Chapter 3
Teaching IR: Curricular Considerations

Daniel Blank, Norbert Fuhr, Andreas Henrich, Thomas Mandl, Thomas Rölleke, Hinrich Schütze, and Benno Stein

3.1 Motivation

Data volumes have been growing since computers were invented, and powerful database and information retrieval technologies have been developed to manage and retrieve large volumes of data in order to turn data into information. Since the mid-1990s, not only the data volume, but in particular the number of people exposed and dependent on information supply and search also, has increased exponentially. Information (Web) search has become an inherent and frequent part in the life of billions of people, and information search is important in both professional and private contexts.

Although the preceding paragraph might seem to be the typical motivation for all texts addressing IR topics, it has important impacts on teaching IR. Whereas before the mid-1990s information search was a task mostly executed by trained and dedicated search professionals such as librarians and database administrators, the professionals, semi-professionals, and hurried end-users today share the same goal: to find relevant information quickly. Consequently, information retrieval (IR) is now part of various curricula for bachelor and master programs. These programs range from library science over information science to computer science; even programs in areas such as management science that used to regard IR as unimportant have now integrated this field as a key qualification. Basic knowledge in search engine usage and literature research is also part of curricular suggestions for school lessons.

Obviously, different target groups for teaching IR implicate different educational objectives. In the intended vocational field, IR systems might be *used*, *implemented*, *designed*, or *managed*. Fernández-Luna et al. (2009) express the variety of perspectives by a *technical continuum* ranging from nontechnical to highly technical. This continuum is spanned starting with the disciplines of

D. Blank (✉)
University of Bamberg, Bamberg, Germany
e-mail: daniel.blank@uni-bamberg.de

E. Efthimiadis et al. (eds.), *Teaching and Learning in Information Retrieval*,
The Information Retrieval Series 31, DOI 10.1007/978-3-642-22511-6_3,

psychology and general linguistics over library and information science, human–computer interaction (HCI), and management information systems to computational linguistics and computer science. These perspectives have to be considered when developing teaching concepts for IR.

There is a long way to go if we try to achieve a well-established understanding of how to teach IR. Even the authors of this chapter do not agree on all aspects considered in this chapter. We see our contribution as a first step, and by no means as a final result. We hope to stimulate discussion and to provoke a fruitful exchange of ideas, and we welcome comments on all opinions expressed in this chapter.

3.2 Toward a Curriculum for IR

To compose a curriculum in IR, we merge suggestions from various text books (cf. Sect. 3.2.3), synoptic articles such as the ones given by Croft (1995), Melucci and Hawking (2006), or Bawden et al. (2007), and IR summer schools. In the following, we will first draw a closer look at the different target groups for teaching IR. Thereafter, we will outline our proposal for an IR curriculum. Finally, we will discuss the adequacy of different forms of teaching for the different aspects and address potential groupings of IR courses as well as educational levels.

3.2.1 Educational Goals

On the background of library and information science, Bawden et al. (2007) distinguish four related, but distinct subject areas: human information behavior (HIB), information seeking (IS), information retrieval (IR), and general topics (Gen). Although the curriculum presented by Bawden et al. (2007) has a strong focus on cognitive aspects, it is useful for our considerations. Even a curriculum for computer scientists should not ignore these aspects. Nevertheless, a more system- and implementation-oriented approach might be better suited for students with a computer science background. At this point, first, important differences between potential target groups become obvious. In an overstated way, one could say that teaching IR as an advanced algorithms and data structures course might be conceivable for computer scientists, whereas an approach starting with human information needs might be appropriate for psychologists. However, in each case, a profound knowledge of the other perspectives on IR is rewarding. An IR course should not restrict itself to one specific perspective on IR but elaborate the multi-disciplinary character.

Despite this multi-disciplinary character for the respective target groups, different aspects of IR will be interesting and – even more important – qualifying for the aimed-at profession. As a consequence, it is necessary to have an understanding that the students in an IR course will have, depending on their study course, different

motivations, expectations, and personal prerequisites. To simplify things, we differentiate the audience with respect to their expected working relationship to IR systems:

1. *IR system user* (U): For students falling into this category, the efficient, goal-oriented use of IR systems is the main focus. Use often refers to research activities, which are in many cases domain specific.
2. *Management* (M): In the future working context of students falling into this category, we expect tasks regarding the supply of data and information in an organization. These professionals integrate IR into the broader picture of information and knowledge management. Consequently, there is a business-oriented view on IR, but with the need for a strong conceptual and technical background.
3. *Administration* (A): Here, the main focus is on the technical administration and optimization of search tools. Examples could be the maintenance for site search or intranet search in enterprises or domain specific Web search tools.
4. *Development* (D): This group comprises students who would like to be part of development projects in the field of IR. They may later develop and optimize systems or their components, and plan and implement innovative search technology applications.

