Web Personalisation and Recommender Systems

Shlomo Berkovsky

Data61, CSIRO, Australia
shlomo.berkovsky@csiro.au

Abstract. The quantity of accessible information has been growing rapidly and far exceeded human processing capabilities. The sheer abundance of information often prevents users from discovering the desired information or aggravates making informed choices. This highlights the pressing need for intelligent personalised applications that simplify information access and discovery, and provide adaptive services that take into account the preferences and needs of their users. One type of personalised application that has recently become tremendously popular both in research and industry is recommender systems. These provide to users personalised recommendations about information and products they may be interested to examine or purchase. This is often achieved by exploiting social methods, which amalgamate past experiences of other users in order to identify most valuable information and products. Extensive research into recommender systems over the last decade has yielded a wide variety of techniques, which have been published at a range of reputable venues and subsequently adopted by numerous Web-sites and services. The course summarised in this paper provided the participants with a broad overview and thorough understanding of algorithms and practically deployed Web and mobile applications of personalised technologies.

Keywords: User modelling, Web personalisation, recommender systems.

1 Introduction

The sheer quantity of information accessible online has been growing relentlessly and far exceeded limited processing capabilities of users. Although this abundance of information may in some cases be perceived as valuable, it often also prevents users from discovering the desired information, or aggravates making informed and correct choices. This situation manifests in many domains and use cases, and is referred to in the literature as the information overloading problem [58]. Examples of information overloading manifest in many scenarios and we include in Figure 1 two illustrative examples from entertainment Web-sites.

The information overloading problem highlights the pressing need for intelligent personalised applications that ease information access discovery by taking into account the preferences and needs of their users [18]. These are exploited in nowadays Web-based and mobile environments for a variety of purposes, such

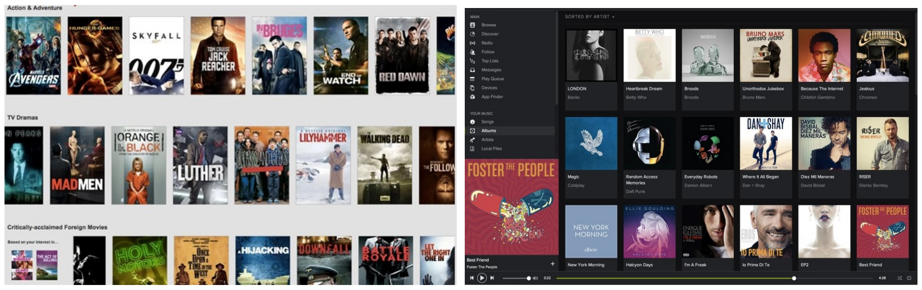


Fig. 1: Information overloading examples.

as finding relevant news items, filtering out junk emails, recommending products to purchase, summarising long textual documents, re-ordering shopping lists, or adaptively visualising complex content for the users. The basis for the personalisation in all these cases is so-called user models, which encapsulate a representation of the user's interests and information needs derived from their past user behavior and interactions [52].

One type of personalised application that has recently become popular in the research community as well as in commercial Web-sites is recommender systems [69]. These provide to users personalised recommendations of services and products they may be interested to examine or purchase. In a sense, recommender systems can be considered complementary to the well-established information filtering tools: the former recommend desired items, whereas the latter filter out the undesired ones. The generation of recommendations typically exploits information collected during the past interactions of users with the system (and other users), and the available domain information. The algorithmic approaches to recommender systems can be broadly partitioned into two fundamental families of methods: social methods that leverage the wisdom-of-the-crowd" [53] and content- and knowledge-based methods that exploit the available domain and expert-engineered knowledge [26].

Research around the areas of personalisation and recommendations has been tremendously popular at recent. In the last decade, extensive research into personalised technologies has yielded a broad spectrum of algorithmic techniques, which have been published at a wide variety of conferences (WWW, IJCAI, AAAI, ICDM, CHI, SIGIR, JCDL, WSDM, ICWSM, to name the just main ones) and journals. These works, however, were successfully taken out of the lab and have subsequently been adopted and deployed by many popular commercial Web-sites, such as Google, Yahoo, eBay, LinkedIn, Facebook, Twitter, Netflix, and many more. Generally speaking, practically any Web-site offers nowadays some form of personalised or recommendation service to its users [57].

The area of personalised technologies had reached the level of rigour and maturity that warranted a dedicated course at RuSSIR. The course summarised in this paper bridged this gap and offered to the participants an encompassing review of personalised technologies. The main objective of the course was to provide the participants with a broad overview and thorough understanding of algorithms and applications deployed by personalised technologies and recommender systems. Specifically, the course consisted of five major components listed below. This is also the flow of this summary:

– User modelling. Motivates the focus on personalisation technologies with examples of online information overloading. Following this, an overview of implicit and explicit methods for collecting user information and observing user interactions is presented. Advanced topic in user model interoperability are discussed.
– Adaptation and Web personalisation algorithms. Covers the main techniques applied for Web personalisation purposes, primarily focussing on techniques for adaptive navigation support and adaptive content presentation. Various adaptive information access methods are shown and compared.
– Collaborative recommendation methods. Introduces the area of recommender systems and then focuses on collaborative recommendation methods. Both memory-based (e.g., item-to-item collaborative filtering) and model-based (e.g., matrix factorisation) algorithms are presented, exemplified, and compared in detail.
– Content-, knowledge-based, and hybrid methods. Covers non-collaborative recommendation methods, mainly content-based and knowledge-based recommenders. These are compared analytically, naturally leading to the hybrid methods employed by many state-of-the-art recommenders. Several hybridisation designs are discussed and exemplified.
– Evaluation and emerging topics. Focuses on the topic of evaluation of personalised technologies. Both online and offline methodologies are discussed, as well as groups of evaluation metrics and specific evaluation functions. Finally, an elaborate discussion of emerging topics and open research directions concludes this summary.

2 User Modelling and Web Personalisation

With the amount of information that can be found online growing relentlessly, it is becoming increasingly harder for users to find information or remain informed about a topic. This is referred to in the literature as *information overloading* [58], term coined for an offline situation, where information is presented at a rate too fast for a person to process. These days information overloading manifests in most online information access scenarios, be it in the news, entertainment, commerce, or health domains.

Three cardinal groups of systems were proposed to address the information overloading problem. Information Retrieval systems (or, simply, search engines) assist users to locate relevant content based on their explicit queries [8]. Information Filtering systems, as the name suggests, filter out irrelevant items from the user's incoming information stream [35]. Somewhat complementary to this are Recommender Systems, which suggest items to users through highlighting the most valuable items in a user's incoming information stream [69]. Either of these technologies eventually aims at easing the information overloading and directing the users towards the more relevant bits of information.

Such a direction, however, can be provided in two ways. The non-personalised (or generic) directions will lead the user towards the most popular items, e.g., most important news articles or most watched movies. Although this partially resolves the information overloading problem, it overlooks the differences between the users. Various system users may have different interests and needs, which will practically make the target items more relevant for ones and less relevant for others. In order to incorporate such user differences, the system need to provide *personalised* services, which are tailored and adapted to individual users [62, 6, 18].

Several definitions personalisation of personalisation can be found in the literature. As collated by Adomavicius and Tuzhilin in [2], personalisation is

> “... the ability to provide content and services tailored to individuals based on knowledge about their preferences and behavior”
> “... the use of technology and customer information to tailor interactions between a business and each individual customer [aiming] to fit that customer’s stated needs, as well as needs perceived by the business based on the available customer information”
> “... the capability to customise customer communication based on knowledge, preferences, and behaviours at the time of interaction”
> “... about building customer loyalty by building a meaningful one-to-one relationship; by understanding the needs of each individual and helping satisfy a goal that efficiently and knowledgeably addresses each individual’s need in a given context”.

Despite the fact the these definitions are coming from four diverse disciplines, they align well and practically highlight the same important property: personalisation deals with tailoring user interactions (underlined in the definitions) through leveraging the available user information (double underlined).

2.1 User Modelling

The latter part, which encapsulates system representation of the user, is referred to in the literature as the *user model* [52], while the process of populating the user model data is user modelling. User modelling is an established research area that developed over the last three decades a broad range of tools and algorithms [29]. They generally aim at obtaining some user information required by the system for the purposes of the subsequent service personalisation. In the following subsections, we will discuss various considerations pertaining to the user models and the user modelling process.

To better illustrate the interplay between user modelling and personalisation, we first introduce the so-called personalisation cycle [10] (see Figure 2). The cycle consists of two main components: the user modelling component and the personalisation component. The former typically interacts with the user or

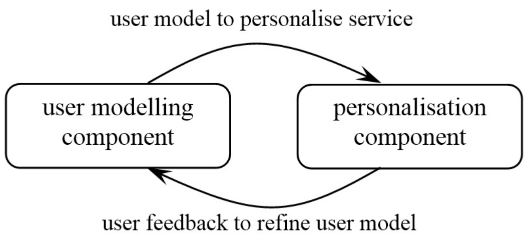

user model to personalise service
user modelling component
personalisation component
user feedback to refine user model

Fig. 2: Personalisation cycle

gathers user information and is in charge of populating the user models, while the latter is the one actually carrying out personalisation for the user, e.g., filtering news items or recommending movies. The two components are linked and equally important. The user modelling components send the user models, which facilitates personalisation, to the personalisation component. When the personalisation task is completed, the outcome is sent back to the user through. It is important to note that any user feedback on the personalised service can be exploited to refine the user models, which closes the feedback cycle. For example, if the user rejects the list of items recommended by the system, this may mean that the user models are imprecise or outdated, and, as such, the user modelling component may need to request new information from the user.

