284–2290. AAAI Press, 2015.

[120] Christopher John Cornish Hellaby Watkins. *Learning from Delayed Rewards*. PhD thesis, King's College, Cambridge, UK, May 1989.

[121] R. W. White and R. A. Roth. Exploratory search: Beyond the query-response paradigm. *Synth. Lec. on Inf. Conc., Retr., and Serv.*, 1(1):1–98, 2009.

[122] Barbara M Wildemuth and Luanne Freund. Assigning search tasks designed to elicit exploratory search behaviors. In *Proc. of the Symposium on HCIIR*. ACM, 2012.

[123] J. Yang, Y.-G. Jiang, A. G. Hauptmann, and C.-W. Ngo. Evaluating bag-of-visual-words representations in scene classification. In *Multimedia Information Retrieval*, pages 197–206, 2007.

[124] Ka-Ping Yee, Kirsten Swearingen, Kevin Li, and Marti Hearst. Faceted metadata for image search and browsing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '03, pages 401–408, New York, NY, USA, 2003. ACM.

[125] Yisong Yue and Thorsten Joachims. Interactively optimizing information retrieval systems as a dueling bandits problem. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 1201–1208, New York, NY, USA, 2009. ACM.

[126] Matthew D. Zeiler and Rob Fergus. *Visualizing and Understanding Convolutional Networks*, pages 818–833. Springer International Publishing, Cham, 2014.

[127] Yi Zhang, Wei Xu, and Jamie Callan. Exploration and exploitation in adaptive filtering based on bayesian active learning. In *Proc. of ICML*, 2003.

[128] F. Zhao, Y. Huang, L. Wang, and T. Tan. Deep Semantic Ranking Based Hashing for Multi-Label Image Retrieval. *ArXiv e-prints*, January 2015.

[129] X. Zhou and T. Huang. Relevance feedback in image retrieval: A comprehensive review. *Multimedia systems*, 8(6):536–544, 2003.

[130] Moshé M. Zloof. Query by example. In *Proceedings of the May 19-22, 1975, National Computer Conference and Exposition*, AFIPS '75, pages 431–438, New York, NY, USA, 1975. ACM.