3.2.2 Contents

Table 3.1 gives an overview of our proposed curriculum. For the different target groups, the appropriate depth of coverage is indicated. In the following, we will discuss the different topic groups – presented in bold face in Table 3.1 – in greater detail.

3.2.2.1 Introduction

Although today everybody is using search engines, the roots and the background of IR need some explanation. To this end, different concrete search situations can be considered and first naive-user experiments can be integrated into the concept.

At first, for all target groups (U, M, A, and D in Table 3.1), a detailed mission statement for IR should be given. The history of IR and its background in library science and information science should be outlined and important terms (e.g., *data*, *knowledge*, and *information*) should be introduced. To communicate the various facets of IR, different usage scenarios can be discussed, starting from Web search engines over search tasks in a digital library up to enterprise search scenarios or market investigation using IR techniques.

The knowledge of certain resources, the knowledge of necessary tools like thesauri, and the efficient use of such tools are sometimes the focus of entire courses. From a computer science perspective, awareness of professional search

Table 3.1 Topics for teaching IR along with their importance for different target groups

	U	M	A	D		U	M	A	D
Introduction					**IR evaluation**				
Motivation and overview	•	•	•	•	Performance factors and criteria	•	•	•	•
History of IR	•	•	•	•	IR performance measures	○	•	•	•
Terms and definitions	•	•	•	•	Test collections		○	•	•
IR topics and usage scenarios	•	•	•	•	System vs. user oriented		•	○	•
Efficient search: Search strategies	•	○	○	○	**Cognitive models and user interfaces**				
					Information seeking	•	•	•	○
Efficient search: Knowledge of resources	•	○	○	○	Information searching	•	•	•	○
IR versus DB-driven Retrieval	○	•	○	•	Strategic support	•	•	•	○
Language analysis					HCI aspects	•	•	•	•
Tokenization	○	○	•	•	Input modes and visualizations	○	•	•	•
Filtering (stop words, stemming, etc.)	•	•	•	•	Agent-based and mixed-initative interfaces	○	○	○	•
Meta-data	○	•	•	•	**Data mining and machine learning for IR**				
Natural language processing		○	○	•	Clustering	○	○	•	•
Text and indexing technology					Classification	○	•	•	•
Pattern matching	○	○	•	•	Mining of heterogenous data	○	○	•	•
Inverted files	○	○	•	•	**Special topics (application oriented)**				
Tree-based data structures		○	○	•	Web retrieval	○	•	•	•
Hash-based indexing		○	•	•	Semantic Web	○	•	•	•
Managing gigabytes	○	○	•	•	Multimedia retrieval	○	○	•	•
IR models					Social netwoks/media	○	•	○	•
Boolean model and its extensions	•	•	•	•	Opinion mining and sentiment analysis		○	○	•
Vector space model and its generalization	•	•	•	•	Geographic IR		○	○	•
Probabilistic retrieval		○	○	•	Information filtering	○	•	•	•
Logical approach to IR		○	○	•	Question answering	○	○	○	•
BM25 (Okapi)	○	•	•	•	**Special topics (Technological)**				
Latent variable models (e.g., LSA)		○	○	•	Cross-language IR		○	○	•
Language modeling		○	○	•	Distributed IR	○	•	•	•
					IR and ranking in databases		○	•	•
					Learning to rank		○	○	•
					Summarization		○	○	•
					XML retrieval	○	•	•	•

• = mandatory
○ = overview only
blank = dispensable

should be created and examples – maybe in a specific domain – should be presented. To this end, we have integrated the topics *search strategies* and *knowledge of resources* into the curriculum (mandatory for target group U and overview for target groups M, A, and D). For instance, the chapter *Models of the Information Seeking Process* in Hearst (2009) gives a compact overview on these aspects.

Finally, in order to sharpen the students' understanding, a discussion of the relationship between databases and IR should be given together with a consideration of the overlap (text extensions for relational databases, metadata search in IR systems, semi-structured data, and the like).

3.2.2.2 Language Analysis

First, students should be introduced to the basic problems of free text search. As a partial solution, traditional IR takes a rather simple approach to compositional semantics: under most IR models, the interpretation of a document is based on the (multi) set of the words it contains; these bag-of-words models ignore the grammatical concepts that govern sentence construction and text composition (Jurafsky and Martin 2008). Students should understand this difference and be able to argue about the loss in the representational power, the analytical simplification, and the algorithmic consequences. In particular, the basic steps to construct a bag-of-words model should be introduced, such as tokenization, term normalization, and term selection.