2.1.1 User Modelling Paradigms

Several user modelling paradigms can be considered. We will briefly overview them and mention their applicability to various situations. The most basic user modelling paradigm is *customisation* [7]. Here, the user controls the content of the user model and can inspect, adjust, and refine this. In essence, this can be considered as a very rudimentary level of user modelling, as the modelling functionality is carried by the user rather than by the system. This also constitutes one of serious limitations of this paradigms, as the users are unlikely to be willing to actively maintain their models.

The next paradigm is *stereotyping*, according to which the user is mapped into a stereotypical group of other users sharing a certain property or characteristic [70]. This is one of the earliest user modelling methods that resembles the personas used for user-centred design tasks. For instance, in the context of a news Web-site, a user can be interested in politics, finances, or sports, whereas a movie recommender can categorise the user as a comedy-, drama-, or thriller-lover. One can argue that since a user is mapped into a stereotypical group, the personalisation that this user modelling paradigm facilitates is not really tailored to them, but rather group-based. On the contrary, this paradigm is fairly simple and can be of use at the initial stages of user interaction, when little information about the user is available.

A more fine-grained view of the user is offered by *overlay models*, which are used often in the eLearning domain and adaptive tutoring systems [19]. These models inherit their representation from the domain models encapsulating various domain concept, e.g., a taxonomy of news topics or an ontology of movie genres, directors, actors, and so on. Following this common representation, the users are represented by numeric values reflecting their interest in (or knowledge of) various domain concepts. This value can either be a qualitative, e.g., low-medium-high, or a quantitative, e.g., probability of interest in the concept, measure. The existence of the links between the domain concepts allows for an inference of unknown interest values for some concepts.

Finally, the most advanced and most commonly used paradigm focuses on the *elicitation* of the user modelling data [61]. Users leave identifiable traces of interaction behind them, such that meaningful interpretation of these traces allows to learn precious information about the users. For example, traces like the Web pages viewed, purchased products, Facebook posts liked, reviews and comments written, online friends, groups followed on Twitter, and many others can indicate both the concepts the user is interested in and the respective level of interest. Some insightful knowledge about the user can potentially be extracted from those traces. It is important to mention that the elicitation of user models can evolve over time, distinguish between short- and long-term models, and spot fine-grained dynamic changes in user interests.

2.1.2 Implicit and Explicit User Modelling

There are two cardinal approaches for user modelling – explicit and implicit – and it is important to distinguish between them, especially in the context of the elicited user models.

The *explicit* user modelling relies on information provided by the user [65]. One of the common locations to collect this data is at the sign-up stages, when the users are often required to provide their demographic and preference information, such as address, age, gender, topics of interest, and type of subscription. However, it is not unusual for users to provide additional information over the course of their interaction with the service. Common examples of such explicit data include item ratings in eCommerce sites, restaurant reviews in Yelp, or even group membership in LinkedIn. As this information is explicitly provided by the users, it is normally up-to-date and reasonably reliable. That said, it requires an explicit user effort, which may not be seen as an inherent part of their interaction with the service, and this is the main shortcoming of the explicit user modelling. In addition to this, the users may not be willing to frequently update their information, e.g., topics of interest, such that the freshness of the explicit models needs to be re-validated occasionally.

On the flip side, *implicit* user modelling does not require any user effort, as the required user modelling information is inferred (or mined) from the observable user interactions [31]. To achieve this, the system monitors user interactions with the system and other users, and then machine learning and data mining techniques are applied to infer the required user information. There is plenty of data that can potentially be valuable for such an inference; consider the browser history, proxy server logs, past search queries, list of purchased items, examined products, bookmarked pages, online followers and friends, posts commented on, links sent to friends, brands and companies liked, events attended, and many other bits of readily available information. This information is rich and abundant, but it needs to be mined in an intelligent and interpretable way.

The data mining techniques that can be applied to this data are also diverse, such that the exact combination of the raw data to be mined and the technique to be used depends mainly on the target application, i.e., what user information needs to be inferred. It should be noted that the abundance of observable user data and the fact that the users do not need to explicitly provide any information make the implicit user modelling method preferable over the explicit one in practical personalised applications. Nevertheless, this method relies on information mined automatically, such that it can naturally be error-prone and less reliable than the explicit user modelling. Hence, much attention needs to be paid to the inference of user modelling data and the appropriateness of the data mining techniques exploited for these purposes.

Finally, the combination of explicit and implicit user modelling is a viable option. In fact, this is often used by commercial personalised systems. For example, Amazon maintains the majority of their user models in the implicit way; they monitor the purchased and examined products, reviews read by the user, sellers examined, and so on. This data informs Amazon's user modelling processes and allows them to infer most of the required user models. On top of this, some explicit information is also collected. For example, Amazon allow their users to rate the items they purchase, leave textual reviews, and even mark products that were purchased as presents for others (i.e., do not reflect user's interests and preferences). The explicit information is used to refine the implicitly built models. Thus, hybrid user modelling, which effectively combines the explicit and implicit method, usually achieves the most accurate user models and is exploited by many practical systems [14, 79, 61].

2.1.3 User Model Facets

The next question that needs to be considered in the context of user modelling refers to the user information that should be modelled by the system. There is no single answer to this question, since, for example, a news filtering system needs to exploit for personalisation purposes user models that are cardinally different from the ones exploited by a movie recommender. Hence, we can only list a number of possible facets of the user models, whereas the exact decision about the type and representation of the user modelling data is left for the system designers.

The possible facets of the raw user modelling data and features that can be derived include:

– *Knowledge.* Particularly important for eLearning systems, this information represents the level of user's knowledge in certain topics or concepts. This can be represented either by a discrete level of knowledge or by a number on a pre-defined scale. Overlay models can also be used, when the domain structure is known, stable, and can be formalised.

- *Background.* Encapsulates user's experience outside the domain of the system and the expected level of background knowledge. Possible features included in the background data are education level, profession or occupation, job responsibilities, experience in similar domains, language skills, and more. Note that these features are reasonably stable and do not change frequently over time. They can be the basis for stereotypical user modelling.
- *Interests.* User interests are crucial for various personalised information access tasks, e.g., information filtering and retrieval. Most established approaches and exploited methods originated from information retrieval, such as weighted lists of keywords and topic models. More powerful methods may exploit the available semantic information linking various concepts and topics. This facilitates hierarchical models that allow bottom-up and top-down reasoning, and, consequently, a better personalisation.
- *Goals and motivation.* This user modelling data contains the user's targets and the underlying reasons for achieving these targets. It may inform the strategic goal of personalisation and be valuable, for example, for personalised persuasion purposes. While the means for achieving the target may change, the overarching goal remains stable for a prolonged period.
- *Personality and traits.* This relatively under-explored facet refer to the user's personality traits that define the user as an individual. These imprinted features are generally stable and can be uncovered using validated behavioural tools (or inventories). They may turn out highly influential, as a plausible explanation for many observable user behaviours. For example, user's high openness to experiences may be the underlying factor explaining the observed curiosity in unusual news items and art-house movies alike.
- *Physiological signals.* Likewise, observable physiological signals may turn out to be very informative. For example, consider eye-tracking, skin conductivity, or blood pressure user data. Combined with information about the stimuli the user was exposed to, this data may paint a very precise picture of the user emotions, and, in turn, their responses to the stimuli. Again, this may reveal precious information about the emotional model of the user and allow for very accurate personalised services.
- *Interaction with others.* While all the previous facets referred to the user as an individual, this category refers to observable user interactions with other users. This information is plentiful in online social networks, which are intended to be the gateways for such interactions. Such data can be highly predictive of the user's interests and preferences, since, according to the homophyly principle, users are likely to friend and be in contact with other users similar to them.

It is important to highlight that no single one-size-fits-all user model representation can be developed, as different application domains and personalisation tasks imply different user information to be derived. Although previous work tackled the development of generic user modelling systems and servers [52, 48], the term 'generic' referred to the system rather than to the user models. That is, the aim was to develop an abstract and re-usable shell user modelling system that could be filled with domain-specific concepts and rules.

2.1.4 Challenges in User Modelling

This user modelling discussion cannot be seen complete without raising several practical challenges faced by the user modelling community. The first one is the *dynamic and sparse* nature of the user modelling data [61] available to personalised systems. It should be mentioned that some user preferences, e.g., news, books, or movies, may change quite dramatically over time. These changes are not unusual and reflect the learning and development of the user as an individual. An outdated (or new, in the extreme cases) user model may hinder the provision of personalised services, so that the information captured by the user model should change gradually and reflect the changes of the user. Questions related to the fusion of old and new user modelling data, validity and expiration of certain facets or the model, and processes that need to be put in place when dealing with a new user require more research.

Another challenge in user modelling is *context-awareness*. According to the definition coined by Dey, context is "any information that characterises the situation of an entity, which could be a person, place, or object relevant to the interaction between a user and a system" [27]. Examples of such contextual parameters are the user's location, presence of other users, time of day, day of week, weather and temperature, mood, user's device, and even communication network conditions. As pointed out in the literature, user preferences are not necessarily stable, but may be context-dependent [42, 71]. For example, a user may listen to a different music when they feel happy and when they feel sad, or prefer different meals when it is hot and when it is cold. Hence, the user modelling information should not be considered in isolation, but rather be context-dependent. While non-contextualised user models can convey the general user preferences, systems should strive to collect more data and split the user models according to the underlying contextual conditions.