Tokenization is the first step in IR language analysis, where the raw character stream of a document is transformed into a stream of units, which will be used as terms later on. The subsequent steps can be grouped into the categories term normalization and term selection. Term normalization aims at the formation of term equivalence classes and includes case-folding, expanding of abbreviations, word conflation, and normalization of dates and numbers. Term selection, on the contrary, aims at extracting the content carrying words from a unit of text. Highly frequent and uniformly distributed terms such as stop words are not well suited to discriminate between relevant and nonrelevant documents, and hence are usually removed. However, students should be aware that for the analysis of a document's genre, sentiment, or authorship, stop words play an important role. Other forms of term selection include collocation analysis or key phrase extraction. Tokenization, term normalization, and term selection are language dependent, thus language identification is mandatory for language analysis. Robust language analysis is crucial to the effectiveness of an IR system. Users (target group U in Table 3.1) should understand the consequences of common language analysis techniques such as stop wording and stemming. Administrators and Developers (target groups A and D) should be aware of the challenges of and the technology for language analysis and should be able to maintain and develop a robust language analysis processing pipeline.

Natural language processing (NLP) is a large research field on its own (Manning and Schütze 1999). Students should learn that, currently, the application of NLP techniques in IR is limited to shallow techniques, but that from a technological viewpoint IR and NLP are growing together. Reasons for the latter are (1) advanced IR tasks such as plagiarism analysis, fact retrieval, or opinion mining; (2) the increased computing power; and (3) the recent advances in NLP owing to the use of machine learning techniques. Because of this development, selected NLP

technologies such as part-of-speech tagging and language modeling (LM) should be considered in the curriculum for advanced student groups (cf. Table 3.1: overview for target group M and A, mandatory for target group D).

3.2.2.3 Text and Indexing Technology

From a computer science perspective, this field is the most traditional one, covering pattern matching, efficient data storage, hashing, and text compression. Besides learning about the various methods, students also should understand their tradeoff between expressiveness and efficiency.

Patterns can be of different types, ranging from simple to complex: terms, substrings, prefixes, regular expressions, and patterns that employ a fuzzy or error-tolerant similarity measure. Consider the phonological similarity between two words as an example for a tolerant measure. Technology for pattern matching comprises classical string searching algorithms and heuristic search algorithms, but requires also sophisticated data structures, such as *n*-gram inverted files, suffix trees and suffix arrays, signature files, or tries.

The central data structure for efficient document retrieval from a large document collection is the inverted file. Specialized variants and advanced improvements exploit certain retrieval constraints and optimization potential – for example, memory size, distribution of queries, proximity and co-occurrence queries, knowledge about the update frequency of a collection, presorted lists, meta-indices, and caching strategies (Witten et al. 1999). Students in target groups A and D should gain hands-on experience with inverted indices, either by implementing a simple indexing component or by using state-of-the-art IR libraries such as Apache Lucene or Terrier.

Another retrieval technology is hashing (Stein 2007). One distinguishes exact hashing, applied for exact search (e.g., with MD5), and fuzzy hashing, also called hash-based similarity search. Students should know about these techniques and typical application areas such as near-duplicate detection and plagiarism analysis.

Text compression is employed to reduce the memory footprint of index components, or to alleviate the bottleneck when loading posting lists into main memory. It is of particular interest to students of types A and D as a means to increase retrieval efficiency and to scale IR systems to large text corpora or a high query load.

3.2.2.4 IR Models

IR models can be viewed as – mostly mathematical – frameworks to define scores of documents. The scores allow us to rank documents, and the ranking is expected to reflect the notion of relevance. Ranking is today standard, whereas the first retrieval model, namely, the Boolean model, did not provide ranking. Models such as coordination level match, extended Boolean (weighting of query terms), and fuzzy retrieval helped to add ranking to Boolean expressions. A main

breakthrough for retrieval was the usage of vector-space algebra, leading to what is referred to as the vector-space model [VSM, promoted by the SMART system (Salton et al. 1975)]. All Students (target groups U, M, A, and D) should know this model not only as a milestone in IR but also as a model delivering a retrieval quality that – until today – is a strong baseline when evaluating IR systems.

The 1970s saw the development of what became known as the probabilistic retrieval model, or more precisely, the binary independence retrieval (BIR) model, by Robertson and Sparck Jones (1976). Foundations such as the probability of relevance and the probabilistic ranking principle should be covered by all IR courses.

The 1980s brought the logical approach to IR. The probability of a logical implication between document and query is viewed to constitute the score. This "model" is mainly theoretical. It is useful to explain other IR models (Wong and Yao 1995).

The 1990s brought the retrieval model BM25 (Robertson et al. 1994). BM25 can be viewed as a successful mix of TF-IDF, BIR, and pivoted document length normalization. At least students in advanced courses (target groups M, A, and D) should not only know BM25 but also understand its background.

The late 1990s saw the paradigm of language modeling to be used in IR (Croft and Lafferty 2003). With some respect, LM is more probabilistic than the previously mentioned BIR model.

The theory and contributions of IR models are covered in extensive literature background including Wong and Yao (1995) (logical framework to explain IR models), Rölleke et al. (2006) (matrix framework to explain IR models), Robertson (2004) (understanding IDF), and a number of textbooks (Rijsbergen 1979; Belew 2000; Grossman and Frieder 2004; Manning et al. 2008; Croft et al. 2009). Overall, students should understand the necessity for ranking, the different theoretic foundations of the various models, and the parameters involved in these.

3.2.2.5 IR Evaluation

The empirical evaluation of the performance of IR systems is of central importance because the quality of a system cannot be predicted based on its components. Since an IR system ultimately needs to support the users in fulfilling their information needs, a holistic evaluation needs to set the satisfaction of the user and his or her work task as the yardstick. In addition to user studies, there is a large tradition of system-oriented evaluations following the Cranfield paradigm. All students should be aware of the different levels of evaluations that can be carried out, their potential results, and their disadvantages.

IR user studies typically provide test users with hypothetical search tasks in order to allow for a comparison. In such experiments, the user is asked to report his satisfaction with the system or its components. If the curriculum also includes classes in HCI, students might already have studied empirical evaluation and usability tests. That knowledge can be reviewed in the class. Otherwise, it should be integrated into the IR class. Students should at least be aware of some of the

difficulties involved in designing user experiments. Optimally, students of types M, A, and D should be asked to design and conduct a small user study within class themselves. They should be aware of tools that can support such a test.

Evaluations based on the Cranfield paradigm need to be the main focus of a lecture on evaluation in IR. Research has adopted this scheme, which tries to ignore subjective differences between users in order to be able to compare systems and algorithms (Buckley and Voorhees 2005). The most important measures based on relevance judgments are recall and precision. All students need to be able to know about recall and precision and should be able to interpret them.

Students of types A and D need to be able to calculate values and should know some other evaluation measure like binary preference (bpref) and cumulative gain (Järvelin and Kekäläinen 2002). In a laboratory class, these students could experiment with different measures to see whether they lead to different results.

Students need to know the main evaluation initiatives TREC[1] (Buckley and Voorhees 2005; Robertson 2008), CLEF,[2] and NTCIR[3] and should know some typical results. New tasks (Mandl 2008) and critical aspects (Al-Maskari et al. 2007) of these initiatives should be addressed for students of types A and D, as well. An advanced course for student type D could deal with the limitations of evaluation resources and the problems of their reusability and might also include the conduction of a small evaluation. Students of type D should learn about alternative approaches to evaluate enterprise or site search systems.

3.2.2.6 Cognitive models and user interfaces

Whereas database systems are mostly accessed from application programs, queries to IR systems are typically entered via a user interface. Thus, students should learn that in order to achieve a high retrieval quality for the user, cognitive aspects of interactive information access and the related problems of HCI have to be addressed.

Cognitive IR models distinguish between information seeking and searching. The former regard all activities related to information acquisition, starting from the point where the user becomes aware of an information need, until the information is found and can be applied. Popular models in this area have been developed by Ellis (1989) and Kuhlthau (1988). In contrast, information searching focuses only on the interaction of the user with an information system. Starting from Belkin's concept of "Anomalous state of knowledge" 1980 or Ingwersen's cognitive model 1992 regarding the broad context of the search, more specific approaches include the berry-picking model (Bates 1989), the concept of polyrepresentation, or Belkin's episodic model. In all these models, the classical view of a static information need is

[1] http://trec.nist.gov/, last accessed: 2010-10-26.

[2] http://www.clef-campaign.org/, last accessed: 2010-10-26.

[3] http://research.nii.ac.jp/ntcir/, last accessed: 2010-10-26.

replaced by a more dynamic view of interaction. For guiding the user in the search process, an IR system should provide strategic support; for this purpose, Bates (1990) identified four levels of search activities that are applied by experienced searchers, for which a concrete system can provide different degrees of system support.

The design of the *user interface* to an IR system also is a crucial topic (Hearst 2009). First, HCI aspects like Shneiderman's design principles 1998 and interaction styles should be introduced. Classical input interfaces include command languages, forms, and menus. A large number of visualizations for IR have been developed (Hearst 2009; Mann 2002), either as static views or allowing for direct manipulation. In order to free the user from routine tasks in search, agent-based interfaces (Lieberman 1995; Shneiderman and Maes 1997) have been proposed, but more recent developments favor mixed-initiative interfaces (Schaefer et al. 2005).