This goal brings to the fore the problem of the user modelling *data sparsity*, as personalised systems may struggle to obtain enough user information, especially at the initial stages of user interaction with the system. One possible solution to this problem is through the mediation of user modelling data [12]. The main premise of the user model mediation is that data sparsity can be resolved by importing user modelling data and personal information from other domains, systems, and devices. These include other personalised systems from the same application domain, related systems from other domains, or even user's personal devices such as the mobile phone and activity tracker. Another interoperability approach considered focuses on cross-system user modelling and is particularly applicable to the Social Web environment [1]. It was observed that users often have accounts on multiple online social networks, e.g., Facebook, Twitter, LinkedIn, and Last.fm. User modelling data collected by one network may turn out valuable for another, and, as such, both could benefit from sharing their information and improving the quality of the personalised services that are provided to users.

2.2 Adaptation and Web Personalisation

Upon collecting accurate enough user modelling data, Web systems can adapt their services. The terms personalisation and adaptation are used interchangeably in this section. That said, they both refer to the same concept of exploiting the collected knowledge about the users in order to improve the functionality of a service or its interface. This can lead to a better appreciated service and higher levels of user engagement, in turn, to more satisfied and more loyal users, and, eventually, to increased profits of the service provider [51].

The research of adaptive hypermedia bloomed at the early stages of the evolution of the Web, as we know it these days. This primarily referred to the hypertext paradigm of Web 1.0 that, although not including the Social Web yet, already posed a significant information overloading problem. The multitude of links and pages on early content Web-sites (consider a news portal or a Web directory service) necessitated the site designers to either include *navigation support* tools in their sites or *adapt the presentation* of their content for the users. Either of these can simplify the discovery of the desired content and improve user experience. Adaptive hypermedia tools were taxonomised by Brusilovsky in his seminal paper [17] and are summarised in Figure 3. We will briefly overview the main techniques exploited by adaptive hypermedia.

2.2.1 Navigation Support

Offline objects are history-rich and can indicate their past usage or popularity through signals like wear and marks. On the contrary, online objects are history-poor; for instance, it is impossible to know which pages of a Web-site were popular among other users or which outgoing links on a Web-page were clicked more frequently than others. As such, Web-sites need to introduce visual cues reflecting past usage and indicating likely navigation directions for the future. These need primarily to be attached to links and pages, in order to support user navigation and simplify browsing choices.

Direct guidance, such as “next post to read” or “most interesting link to click” was among the first Web navigation support tools exploited online [46]. These may highlight links to the next content item of interest and simplify navigation through indirect link recommendations. Although often seen by users as potentially biased (recall sponsored links shown by search engines) and, thus, hardly used nowadays, this navigation support practice is still of relevance in the eLearning domain, where the content is ordered logically, objective, and highly structured. There, direct guidance may streamline user’s navigation through the educational content and at the same time not be perceived biased.

As the name suggests, *link annotation* techniques deal directly with augmenting links with visual cues reflecting their potential value for the user [40] (see examples in Figure 4). The visual cues normally include icons communicating the value of the content behind the annotated link for the user. Consider simple yet effective and easy-to-grasp signals like traffic lights, stars, heat indicators, ‘new content’ icons, or popularity scores. These can be enriched by a brief tex-

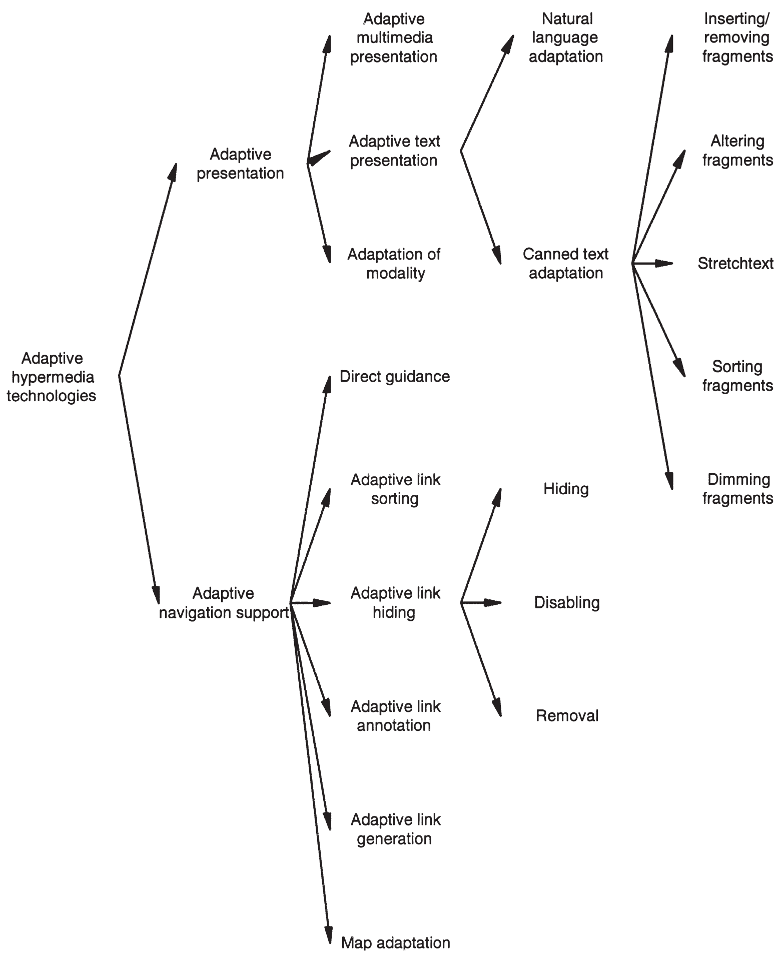
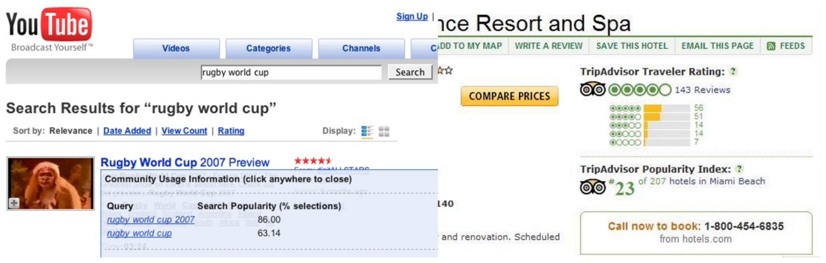


Fig. 3: Taxonomy of adaptive hypermedia techniques, from [17].

tual justification of the value of the content for the user, which can be shown, for example, when the mouse hovers over the visual cue.

Another tool, somewhat controversial and not very popular nowadays is *link ordering* [47]. It has been numerously shown that the top-shown links naturally attract stronger levels of user attention than bottom links, e.g., in user interaction with Web search engines [45]. This can be leveraged by putting the most valuable links at higher positions in the shown lists. However, this solution cannot be applied in some cases. For example, consider structured or ordered content, geographical maps, images, or even links embedded in natural language texts. In all of these, re-ordering the links may break the logical structure of the content and not be accepted by users.

Link generation (or hiding) is an effective means to promote valuable (or demote irrelevant) Web content [16]. This allows to direct (or restrict) navigation


YouTube
Broadcast Yourself™
Sign Up
Videos
Categories
Channels
rugby world cup
Search
Search Results for "rugby world cup"
Sort by: Relevance | Date Added | View Count | Rating
Display:
Rugby World Cup 2007 Preview
Community Usage Information (click anywhere to close)
nce Resort and Spa
ADD TO MY MAP
WRITE A REVIEW
SAVE THIS HOTEL
EMAIL THIS PAGE
FEEDS
COMPARE PRICES
TripAdvisor Traveler Rating:
143 Reviews
56
51
14
14
7
TripAdvisor Popularity Index:
#23 of 207 hotels in Miami Beach
and renovation. Scheduled
Call now to book: 1-800-454-6835
from hotels.com

Query	Search Popularity (% selections)
rugby world cup 2007	86.00
rugby world cup	63.14

Fig. 4: Content Annotation Examples

to content that is (or is not) expected to be valuable for the user. It is important to note that link generation or hiding may substantially damage the Web-site usability. For instance, returning site visitor may be disappointed if unable to find the links they followed at their previous visit. Thus, this technique should be applied in moderation and only when the levels of confidence in the expected value of the content for the user are high.

Finally, *site map* is an outdated navigation support tool that allows users to see the hierarchical structure of a Web-site content. This can potentially be updated in a personalised manner, to highlight the most relevant for the user content items. However, the risks of re-organising the site map are the same as those of re-ordering the links. The site usually has a pre-determined structure and altering the site map will not actually affect the structure of the site, such that the users may find this change negatively affecting usability. Thus, this technique is not in use any more.

2.2.2 Content Adaptation

While the above techniques consider navigation as a traversal of a graph linking multiple documents, the presentation (or the content) of each individual document can also be adapted. Perhaps, the most straightforward form of adaptation deals with the *adaptive presentation* of the textual content, which can be tailored to a variety of user characteristics, such as their background, language, knowledge, interests, or goals [20]. Note that here we refer to the content only, i.e., sentences, paragraphs, or even entire pages. For instance, adaptive presentation can include more content on topics of high interest or, on the contrary, remove irrelevant or already known to the user content. This technique is widely-used in the eLearning domain, where large volumes of content are available and the presented content can be tailored to the knowledge and level of the student.

Alternatively, the *presentation modality* can be adapted as well. Consider audio files that substitute written text for visually impaired people or pre-school children who cannot read yet [28]. Even for the same user the content maybe presented in different ways, e.g., depending on the user device and connectivity conditions. For example, the layout and rendering of Web-sites often differs quite substantially from the rendering of their same mobile sites, primarily due to the differences in input technologies and user interaction patterns. A more interesting form of adaptation will be across modalities [59]. For example, cooking instructions can be presented in plain text, accompanied by photos, or be filmed, so that the choice of the modality may depend on the user's cooking skills.

2.2.3 Challenges in Web Personalisation

There are several open challenges on the boundary of user modelling and personalisation, which require attention of the research community. The first one is the so-called *filter bubble*, which refers to the natural restrictions posed by personalised systems in terms of the information the user gets exposed to [66]. Naturally, personalised systems will direct users to content that matches their interests and information needs. But, if done too aggressively, this may never surface information outside this 'bubble'. For instance, consider a news filtering system, which may inadvertently hide the news items about politicians, with whom the user disagrees. Although these may be beyond the user's direct interests, important news items about them should still be shown by the system, in order to prevent the 'bubble' effect.

Another potential detrimental effect of personalisation is the *privacy leak*. On the one hand, the quality of personalised services provided to the users is directly correlated with the amount and freshness of information in the user models. On the other hand, the level of user's privacy is inversely correlated with the amount of user information that gets exposed. Hence, there is a clear conflict here, which is referred to as the privacy-personalisation trade-off [13, 32]. Indeed, many previous works showed that it is difficult to optimise both personalisation quality and users' privacy, such that various solutions that aim to find the sweet spot between the two are required. The solution is probably user- and application-dependent, and, as such, each needs to be considered individually.

Finally, an important challenge for the personalisation community is posed by the ever growing *Social Web*. This new setting is unique and troublesome at the same time, since every user is not only the consumer, but also the producer of information. Thus, the quality of information is potentially lower than on established curated channels and the information overloading problem is more acute. The user model sparsity problem in the dynamic environment of the Social Web should not be discounted either, as new information items appear more frequently and in larger volumes. Numerous personalised solutions for the Social Web have appeared recently [1]. Consider personalisation of tags in folksonomies, recommendations of friends or events, network activity feed re-ordering, or even job or company recommendations. Nevertheless, we believe that there is a broad range of yet untapped use cases that need further work.

3 Recommender Systems

Recommender systems implement a specific type of personalisation that, as the name suggests, recommends items to users. Recommender systems help users to make choices without sufficient personal experience of the alternatives. Although various items can be recommended, the roots of the widely-deployed recommenders can be traced to the eCommerce and entertainment domains, where they were used to recommend products to customers [56, 74, 75]. In a slightly simplified form, eCommerce recommenders can be seen as tools that convert Web-site visitors into customers, or at least, actively contribute to such a conversion. Nowadays, recommender systems have spread far beyond eCommerce applications and can be found in online social networks, eHealth applications, tourism planning sites, dating sites, and many more.

Just like previously discussed personalised systems, recommenders can be of value both for the user and for the service provider. For the former, they can primarily help users to explore the range of options (think of an ocean of products on Amazon or eBay), narrow down the set of relevant choices, find posts or documents that are interesting, discover new opportunities, or even be a source of entertainment. For the service provider, recommenders mainly facilitate provision of a unique personalised service to the customers. This allows to increase trust and customer loyalty, leads to more sales, improved click trough rates, or higher conversion rates, opens new opportunities for promotion and persuasion, and, ultimately, also allows to collect more user information and enhance future recommendations. As a result, more and more Web-sites and services integrate recommender systems into their systems and this technology is deployed nowadays in practically any large-scale Web-site.

One of the comprehensive formulations of the recommendation problem was given by Adomavicius and Tuzhilin in [3]. Given the profile of the *active user* that includes past user behaviour/preferences and additional information (demographic, social, task specific, and so on), the problem is to predict user-personalised relevance scores for unseen and recommendable items. Having computed these scores, the next task is to compile promising items with high score into a good[1] recommendation list and to present the recommendation list to the active user in the most way compelling way that will make them consume the recommended items.

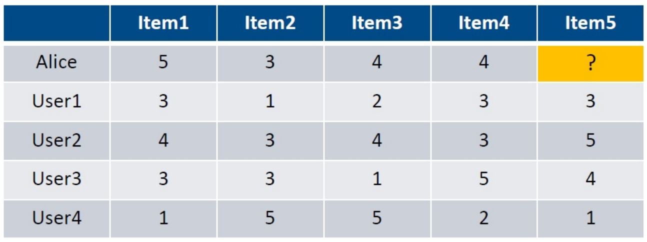
3.1 Collaborative Recommendation Methods

Collaborative recommendation methods are traditionally split into two families: *memory-based* [63] and *model-based* [53]. These two are among the most studied and most widely deployed recommendation methods. Thus, in the following subsection we will elaborate on these methods and discuss them in detail.

3.1.1 Memory-Based Collaborative Filtering

The ideas of recommenders can be traced back to the mid-90s. GroupLens mail filtering service was among the first applications [67]. This capitalised on what

[1] Questions like "what is a good recommendation list?" and "how to evaluate recommendations?" are complex questions that will be addressed later on.



	Item1	Item2	Item3	Item4	Item5
Alice	5	3	4	4	?
User1	3	1	2	3	3
User2	4	3	4	3	5
User3	3	3	1	5	4
User4	1	5	5	2	1

Fig. 5: Example collaborative filtering rating matrix.

nowadays would be called the wisdom-of-the-crowd and paved the way to *Collaborative Filtering* (CF), one of the most widely-used recommendation techniques. The underlying idea of user-based CF is that people who agreed in the past, i.e., whose opinions were found to correlate, are also likely to agree in the future. Thus, to predict in a personalised way a user's relevance score for an item, CF uses the scores of like-minded users, namely, of a set of very similar users. This derives the three main steps of memory-based CF [63], which can be seen as a variant of the well-known *K Nearest Neighbours* classification algorithm [49]:

– *Similarity computation.* Compute the similarity of the target user to other users from their past rating scores.
– *Neighbourhood formation.* Select a small subset of users with the highest similarity score to the target user.
– *Rating score prediction.* Aggregate the ratings of the neighbouring users to predict the rating of the target user.

Consider the example matrix shown in Figure 5. This contains the ratings of five users for five items given explicitly on a 1-to-5 star rating scale. The goal of the CF process in this case is to predict the rating of Alice for Item5. In the similarity computation step, the degree of Alice's similarity with each of the other four users is computed. This can practically be done using a wide variety of similarity metrics, such as the cosine similarity, Pearson's correlation, Euclidian distance, ranking correlation, or any other ordinal similarity metric [5]. The similarity score is typically normalised to the [0,1] range, where high scores indicate a high degree of user-to-user similarity.

Following this, the neighbourhood formation step typically aims at selecting a subset of most similar users. This can be done in two ways: either (i) selecting all the users, whose similarity with the target user is above a pre-defined threshold, or (ii) selecting a set of K most similar users [37]. In our example rating matrix, $K = 2$ most similar neighbours of Alice will be User2 and User3, since both, just like Alice, rated highly Item1, Item3, and Item4. The number of most similar users needs in general to be tuned to the data. Too low number of neighbours may not have enough data representativity and undermine the reliability of the predicted scores, whereas too high number of neighbours may bring into the neighbourhood users, who are not sufficiently similar to the target user. A similar question will arise if using a similarity threshold for inclusion of users in the neighbourhood.

Once the neighbours are selected, the final rating prediction step of CF deals with the extrapolation of the target user's rating from the neighbours' ratings. Assuming that User2 and User3 are included in Alice's neighbourhood, it is reasonable to believe that Alice will like Item5, since both of her neighbours rated Item5 highly. In practical CF systems, rating prediction is normally computed as a weighted average of the neighbours' ratings according to their user-to-user similarity score [36]. This way, the ratings of more similar users are assigned higher weights. Sometimes, the actual user ratings are normalised with respect to the average rating of the user (or average rating for the item), in order to eliminate individual biases.

Another variant of memory-based CF is item-based CF [56]. This can be seen as a transposed variant of the user-based CF. Namely, the similarity is computed between items rather than between users and a neighbouring set of most similar items is identified. Following this, the rating prediction is done through extrapolating the item rating from the ratings of the target user for similar items. Technically speaking, the main CF logic of K Nearest Neighbours remains unchanged; only that the similarity this time is item-to-item similarity. Thus, in the example matrix in Figure 5, the similarity of Item5 with the four other items will be computed. Assuming only $K = 1$ most similar item and that Item1 included in the neighbourhood, the rating for Item5 is again expected to be high.

Finally, we briefly discuss the advantages and shortcomings of memory-based CF. The method is simple and easy to implement. In addition, in the most basic form, it is a pure statistical method that does not rely on any *side information* about either users or items [72]. However, memory-based CF requires a large set of ratings to be available, which limits its applicability to sparse rating matrices, new (or cold) users, and new (or cold) items. Since users rarely provide feedback on many items, the sparsity problem is a significant limitation, which has been addressed in numerous works [41, 64]. In addition, the complexity of CF in quadratic, with both the number of users and items in the rating matrix. Thus, the scalability of the method is limited and it may not be suitable for large-scale Web-based recommender systems. Again, numerous works tried to improve the scalability of memory-based CF [76, 25], the most notable of which led to the development of model-based CF.