3.2.2.7 Data Mining and Machine Learning for IR

Classification methods and data mining techniques like clustering – which we will jointly refer to as "machine learning" – were originally a neglected part of the information retrieval curriculum. However, in recent years, the importance of machine learning for IR has increased significantly, both in research and in practical IR systems. This is partly due to the fact that documents are closely integrated with other data types, in particular with links and clicks on the Web; and exploiting data types such as links and clicks often necessitates the use of machine learning. Closely connected to the heterogeneity of data types in large IR systems is the fact that documents in today's typical collections are extremely diverse in quality and origin. Classification is often needed to classify documents according to their expected utility to the user. Spam detection is perhaps the most important example for this. Finally, many recent improvements in core information retrieval have come from classification and clustering, e.g., viewing document retrieval as a text classification problem (Manning et al. 2008, Chaps. 11 and 12) or improving retrieval performance using clustering (Liu and Croft 2004). These uses of machine learning in IR theory and applications should guide the selection of machine learning topics for IR courses. Machine learning methods frequently used in the context of IR include Naive Bayes, Rocchio, and Support Vector Machines (SVMs).

For clustering, the classical hierarchical clustering methods such as single-link and complete-link clustering offer students who are new to the subject easy access to the basic ideas and problems of clustering. It is important to present clustering in the context of its applications in IR such as search results clustering (Manning et al. 2008, Chap. 16) and news clustering,[4] because it is sometimes not immediately obvious to students how clustering contributes to the core goal of information finding.

[4] See, e.g. http://news.google.com/, last accessed: 2010-10-26.

PageRank (Brin and Page 1998) should be considered as a data mining technique other than clustering, since it exemplifies the interaction of textual documents with complex metadata such as links and clicks. In our experience, students show great interest in link analysis algorithms because they would like to understand how the search engines they use every day rank documents.

Much work in machine learning requires a deeper knowledge of mathematical foundations in analysis and algebra. It is, therefore, important to avoid machine learning methods that are beyond the capabilities of most students. Naive Bayes, Rocchio, hierarchical clustering, and PageRank are examples of algorithms that all students should be able to understand and are, therefore, good choices for every IR course. More advanced topics should be included in courses for target groups A and D.

3.2.2.8 Special Topics

There are many active research fields in information retrieval. Some of them are already of great commercial importance and others will have to show their potential in the future or have found their niche. One indication for which topics are currently hot is given by the sessions and workshops organized at the bigger IR conferences such as the Annual International ACM SIGIR Conference or the European Conference on IR Research (ECIR). Another indication might be seen in the evaluation tracks considered at TREC, CLEF, or the INitiative for the Evaluation of XML-Retrieval (INEX).[5]

In Table 3.1, a selection of topics is given together with a rough assessment of their importance for the target groups. In our perception, even IR users at an academic level should be aware of Web search topics such as the PageRank algorithm, problems of crawling, or the basics of search engine optimization. Semantic Web technology (Shadbolt et al. 2006), multimedia objects, and structured documents – especially XML documents – have had a strong influence on IR research, and basic knowledge in these areas will be important to assess innovations in IR in the next years. Since IR systems themselves and the collections they have to cover are becoming more and more distributed, a basic understanding of related aspects such as source selection strategies or schema integration methods seems essential. Furthermore, we have added *question answering* and *information filtering* to the topics that should be covered at least in a cursory manner for IR users because they represent specialized perspectives demonstrating the broader applicability of IR techniques.

Other topics such as *social media IR*, *cross language IR*, *geographic IR*, or *opinion mining* might also be of interest to IR users (target group U), but seem more dispensable for this target group if there is not enough time to cover these topics.

[5] http://www.inex.otago.ac.nz/about.html, last accessed: 2010-10-26.

3.2.3 Literature and Forms of Teaching

The more stable aspects of the topics listed in Table 3.1 are covered in IR textbooks (Grossman and Frieder 2004; Baeza-Yates and Ribeiro-Neto 1999; Manning et al. 2008; Croft et al. 2009). The more advanced topics currently discussed in research are addressed in IR conferences and journals such as SIGIR or ECIR.

For the different topics, different forms of teaching might be adequate. First of all, there is the *classical lecture* with the professor giving a talk and trying to engage students by interspersing questions and short discussions. Obviously, the extent to which meaningful interaction is possible depends on the number of students in the class. Another concept is the *reading club or seminar-style class*. Here, chapters of a book, research papers, or research topics are given to the students. The students have to work through these topics till the next meeting and then the contents are discussed. Obviously, this concept is more appropriate for small groups and advanced topics. However, in such situations, the dialog-oriented style of a reading club can motivate the students and foster autonomous learning. Besides lectures, there are tutorials, lab classes with hands-on training (usually performed on one's own), and projects (usually performed in groups). We will discuss the latter three in Sect. 3.3.