3.1.2 Model-Based Collaborative Filtering

Although early works on model-based CF had been published a long time ago [39], this research area has been substantially boosted by the Netflix Prize competition that started in 2007 [9]. There, Netflix offered a $1M award for improving the accuracy of the then deployed movie recommender system by 10%. Hundreds of teams participated in the competition for about three years and many of them used model-based methods. Since then, many model-based methods have been proposed and evaluated, the most prominent one being a latent model called *Matrix Factorisation* (MF) [53].

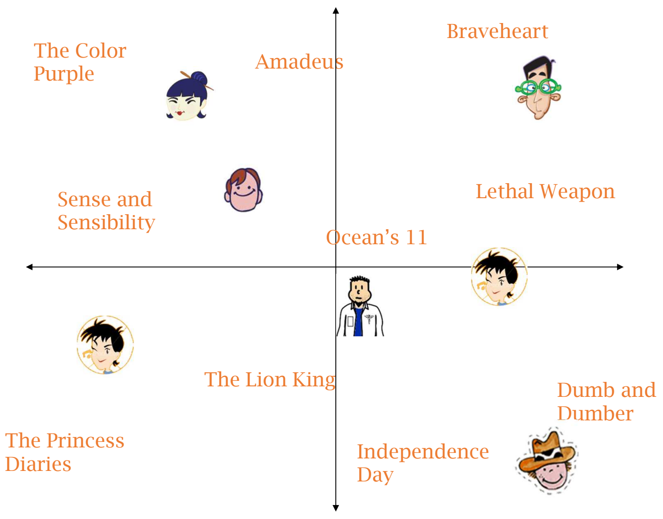


Fig. 6: Simplified Latent Space.

The main idea underlying MF assumes that the rating matrix, such as the one shown in Figure 5 can be decomposed (or factorised) into a product of two low-dimension *latent* matrices – latent user matrix and latent item matrix. These encapsulate the representations of users (or items) in the latent space, such that every row of the matrix corresponds to a user (or item) vector. The main premise of such a factorisation is that if the product of the two latent matrices can accurately approximate the existing ratings, it can also accurately predict the missing ratings. The prediction of a user rating for an item is computed as a product of the two relevant latent vectors.

For example, consider a sample two-dimensional latent space, as shown in Figure 6. Note that the dimensions of the space are latent and not necessarily explainable. However, both user vectors and item vectors are consistently mapped to the same latent space in a way that reflects their similarity by their closeness in the space, i.e., low distance. Thus, the vectors in the factorised latent matrices are mapped to the latent space, such that similar users and similar items are located nearby in the space. In this setting, the recommendation task can be seen as an expanding localised search. That is, items located near the target user in the latent space are likely to be similar to the user, i.e., to be liked by the user.

Overall, MF trades complex factorisation-based model building for fast rating prediction generation, such that the main task of MF can be seen as the factorisation of the original rating matrix into the product of the two latent matrices. For this, the latent matrices are initialised and then iteratively refined, such that the overall error function measuring the differences between the actual ratings in the dataset and the predicted ratings is being minimised. There are two popular approaches to refining the latent user and item matrices. Stochastic

Gradient Descent concurrently optimises the two matrices using the error function, whereas Alternating Least Squares fixes one of the matrices and optimises the other one and then vice versa [54].

To sum up the collaborative methods, we compare model-based CF to its memory-based MF counterpart. The most important advantage of MF over CF lies in its scalability. Since the factorisation and the production of the latent matrices can be taken offline, whereas the rating prediction only requires multiplying the relevant latent vectors, MF achieves constant-time recommendations. On top of this, the Netflix Prize competition clearly showed that MF allows the recommender to achieve better accuracy levels than CF [9, 73]. This is mainly attributed the the ability of MF to uncover often unexpected dependencies in the data, which come through the latent matrices. That said, the parameters of MF need to be tuned carefully, to prevent overfitting and allow incremental updates of the rating matrix without complete re-factorisation. Also, the reasons for item recommendations in MF may not be as evident as in CF, which complicates the explanation of recommendations and potentially detracts from user experience.

3.2 Content- and Critique-Based Methods

One of the immediate limitations of CF – be it memory- or model-based – is its pure statistical nature, which disregards any item information that may be available. However, this information may turn out valuable. For example, if the user prefers to read sports news items, and, particularly, about the games of their favourite football team, this information can drive the recommendation process. This is the underlying idea of *Content Based* (CB) filtering – it aims to learn features of the items preferred by the user and then to recommend similar items having the same features [26].

CB filtering techniques can be traced back to Information Retrieval methods, where they were extensively used to retrieve similar textual documents. The documents, as well as the user model, were represented by their textual content, e.g., using the TF-IDF vectors [8], and documents were retrieved based on a comparison between their content and the user model [55]. Turning to CB item recommendations, the main difference lies in the representation of the items and users through a set of domain features, while the filtering itself can be realised by practically any machine learning method.

A question that needs to be addressed in this context is "what can be seen as the content features"? The answer is clearly domain-dependent. For instance, for book recommendations, the book title, genre, author, type, price, and keywords can be the features, while for movie recommendations we would use the director, producer, actor, genre, and plot summary. Although the features are domain-dependent, note that the 'genre' feature is common to both recommendation tasks. Thus, as will be discussed later, it can be the basis for knowledge transfer and cross-domain recommendations [22].

It is also important to highlight the cardinal differences between CF and CB recommendations. The notion of the former is "recommend items liked by similar users", whereas the latter basically "recommends items similar to items liked by the user". Thus, the reliance of CB on the availability of numerous ratings is weaker than that of CF, and, consequently, its data sparsity problem is not as acute [11]. Indeed, a CB recommender can be bootstrapped by a relatively small set of ratings provided by the user and the sparsity problem will be alleviated by the domain features. Since no ratings of other users are required, the system cold start in CB recommendations is easier to overcome than in CF.

That said, a necessary pre-condition of CB filtering is a considerable domain engineering and knowledge, in order to be able to populate the item features. These features may not be easily available for some types of items, e.g., audio and video content, or they may include features that are difficult to extract even from textual documents, e.g., writing style, sentiment, or aesthetics. Finally, since CB recommendations rely only on the ratings provided by the target user, it may over-specialise and recommend only a narrow group of similar items [23]. This may practically limit the diversity of the recommendations and should also be considered by system designers.

Another type of recommendations that also relies heavily on the available domain-knowledge is *critique-based* recommenders [24], which can be seen as a re-incarnation of earlier knowledge-based recommenders [21] (were also referred to in the past as conversational recommenders). These are interactive recommenders that help users navigate through the search space of available options by gradually uncovering and refining the users' needs. A typical interaction with a critique-based recommender starts with the user explicitly defining their needs using a set of domain features. In response to this, the system will show a set of recommended items that answer these needs. Then, cycles of critique start: the user examines the shown items, refines their needs, and is shown with a new set of recommendations.

Note that in every cycle the user provides their 'critiques' of the shown items; thus, the name of the approach. The critiques usually come to refine the previously stated user needs and are also expressed using the same set of domain features. For example, the user may ask for cheaper items through the 'price' feature or for more recent items through the 'age' feature. Moreover, the user can ask for cheaper and more recent items at the same, which is referred to in the literature as a compound critique. Also note that in every cycle of interaction with the recommender, the recently critiqued features are assigned the highest relative weight [68].

Critique-based recommenders are often applied in dynamic environments, where the users' needs are likely to change frequently. Thus, user interaction with these recommenders resembles a dialogue with an experienced sales assistance, which puts the user in control and can potentially improve user experience. However, it is unclear to what extent critique-based recommenders are actually recommenders. They can be seen as exploration tools that only gradually elicit user preferences and assist them in navigating the search space and do not exhibit much intelligence on their own. The cost of knowledge engineering required for modelling the domain and relationships between various items in critique-based recommenders should also not be disregarded.

Hybridization method	Description
Weighted	The scores (or votes) of several recommendation techniques are combined together to produce a single recommendation.
Switching	The system switches between recommendation techniques depending on the current situation.
Mixed	Recommendations from several different recommenders are presented at the same time
Feature combination	Features from different recommendation data sources are thrown together into a single recommendation algorithm.
Cascade	One recommender refines the recommendations given by another.
Feature augmentation	Output from one technique is used as an input feature to another.
Meta-level	The model learned by one recommender is used as input to another.

Fig. 7: Taxonomy of hybrid recommender systems, from [21].

3.3 Hybrid Recommendations

As already observed, every recommendation method comes with its own advantages and shortcomings. Thus, it is only natural to consider whether individual methods can be combined, in order to produce a *hybrid* recommender system that leverages the advantages and masks the shortcomings of its individual components. The potential of hybrid recommenders has been widely recognised [33, 15] – they were found to yield the most accurate and satisfactory recommendations. This was also clearly shown by the Netflix Prize competition, where the winning approaches were ensemble models including more than 100 recommenders. Furthermore, most practically deployed commercial recommenders are rather hybrid systems.

Hence, we will list a number of high-level ways for hybridising recommender systems. Note that we only list only the ways to hybridise recommenders (or designs of hybrid systems, see Figure 7) as outlined in Burke's seminal work [21], and neither discuss specific systems nor compare their performance. Three hybridisation designs should be distinguished:

- *Pipelined Design.* In this design, output of one system – be it user models, recommendations, or any other data – serves as the input for the next system. This group includes (i) cascaded hybridisation, where one recommender produces a coarse list of recommended items that subsequently gets refined by another recommender; and (ii) meta-level hybridisation, where one recommender collects user modelling data that is subsequently re-used by another recommender.
- *Parallelised Design.* In this design, several system co-exist and independently produce their outputs, which are then combined. Hence, parallelised design can be seen as the least invasive hybridisation design. This group includes (i) weighted hybridisation, where several recommenders predict item ratings and their predictions are combined in a weighted manner; (ii) switching hybridisation, where several recommenders predict item ratings and a criterion is used to decide which one will be used for every recommendation; and (iii) mixed hybridisation, where several recommenders generate recommendation lists that are presented to the user for their selection.
- *Monolithic Design.* In this design, which is often seen as a virtual hybridisation, there exists a single recommender system that exploits user models

and data characterising several recommendation methods. This groups includes (i) feature combination, where several recommenders share and combine their user modelling data, so that the combined data is used by the target recommender; and (ii) feature augmentation, where several recommenders produce technique-specific features and data that are all fed into the target recommender.

As mentioned earlier, hybrid recommenders leverage the advantages and at the same time mask the shortcomings of individual recommendation methods. Hence, they are the most popular and widely-deployed approach in practical recommender systems. That said, there are plenty of methods to hybridise recommendation methods, there is no standard, and even in the research community there are many under-explored options. The most studied methods are weighted, switching, and mixed hybridisations, some of which are covered by [21]. On the other hand, limited work investigated cascade and feature augmentation hybridisations, whereas very little work concentrated on feature combination and meta-level hybridisations.

3.4 Evaluation of Recommender Systems

A pivotal question in recommender systems deals with the evaluation of the system performance and the quality of the generated recommendations. This is a complex and multi-faceted question that can be addressed from various perspectives, such as what items should be recommended, how to increase the uptake of recommendations, and even what recommendation strategy maximises the revenues. We argue that these questions should be methodically addressed in two ways: (i) *how to evaluate* the performance of a recommender, and (ii) *how to measure* the performance of a recommender. Thus, we will first discuss the methodologies [30] and then specific metrics [34] that can be used to evaluate recommender systems.

3.4.1 Evaluation Methodologies

To start with, we should clearly differentiate between academic and industry evaluation practices. The former usually exploit offline datasets and focus on novelty and reproducibility of the work. Thus, they typically compare the performance of their system with prior works using a relatively small set of metrics. On the other hand, practical industry evaluations focus on large-scale user sets and tangible metrics, such as click-through rates, customer satisfaction, or even sales and revenues. As a result, the methodology and evaluation metrics in the industry are highly customised and tailored to their specific recommendation scenario and application domain. Nevertheless, we will outline here a canonic way to evaluate recommenders [30, 34].

System evaluations typically start with an *offline evaluation* that uses past logs in order to train and evaluate recommendation algorithms, and establish their internal validity. For this, one randomly selected share of logs – test set –

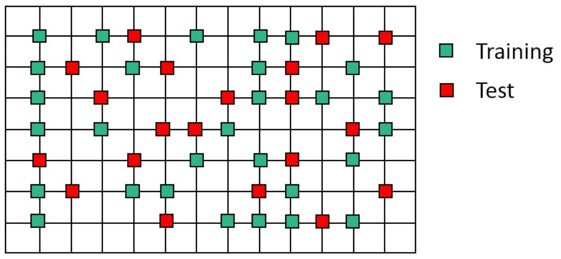


Fig. 8: Training and test data split.

is withheld and used for performance evaluation. The remaining logs are used as an input for training the algorithms and building the model. The model then is applied to the test set and the outputs of the algorithms are compared to the withheld data. The procedure of splitting the logs into the training and test set, training the algorithms on the former, and evaluating them of the latter, is repeated many times to establish statistical validity. Popular evaluation methodologies used in numerous works are 5- and 10-fold validations.

It should be noted that the training and test set should be disjoint, so that the training data does not directly affect the recommendations and contaminate the obtained results. Moreover, the test set should be selected at random across the entire rating matrix, as shown in Figure 8. Selection of complete profiles of random users for the test set will lead to cold users, for whom no training data is available, and for whom, consequently, no recommendations can be generated either. Likewise, selection of complete items for the test set will lead to cold items, for which no recommendations will be generated. Thus, it is important to ensure that all the users and all the items have an adequate representation in the training data.

Once non-performing algorithms are filtered out, the evaluation proceeds to *user studies*, which may provide a deeper insight than offline evaluations. These can be carried out as either lab studies created specifically for the evaluation purposes or field studies conducted in pre-existing real-world environments. In the lab studies, many parameters and extrinsic variables can be controlled by the experimenters, although the participants often only mimic real-world interactions and are not real system users, i.e., they do not actually purchase products or use services, which may bias the obtained results. On the contrary, field studies have the realism and validity of real users, systems, and interactions. That said, field studies usually involve only small groups of users and the parameters are much harder to control. Also, the users are intrinsically motivated to use the system, which may lead to overly positive results.

User studies typically evaluate a number of variants of the system, and, in addition to recommendation algorithms, can evaluate also user interaction aspects, such as interfaces or explanations. Thus, several experimental variants of the systems are developed, and the users are recruited and randomised across these variants. Note that intra-group, i.e., all the users are exposed to all the variants, and inter-group, i.e., different groups of users are exposed to different variants, randomisations are possible. The users then use the variants they are randomised to and various measurements are taken. These are either observations collected during the interactions or questionnaires that are administered (before and) after the experiment. Also note that qualitative methods, like questionnaires, interviews, and focus groups, and quantitative methods, like evaluation metrics and statistical tests, can both be exploited, depending on the goals of the study.

Finally, *live deployment*, as the name suggests, deals with deploying the recommender in practice and evaluating it “in the wild”. It should be noted again that this evaluation methodology is available primarily to large companies with millions of users, who can deploy and, thus, thoroughly evaluate the performance of their recommenders. They are also in the position to define their own performance metrics that reflect their business goals. Notably, this type of evaluation is perhaps the most objective and reliable one.

3.4.2 Evaluation Metrics

There is a wide range of metrics that can be used to evaluate recommender systems. Our brief overview of evaluation metrics will mainly rely on the classification introduced by Herlocker et al. [38] and recently re-visited by Gunawardana and Shani [34]. As both highlight, the metrics they cover are by no means exhaustive, and system designers should consider their own success criteria and derive respective performance metrics.

The family of *predictive accuracy* metrics focuses solely on the predicted ratings of the recommended items. Thus, it measures how close the predicted system ratings are to the real user ratings. Many derivations of the predictive accuracy metrics exist, the most popular being the Mean Average Error (MAE) and the Root Mean Square Error (RMSE). Both MAE and RMSE are predictive error metrics, such that their low values mean lower errors and accurate recommendations, while high MAE or RMSE values – inaccurate recommendations. Normalised metrics or MAE and RMSE, namely NMAE and NRMSE, are also in use. One real example of the RMSE metric was the Netflix Prize competition [9], where the participants targeted to reduce the RMSE of the recommender deployed by Netflix at the time by 10%.

Another family of metrics deals with the *classification accuracy.* These metrics, mainly inspired by Information Retrieval [8], disregard the actual item ratings, but rather focus on separating relevant and irrelevant items. This requires a relevance threshold to be defined, which may also rely on the predicted score of the items. Two popular metrics in this family are precision, which quantifies the fraction of relevant recommendations out of all the recommendations produced by the system, and recall, which quantifies the fraction of relevant recommendations out of all the good items that can be recommended by the system. We would argue that recall is less common in recommender systems than in Information Retrieval, since the complete set of relevant items is usually unknown. Also, there exist several by-products of these metrics, such as F1, Precision@K, Mean Average Precision (MAP), and more.

The last large family is *ranking accuracy* metrics. These go even further away from the predicted scores and measure whether the system ranks items in the correct way. Again, several levels of granularity for ranking accuracy metrics can be considered. For instance, rarely used ranking correlation metrics measure whether the recommended and the real items are ranked in the same order. Since the real ranking may not be known to the system, various relaxations of this metric measure, for example, if the pair-wise order of the items is correct (Normalised Distance based Performance Measure, NDPM) or if the top items in the recommendation list are good (Discounted Cumulative Gain, DCG). A normalised metric of DCG, called NDCG, is another popular metric that combines properties of the ranking and classification accuracy metrics.

Many other metrics exists and are deployed in various recommendation scenarios. We will just list and describe a few of them:

– Coverage: What portion of items (users) can be recommended (get recommendations)?
– Confidence: To what extent does the system trust its recommendations?
– Trust: How much do the users trust the recommendations generated by the system?
– Novelty: Did the user know about the recommended items or are these new to them?
– Serendipity: How surprising are the recommendations generated by the system?
– Diversity: How diverse are the items included in the recommendation list?
– Utility: How much revenue (or clicks) do the recommendations create?
– Risk: How much risk for the users do the recommendations introduce?
– Robustness: How stable are the system recommendations to attacks?
– Privacy: How much sensitive information can the recommendations leak?
– Adaptivity: How quickly can the recommender adapt to environment changes?
– Scalability: Can the recommender still function when user/item base grows?