3.2.4 Packages and Levels

One problem with curricular considerations is that in the end, a course or a group of courses has to fit into the framework of bachelor or master programs. In this context, the available workload is usually predefined – in Europe, frequently measured in ECTS (European Credit Transfer and Accumulation System) credit points. Assuming that one ECTS credit point corresponds to a workload of 30 h for the average student, a group of comprehensive IR modules including lectures, exercises, and projects could easily comprise 20 or more ECTS credits. However, in many programs only a smaller portion will be available.

Another problem comes from the fact that at least three types of students have to be distinguished. There are *bachelor* and *master* students in programs where IR should be a part of the core curriculum. Such programs will usually be computer science, applied computer science, or information science programs. Obviously, there should be courses for both groups and, therefore, in many cases, there will be the need for an IR course for bachelor students and an (advanced) IR course for master students. With respect to the topics listed in Table 3.1, a course for bachelor students could, for example, be restricted to the extent indicated for "IR system users" in the left column. If considered useful, basic implementation techniques and additional IR models can be added if the available credit points permit. In any case, exercises and small projects should be included already in bachelor level courses to facilitate the learning success. For master students, the remaining topics together with more comprehensive projects can be offered.

Finally, there is a growing need to provide IR courses as a *secondary subject* for students in more loosely related programs. In fact, basic IR competence can be seen as a domain-spanning key qualification. If enough teaching capacity is available and the potential audience is big enough, specialized courses for IR as a secondary subject can be beneficial in this respect, because otherwise there is the danger that the expectations of the students and the previous knowledge are too diverse. On the contrary, one could argue that such a mixed audience is beneficial for the students, since it is a good preparation for working in interdisciplinary teams. Although this argument has some truth, the challenge for the lecturer is high.

3.3 Tutorials, Exercises, and IR Projects

Each IR course has to integrate practical exercises in order to improve the problem understanding and the problem-solving competence. When teaching IR in tutorials, exercises, and IR projects, various software tools can be used (e.g., search engines, catalogs, tagging systems, digital libraries, and existing research prototypes in the Web). For many algorithms in the context of IR, applets and animations can be inspected by students. The following tasks are possible even if the students do not have any programming skills:

- *Using retrieval systems to find documents relevant for given information needs*: Such exercises can help students understand why search is a hard problem and what typical capabilities of today's search systems are.
- *Evaluating and comparing the quality of retrieval results*: Given an information need, students can use search engines and compare their performance by calculating typical IR performance measures. Another interesting experience might be to examine different types of query formulation and their consequences for retrieval, e.g., in the context of image retrieval: query by sketch, query by example, and tag-based image retrieval.
- *Applying algorithms and formulas manually*: There is a rich set of fundamental IR algorithms that can be applied manually in order to foster understanding. Examples are the PageRank algorithm (Brin and Page 1998) and algorithms determinating the k most similar documents when applying the vector space model (Buckley and Lewit 1985). In addition, IR models are well suited for performing basic calculations by hand. Document representations for a small set of sample documents can be computed and matched against sample queries manually.
- *Reading exercises*: Especially in a master course, students are encouraged to gain some insights into research. Therefore, reading, summarizing, and discussing classical IR papers [e.g., from Sparck Jones and Willett (1997); Moffat et al. (2005)] or selected papers from recent IR conferences are a beneficial experience.

Students with basic programming skills can be asked to implement IR algorithms. Small source skeletons can aid in focusing on critical aspects of the algorithms and avoid tedious programming. Of course, there is also a huge number of IR libraries for different aspects of the curriculum that can be used.[6] Unix tools can also be applied to realize IR systems (Riggs 2002).

Having focused on more fine-grained exercises so far, we will now briefly describe three best practices of IR programming projects:

- *Implementing a basic IR framework from scratch*: Within this project, a small IR framework is implemented using only standard programming libraries without applying a specialized IR library or framework. The project is well suited for a bachelor course in IR. Basic programming skills as well as a course on algorithms and data structures are compulsory. Various subtasks can be identified in order to structure the work packages such as the implementation of a directory crawler, a tokenizer, several filtering steps (case-folding, stop word removal, and stemming), an inverted index, Boolean retrieval, document representations based on TF-IDF, top-*k* query processing, etc. All programming tasks are extensively explained in short briefings at the beginning of a session. Students can work in teams. If there is additional time, the framework can be extended in many directions, e.g., integrating Web crawling facilities, designing a user interface, or evaluating the system. The educational objective of this project is to deepen the students' understanding of basic IR algorithms.
- *Implementing desktop search using frameworks and libraries*: IR libraries such as Apache Lucene[7] can be used to design a small desktop search engine. Alternatively, one could devise a project concerned with the design of a prototypical Web search engine (Cacheda et al. 2008). At the beginning, the basics of the IR library that is used are explained to the students. Key concepts such as *analysis*, *documents*, and *fields* are emphasized. In a first step, students index their local file system with the help of a file crawler. Afterward, libraries for extracting the content of different document types are employed. Tools for inspecting the index such as Luke[8] can be employed analyzing the consequences of tokenizing and filtering. After having introduced the basic properties of the query engine (query syntax, document scoring, etc.), students are asked to implement query processing. There are many possibilities to extend this project: designing a user interface, extending the framework with a Web crawler, including linguistic analysis, etc.
- *Design and development of a (small) Web search engine in a Unix environment*: This project covers the aspects of IR from data analysis over indexing to retrieval

[6] Middleton and Baeza-Yates (2007) give an overview and compare multiple search engine libraries. A list of links pointing to tools and libraries can also be found in the *Teaching IR* subtree on the web site of FG-IR (http://www.fg-ir.de, last accessed: 2010-10-26).

[7] http://lucene.apache.org/, last accessed: 2010-10-26.

[8] http://code.google.com/p/luke/, last accessed: 2010-10-26.

and evaluation. Students build a tokenizer to analyze some Web pages (can be easily gathered via wgetUnix command). Then, the collection is indexed, and the students prepare a layer that receives queries and returns results and result pages (page construction, snippet generation). The project involves the development of a basic GUI. This project trains the IR and software engineering skills of students, and the motivation is to "beat" a favorite Web search engine for selected queries. Unix tools form a powerful basis for such a project (Riggs 2002).

3.4 Conclusion

When designing a curriculum for IR, the designated content, the appropriate forms of teaching, a useful breakdown into courses, and the relevance for the different target groups have to be considered. In this chapter, we tried to contribute in this respect.

Feedback on our courses which only partly implement the presented ideas at this time shows that in particular the heterogeneity in the previous knowledge and the expectations of the students are a big challenge. Specific courses for the target groups might be a solution – as far as the teaching capacity permits. However, a mixed audience can also be seen as a good preparation for practical tasks, and especially IR-related projects can benefit from the various points of view.

References

Al-Maskari A, Sanderson M, Clough P (2007) The relationship between IR effectiveness measures and user satisfaction. In: Proceedings of the 30th annual international ACM SIGIR conference, Amsterdam, pp 773–774

Baeza-Yates R, Ribeiro-Neto B (1999) Modern information retrieval. Addison-Wesley, England

Bates MJ (1989) The design of browsing and berrypicking techniques for the online search interface. Online Inf Rev 13(5):407–424

Bates MJ (1990) Where should the person stop and the information search interface start? Inf Process Manag 26(5):575–591

Bawden D, Bates J, Steinerov J, Vakkari P, Vilar P (2007) Information retrieval curricula: contexts and perspectives. In: First international BCS workshop on teaching and learning of information retrieval (TLIR 2007), London, UK. http://www.bcs.org/server.php?show = ConWebDoc.8777

Belew RK (2000) Finding out about: a cognitive perspective on search engine technology and the WWW. Cambridge University Press, Cambridge, UK

Belkin NJ (1980) Anomalous states of knowledge as a basis for information retrieval. Can J Inf Sci 5:133–143

Brin S, Page L (1998) The anatomy of a large-scale hypertextual web search engine. Comput Netw ISDN Syst 30(1–7):107–117

Buckley C, Lewit FA (1985) Optimization of inverted vector searches. In: Proceedings of the 8th annual international ACM SIGIR conference, Montréal, Québec, Canada, pp 97–110