As can be seen, the list is quite diverse and the desired metrics to optimise depend on the recommendation task and the application domain, and should be picked by the system designers. In practice, the majority of research evaluations focus on the predictive and classification accuracy metrics, while the others are less popular or are measured in combination with them.

The qualitative methods are also important and should not be discounted when considering the evaluation of recommender systems [50]. Qualitative tools have been long applied in human-computer interaction in order to understand users and their behaviour. Although not as scalable as their quantitative counterparts, they allow for a deeper insight into the observable patterns of user interaction with the system. Hundreds of validated tools for qualitative research exist, including many specifically focussing on recommender systems. To name just a few, there are validated tools for measuring perceived recommendation quality and variety, perceived system effectiveness and usefullness, cognitive effort required to use the system, choice difficulty and satisfaction, interaction and interface adequacy, intention to provide feedback, confidence and trust, and system-specific privacy concerns. These are valuable and can contribute to an encompassing system evaluation.

3.5 Challenges in Recommender Systems

We will conclude the section on recommender systems with a few practical challenges. The first one is *cross-domain recommendations* [22]. It is common for user preferences to span a number of application domains or recommender systems. For instance, it is not surprising for a user to watch science fiction movies and also read science fiction books. Cross-domain recommender systems assume that user preferences in one domain may inform user preferences in another domain and, as such, drive recommendations in another domain. This assumption has been studied and validated in many works, and also yielded a number of methods for knowledge transfer and cross-domain reasoning.

Another interesting challenge refers to *group recommendations* [60]. In many domains the consumption of the recommended items occurs not individually, but in small groups, e.g., family eating out in a restaurant or a group of friends watching a movie together. In this setting, the recommender should consider not only the preferences of the target user, who requested the recommendation, but also of other members of the group. This topic is challenging since the group preferences may not be known to the system. Although the preferences of individual users may be known, their preferences as a group potentially reflect complex social relationships between the group members. For example, adults are likely to accommodate the preferences of their children and not order spicy food in a restaurant.

A related challenge is the one of *bundle and sequential* recommendations [78, 77]. Considering a prolonged user interaction with a recommender, we observe that a recommendation is rarely requested individually. Sometimes, it comes as part of a bundle that includes accompanying product or services, e.g., hotel reservation that accompanies a booked flight or desert that accompanies a meal. In these scenarios, the two items should not considered as two standalone recommendations, but rather as part of a recommended pair of items, or a bundle. In a similar way, the user may interact with the recommender on a number of occasions, e.g., requesting for a meal recommendation several days in row. Although user preferences for a certain type of food may be clear and stable, the system should preferably diversify its recommendations and not recommend the same meal day after day.

Finally, another challenge that requires attention is that of *user-centric* matters in recommender systems [43]. Unlike other data mining applications, recommender is a user facing system, the success of which depends to a large extent on whether the users follow the recommendations. Thus, in addition to pure algorithmic challenges, much attention needs to be paid to ‘soft’ issues, such as explanation of recommendations, their persuasiveness, presentation of the recommendations, perceived user trust in the system, and others. We argue that these issues are as important as the algorithmic ones and should receive appropriate attention from the research community.

4 Summary

This paper summarises the course on Web Personalisation and Recommender Systems, which was delivered at RussIR. The course included a thorough review of the user modelling (section 2.1) and personalisation (section 2.2) technologies, as well as collaborative (section 3.1), content-based (section 3.2), and hybrid (section 3.3) recommendation methods. The techniques were accompanied by numerous examples and references to specific works, addressing various aspects of the discussed technologies.

These are all popular and steadily growing research areas. While user modelling and personalisation are established research areas[2], recommender systems is a relatively new player that attracts nevertheless much attention[3]. In addition to the conferences, two academic journals that cover these areas in depth and almost exclusively focus on personalisation are User Modelling and User Adapted Interaction (UMUAI[4]) and ACM Transactions on Interactive Intelligent Systems (TiiS[5]). The course summarised here can by no means cover such a large body of research. Therefore, we refer the interested readers to additional resources, where many more details and much more elaborate descriptions of these topics can be found.

A great starting point is "The Adaptive Web" book edited by Brusilovsky, Kobsa, and Nejdl [18]. This books covers many early techniques for user modelling and Web adaptation, as well as several applications and challenges. The basics of recommender systems are also presented in the book. However, much more detailed discussion of recommender systems can be found in the "Introduction to Recommender Systems" book by Jannach, Zanker, Felfernig, and Friedrich [44]. It is also worth to mention the recently published "Recommender Systems Textbook" by Aggarwal [4] and "Recommender Systems Handbook" edited by Ricci, Rokach, and Shapira [69]. In addition to these, an important online reference is the Coursera specialisation on Recommender Systems created by Konstan and researchers from the University of Minnesota.

We hope that this course attracted the interest of the RussIR community to personalisation and recommendations, and that it will trigger more work and submissions to the above conferences and journals.

References

1. F. Abel, E. Herder, G. Houben, N. Henze, and D. Krause. Cross-system user modeling and personalization on the social web. *User Modelling and User-Adapted Interaction*, 23(2-3):169–209, 2013.
2. G. Adomavicius and A. Tuzhilin. Personalization technologies: a process-oriented perspective. *Communication of ACM*, 48(10):83–90, 2005.

[2] The history of the UMAP conferences can be seen at `http://www.um.org/conferences/past-conferences`.
[3] The history of the RecSys conferences can be seen at `https://recsys.acm.org/`.
[4] `http://www.umuai.org/`
[5] `http://tiis.acm.org/`

3. G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, 2005.
4. C. C. Aggarwal. *Recommender Systems - The Textbook*. Springer, 2016.
5. H. J. Ahn. A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem. *Information Sciences*, 178(1):37–51, 2008.
6. S. S. Anand and B. Mobasher. Intelligent techniques for web personalization. In *Proceedings of the IJCAI Workshop on Intelligent Techniques for Web Personalization, ITWP*, pages 1–36, 2003.
7. N. Arora, X. Dreze, A. Ghose, J. D. Hess, R. Iyengar, B. Jing, Y. Joshi, V. Kumar, N. Lurie, S. Neslin, et al. Putting one-to-one marketing to work: Personalization, customization, and choice. *Marketing Letters*, 19(3-4):305–321, 2008.
8. R. Baeza-Yates and B. Ribeiro-Neto, editors. *Modern information retrieval*. Addison Wesley / ACM Press, 1999.
9. R. M. Bell and Y. Koren. Lessons from the netflix prize challenge. *SIGKDD Explorations*, 9(2):75–79, 2007.
10. S. Berkovsky, J. Freyne, and H. Oinas-Kukkonen. Influencing individually: Fusing personalization and persuasion. *ACM Transactions of Intelligent Interactive Systems*, 2(2):9, 2012.
11. S. Berkovsky, T. Kuflik, and F. Ricci. Cross-technique mediation of user models. In *Proceedings of the International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems, AH*, pages 21–30, 2006.
12. S. Berkovsky, T. Kuflik, and F. Ricci. Mediation of user models for enhanced personalization in recommender systems. *User Modelling and User-Adapted Interaction*, 18(3):245–286, 2008.
13. S. Berkovsky, T. Kuflik, and F. Ricci. The impact of data obfuscation on the accuracy of collaborative filtering. *Expert Systems with Applications*, 39(5):5033–5042, 2012.
14. D. Billsus and M. J. Pazzani. A hybrid user model for news story classification. In *Proceedings of the User Modeling Conference UM*, pages 99–108, 1999.
15. S. Bostandjiev, J. O'Donovan, and T. Höllerer. Tasteweights: a visual interactive hybrid recommender system. In *Proceedings of the ACM Conference on Recommender Systems, RecSys*, pages 35–42, 2012.
16. P. D. Bra and L. Calvi. AHA! an open adaptive hypermedia architecture. *The New Review of Hypermedia and Multimedia*, 4:115–140, 1998.
17. P. Brusilovsky. Adaptive hypermedia. *User Modelling and User-Adapted Interaction*, 11(1-2):87–110, 2001.
18. P. Brusilovsky, A. Kobsa, and W. Nejdl, editors. *The Adaptive Web, Methods and Strategies of Web Personalization*, volume 4321 of *Lecture Notes in Computer Science*. Springer, 2007.
19. P. Brusilovsky and E. Millán. User models for adaptive hypermedia and adaptive educational systems. In *The Adaptive Web, Methods and Strategies of Web Personalization*, pages 3–53, 2007.
20. A. Bunt, G. Carenini, and C. Conati. Adaptive content presentation for the web. In *The Adaptive Web, Methods and Strategies of Web Personalization*, pages 409–432, 2007.
21. R. D. Burke. Hybrid recommender systems: Survey and experiments. *User Modelling and User-Adapted Interaction*, 12(4):331–370, 2002.
22. I. Cantador, I. Fernández-Tobías, S. Berkovsky, and P. Cremonesi. Cross-domain recommender systems. In *Recommender Systems Handbook*, pages 919–959. 2015.