Buckley C, Voorhees EM (2005) TREC: experiment and evaluation in information retrieval. Retrieval system evaluation. Digital libraries and electronic publishing series. MIT, Cambridge, MA, pp 53–75
Cacheda F, Fernandez D, Lopez R (2008) Experiences on a practical course of web information retrieval: developing a search engine. In: Second international BCS workshop on teaching and learning of information retrieval (TLIR 2008), London, UK. http://www.bcs.org/server.php?show = conWebDoc.22357
Croft B (1995) What do people want from information retrieval? (the top 10 research issues for companies that use and sell IR systems). D-Lib Mag 1:5
Croft B, Lafferty J (2003) (eds) Language modeling for information retrieval. The information retrieval series, vol 13. Kluwer, Amsterdam
Croft B, Metzler D, Strohman T (2009) Search engines: information retrieval in practice. Pearson Higher Education, Old Tappan, NJ
Ellis D (1989) A behavioural approach to information retrieval system design. J Document 45(3):171–212
Fernández-Luna JM, Huete JF, Macfarlane A, Efthimiadis EN (2009) Teaching and learning in information retrieval. Inf Retr 12(2):201–226
Grossman DA, Frieder O (2004) Information retrieval: algorithms and heuristics. The information retrieval series, vol 15, 2nd edn. Springer, Dordrecht
Hearst MA (2009) Search user interfaces. Cambridge University Press, Cambridge
Ingwersen P (1992) Information retrieval interaction. Taylor Graham, London
Järvelin K, Kekäläinen J (2002) Cumulated gain-based evaluation of IR techniques. ACM Trans Inf Syst Security 20(4):422–446
Jurafsky D, Martin J (2008) Speech and language processing. Prentice Hall, Upper Saddle River, NJ
Kuhlthau CC (1988) Developing a model of the library search process: cognitive and affective aspects. Ref Quart 28(2):232–242
Lieberman H (1995) Letizia: an agent that assists Web browsing. In: International joint conference on artificial intelligence, Montréal, Québec, Canada, pp 924–929
Liu X, Croft BW (2004) Cluster-based retrieval using language models. In: Proceedings of the 27th annual international ACM SIGIR conference, Sheffield, UK, pp 186–193
Mandl T (2008) Recent developments in the evaluation of information retrieval systems: moving towards diversity and practical relevance. Informatica 32:27–38
Mann TM (2002) Visualization of search results from the world wide web. Ph.D. thesis, University of Constance, http://kops.ub.uni-konstanz.de/volltexte/2002/751/pdf/Dissertation_Thomas.M.Mann_2002.V.1.07.pdf
Manning CD, Schütze H (1999) Foundations of statistical natural language processing. MIT, Cambridge, MA
Manning CD, Raghavan P, Schütze H (2008) Introduction to information retrieval. Cambridge University Press, Cambridge, MA
Melucci M, Hawking D (2006) Introduction: a perspective on web information retrieval. Inf Retr 9(2):119–122
Middleton C, Baeza-Yates R (2007) A comparison of open source search engines. Technical Report. http://wrg.upf.edu/WRG/dctos/Middleton-Baeza.pdf
Moffat A, Zobel J, Hawking D (2005) Recommended reading for IR research students. SIGIR Forum 39(2):3–14
Riggs KR (2002) Exploring IR with Unix tools. J Comput Sci Coll 17(4):179–194
Rijsbergen CJv (1979) Information retrieval. Butterworth, London, UK. http://www.dcs.gla.ac.uk/Keith/Preface.html
Robertson S (2004) Understanding inverse document frequency: On theoretical arguments for idf. J Document 60(5):503–520
Robertson S (2008) On the history of evaluation in IR. J Inf Sci 34(4):439–456
Robertson SE, Sparck Jones K (1976) Relevance weighting of search terms. J Am Soc Inf Sci 27(3):129–146

Robertson SE, Walker S, Jones S, Hancock-Beaulieu M, Gatford M (1994) Okapi at TREC-3. In: NIST Special Publication 500–226: Overview of the Third Text Retrieval Conference (TREC-3), pp 109–126

Rölleke T, Tsikrika T, Kazai G (2006) A general matrix framework for modelling information retrieval. Inf Process Manag 42(1):4–30

Salton G, Wong A, Yang CS (1975) A vector space model for automatic indexing. Commun ACM 18(11):613–620

Schaefer A, Jordan M, Klas C-P, Fuhr N (2005) Active support for query formulation in virtual digital libraries: a case study with DAFFODIL. In: 9th European conference on digital libraries, Vienna, Austria, pp 414–425

Shadbolt N, Berners-Lee T, Hall W (2006) The semantic web revisited. IEEE Intell Syst 21(3):96–101

Shneiderman B (1998) Designing the user interface. Addison-Wesley, Boston, MA

Shneiderman B, Maes P (1997) Direct manipulation vs interface agents. ACM Interact 4(6):42–61

Sparck Jones K, Willett P (eds) (1997) Readings in information retrieval. The Morgan Kaufmann series in multimedia information and systems. Morgan Kaufmann, San Francisco

Stein B (2007) Principles of hash-based text retrieval. In: Proceedings of the 30th annual international ACM SIGIR conference, Amsterdam, pp 527–534

Witten I, Moffat A, Bell T (1999) Managing gigabytes: compressing and indexing documents and images. Morgan Kaufmann, San Francisco

Wong SKM, Yao Y (1995) On modeling information retrieval with probabilistic inference. ACM Trans Inf Syst Security 13(1):38–68