23. P. Castells, N. J. Hurley, and S. Vargas. Novelty and diversity in recommender systems. In *Recommender Systems Handbook*, pages 881–918. 2015.
24. L. Chen and P. Pu. Critiquing-based recommenders: survey and emerging trends. *User Modelling and User-Adapted Interaction*, 22(1-2):125–150, 2012.
25. A. Das, M. Datar, A. Garg, and S. Rajaram. Google news personalization: scalable online collaborative filtering. In *Proceedings of the International Conference on World Wide Web, WWW*, pages 271–280, 2007.
26. M. de Gemmis, P. Lops, C. Musto, F. Narducci, and G. Semeraro. Semantics-aware content-based recommender systems. In *Recommender Systems Handbook*, pages 119–159. 2015.
27. A. K. Dey. Understanding and using context. *Personal and Ubiquitous Computing*, 5(1):4–7, 2001.
28. N. Fernandes, N. Kaklanis, K. Votis, D. Tzovaras, and L. Carriço. An analysis of personalized web accessibility. In *Proceedings of the International Web for All Conference, W4A*, pages 1–10, 2014.
29. J. Fink and A. Kobsa. A review and analysis of commercial user modeling servers for personalization on the world wide web. *User Modelling and User-Adapted Interaction*, 10(2-3):209–249, 2000.
30. J. Freyne and S. Berkovsky. Evaluating recommender systems for supportive technologies. In *User Modeling and Adaptation for Daily Routines*, pages 195–217. Springer, 2013.
31. E. Frias-Martinez, S. Y. Chen, and X. Liu. Survey of data mining approaches to user modeling for adaptive hypermedia. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 36(6):734–749, 2006.
32. A. Friedman, B. P. Knijnenburg, K. Vanhecke, L. Martens, and S. Berkovsky. Privacy aspects of recommender systems. In *Recommender Systems Handbook*, pages 649–688. 2015.
33. A. Gunawardana and C. Meek. A unified approach to building hybrid recommender systems. In *Proceedings of the ACM Conference on Recommender Systems, RecSys*, pages 117–124, 2009.
34. A. Gunawardana and G. Shani. Evaluating recommender systems. In *Recommender Systems Handbook*, pages 265–308. 2015.
35. U. Hanani, B. Shapira, and P. Shoval. Information filtering: Overview of issues, research and systems. *User Modelling and User-Adapted Interaction*, 11(3):203–259, 2001.
36. J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the Annual International ACM Conference on Research and Development in Information Retrieval, SIGIR*, pages 230–237, 1999.
37. J. L. Herlocker, J. A. Konstan, and J. Riedl. An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. *Information Retrieval*, 5(4):287–310, 2002.
38. J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, 22(1):5–53, 2004.
39. T. Hofmann. Latent semantic models for collaborative filtering. *ACM Transactions on Information Systems*, 22(1):89–115, 2004.
40. H. Hohl, H. Böcker, and R. Gunzenhäuser. Hypadapter: An adaptive hypertext system for exploratory learning and programming. *User Modelling and User-Adapted Interaction*, 6(2-3):131–156, 1996.

41. Z. Huang, H. Chen, and D. D. Zeng. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Transactions on Information Systems*, 22(1):116–142, 2004.
42. A. Jameson. Modelling both the context and the user. *Personal and Ubiquitous Computing*, 5(1):29–33, 2001.
43. A. Jameson, M. C. Willemsen, A. Felfernig, M. de Gemmis, P. Lops, G. Semeraro, and L. Chen. Human decision making and recommender systems. In *Recommender Systems Handbook*, pages 611–648. 2015.
44. D. Jannach, M. Zanker, A. Felfernig, and G. Friedrich. *Recommender Systems - An Introduction*. Cambridge University Press, 2010.
45. T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD*, pages 133–142, 2002.
46. T. Joachims, D. Freitag, and T. M. Mitchell. Web watcher: A tour guide for the world wide web. In *Proceedings of the International Joint Conference on Artificial Intelligence, IJCAI*, pages 770–777, 1997.
47. C. A. Kaplan, J. Fenwick, and J. Chen. Adaptive hypertext navigation based on user goals and context. *User Modelling and User-Adapted Interaction*, 3(3):193–220, 1993.
48. J. Kay, B. Kummerfeld, and P. Lauder. Personis: A server for user models. In *Proceedings of the International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems, AH*, pages 203–212, 2002.
49. J. M. Keller, M. R. Gray, and J. A. Givens. A fuzzy k-nearest neighbor algorithm. *IEEE Trans. Systems, Man, and Cybernetics*, 15(4):580–585, 1985.
50. B. P. Knijnenburg and M. C. Willemsen. Evaluating recommender systems with user experiments. In *Recommender Systems Handbook*, pages 309–352. 2015.
51. B. P. Knijnenburg, M. C. Willemsen, Z. Gantner, H. Soncu, and C. Newell. Explaining the user experience of recommender systems. *User Modelling and User-Adapted Interaction*, 22(4-5):441–504, 2012.
52. A. Kobsa. Generic user modeling systems. *User Modelling and User-Adapted Interaction*, 11(1-2):49–63, 2001.
53. Y. Koren and R. M. Bell. Advances in collaborative filtering. In *Recommender Systems Handbook*, pages 77–118. 2015.
54. Y. Koren, R. M. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *IEEE Computer*, 42(8):30–37, 2009.
55. M. S. Lew, N. Sebe, C. Djeraba, and R. Jain. Content-based multimedia information retrieval: State of the art and challenges. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 2(1):1–19, 2006.
56. G. Linden, B. Smith, and J. York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, 2003.
57. J. Lu, D. Wu, M. Mao, W. Wang, and G. Zhang. Recommender system application developments: A survey. *Decision Support Systems*, 74:12–32, 2015.
58. P. Maes. Agents that reduce work and information overload. *Communications of ACM*, 37(7):30–40, 1994.
59. A. Marinc, C. Stocklöw, A. Braun, C. Limberger, C. Hofmann, and A. Kuijper. Interactive personalization of ambient assisted living environments. In *Proceedings of the Symposium on Human Interface, HCII*, pages 567–576, 2011.
60. J. Masthoff. Group recommender systems: Aggregation, satisfaction and group attributes. In *Recommender Systems Handbook*, pages 743–776. 2015.
61. B. Mobasher. Data mining for web personalization. In *The Adaptive Web, Methods and Strategies of Web Personalization*, pages 90–135, 2007.

62. B. Mobasher, R. Cooley, and J. Srivastava. Automatic personalization based on web usage mining. *Communications of ACM*, 43(8):142–151, 2000.
63. X. Ning, C. Desrosiers, and G. Karypis. A comprehensive survey of neighborhood-based recommendation methods. In *Recommender Systems Handbook*, pages 37–76. 2015.
64. W. Pan, E. W. Xiang, N. N. Liu, and Q. Yang. Transfer learning in collaborative filtering for sparsity reduction. In *Proceedings of the AAAI Conference on Artificial Intelligence, AAAI*, 2010.
65. C. L. Paris. The use of explicit user models in a generation system for tailoring answers to the users level of expertise. In *User Models in Dialog Systems*, pages 200–232. Springer Berlin Heidelberg, 1989.
66. E. Pariser. *The filter bubble: What the Internet is hiding from you.* Penguin UK, 2011.
67. P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of the Conference on Computer Supported Cooperative Work, CSCW*, pages 175–186, 1994.
68. F. Ricci and Q. N. Nguyen. Acquiring and revising preferences in a critique-based mobile recommender system. *IEEE Intelligent Systems*, 22(3):22–29, 2007.
69. F. Ricci, L. Rokach, and B. Shapira, editors. *Recommender Systems Handbook*. Springer, 2015.
70. E. Rich. User modeling via stereotypes. *Cognitive Science*, 3(4):329–354, 1979.
71. A. Said, S. Berkovsky, E. W. D. Luca, and J. Hermanns. Challenge on context-aware movie recommendation: Camra2011. In *Proceedings of the ACM Conference on Recommender Systems, RecSys*, pages 385–386, 2011.
72. Y. Shi, M. Larson, and A. Hanjalic. Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges. *ACM Computing Surveys*, 47(1):3:1–3:45, 2014.
73. D. Vallet, A. Friedman, and S. Berkovsky. Matrix factorization without user data retention. In *Proceedings of the Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining, PAKDD*, pages 569–580, 2014.
74. J. Wang, B. Sarwar, and N. Sundaresan. Utilizing related products for post-purchase recommendation in e-commerce. In *Proceedings of the ACM Conference on Recommender Systems, RecSys*, pages 329–332, 2011.
75. M. Xu, S. Berkovsky, S. Ardon, S. Triukose, A. Mahanti, and I. Koprinska. Catch-up TV recommendations: show old favourites and find new ones. In *Proceedings of the ACM Conference on Recommender Systems, RecSys*, pages 285–294, 2013.
76. G. Xue, C. Lin, Q. Yang, W. Xi, H. Zeng, Y. Yu, and Z. Chen. Scalable collaborative filtering using cluster-based smoothing. In *Proceedings of the Annual International ACM Conference on Research and Development in Information Retrieval, SIGIR*, pages 114–121, 2005.
77. H. Yu and M. O. Riedl. A sequential recommendation approach for interactive personalized story generation. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems, AAMAS*, pages 71–78, 2012.
78. T. Zhu, P. Harrington, J. Li, and L. Tang. Bundle recommendation in ecommerce. In *Proceedings of the Annual International ACM Conference on Research and Development in Information Retrieval, SIGIR*, pages 657–666, 2014.
79. P. Zigoris and Y. Zhang. Bayesian adaptive user profiling with explicit and implicit feedback. In *Proceedings of the ACM International Conference on Information and Knowledge Management, CIKM*, pages 397–404, 2006.